Interactive Degraded Document Binarization:
An Example (and Case) for Interactive Computer Vision

Zheng Lu   Zheng Wu   Michael S. Brown
School of Computing; National University of Singapore
luzheng|wuz|brown@comp.nus.edu.sg

Abstract

This paper describes a user-assisted application to perform adaptive thresholding (i.e. binarization) on degraded handwritten documents. While existing adaptive thresholding techniques purport to be automatic, they in fact require the user to perform non-intuitive parameter tuning to obtain satisfactory results. In our work, we recast the problem into one where the user needs only to coarsely markup regions in the thresholded image that have unsatisfactory results. These regions are then segmented and processed locally – no parameter tuning is necessary. Our user study shows that not only do the majority of users prefer our application over parameter tuning, but our final results are better than existing algorithms due to the more targeted processing. While our main contribution is an effective user-assisted application for document binarization, we use this as an example to advocate the need to rethink how many computer vision solutions, notoriously reliant on parameter tuning, can be reworked to exploit meaningful user interaction.

1. Introduction and Motivation

This paper focuses on an application for thresholding degraded documents to separate foreground text from the background. Such image binarization is essential for subsequent post-processing and to enhance readability. Due to adverse handling and storage conditions, as well as age and poor scan quality, the document images exhibit various degradations including non-uniform intensities, shadows, smears and poor contrast. For such degraded images, adaptive thresholding techniques are needed.

Adaptive thresholding methods generally claim to be automatic, however, in actuality, they require the user to tune a parameter, typically referred to as k in the related literature (e.g [5, 7, 10, 13]). To complicate matters, the parameter k has no discernable meaning with respect to the input image or the algorithm itself; obtaining a satisfactory result therefore is a matter of trial and error. Such reliance on parameter tuning represents a major hurdle when developing a real world application and is the impetus of our work.

In our application, we remove the need for parameter tuning and replace it instead with user interaction in the form of coarse scribble marked up on the image by the user. While we do not remove the user from the loop, our studies show that asking the user to perform meaningful interaction on the image is greatly preferred to tuning a "magic number".

The main focus of this paper is to show how we converted a traditional algorithm that requires parameter tuning into one that instead exploits user interaction that is easy, and more importantly intuitive, for the user to perform. In particular, we show how we established a statistical relationship between the input images such that we can compute an initial binarization that is reasonable, but not optimal. User interaction is then used to mark up regions in the image that were incorrectly thresholded, something that currently cannot be automated. Based on the user-supplied markup, we segment out the erroneous region and threshold it individually using regional statistics. Our user study found that not only do users prefer this type of interaction, they also prefer our results.

Although our adaptive thresholding application is a specific one in computer vision, the problem of parameter tuning is general. Thus, the secondary focus of this paper is to advocate the need for many computer vision algorithms to reconsider the reliance on user-tuned parameters and opt instead for meaningful interaction. While this may seem obvious, the use of user interaction is still considered tabooed in computer vision even though the research community is aware of the endemic problem that parameter tuning has on real world applications. As such, we offer this paper an example of one type of problems where interactive computer vision can be successfully applied.

The remainder of this paper is organized as follows: section 2 discusses related work; section 3 describes how we reformulate the adaptive thresholding task to remove the reliance on a parameter and instead incorporate interaction; section 4 shows results including feedback from a user study of 20 participants; section 5 concludes our work.

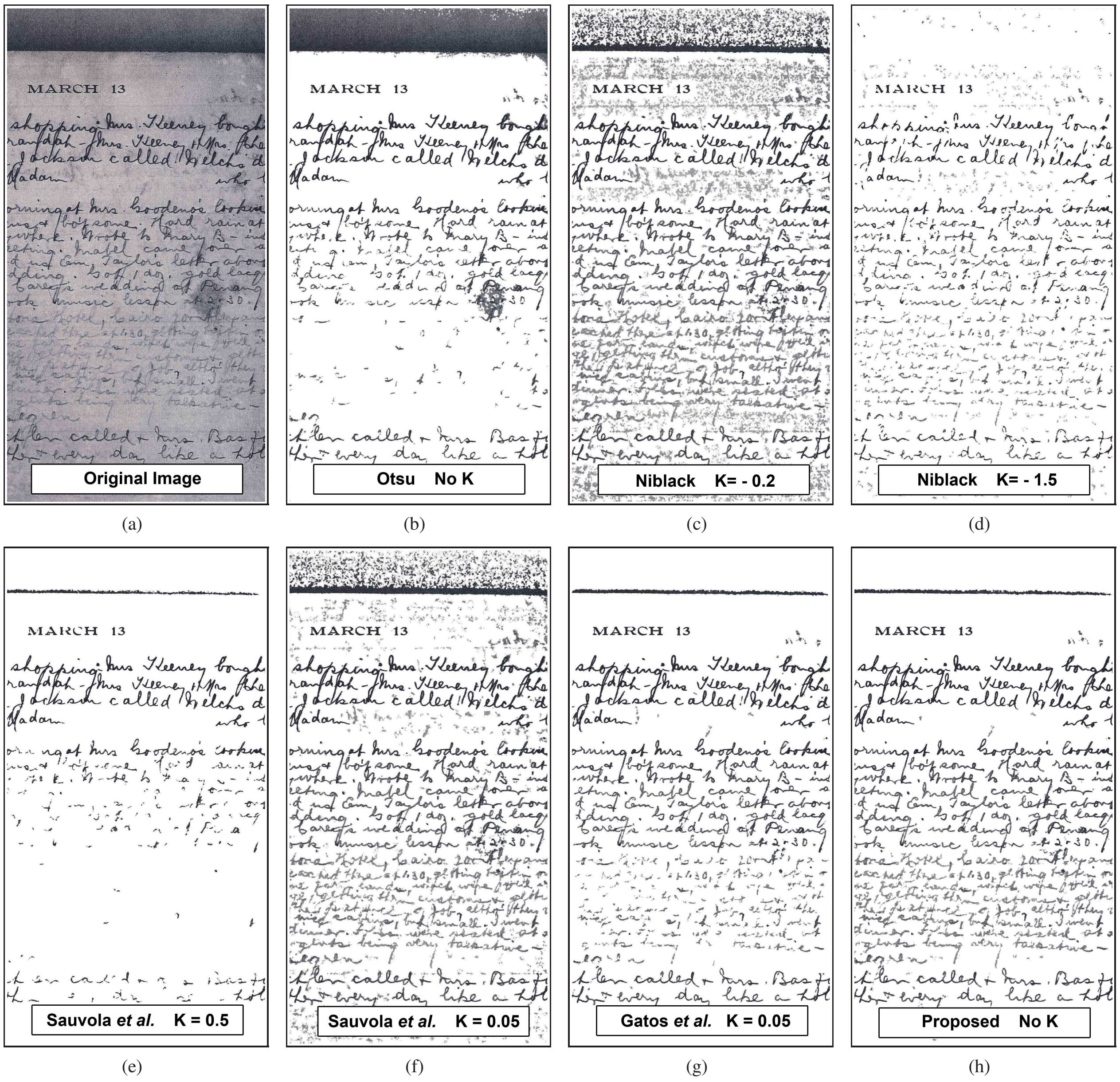


Figure 1. Part of a degraded document image and its binarization results obtained by different thresholding methods. (a) Original image, (b) Otsu's method, (c) Niblack's method (k=-0.2), (d) Niblack's method (k=-1.5), (e) Sauvola and Pietikainen's method (k=0.5), (f) Sauvola and Pietikainen's method (k=0.05), (g) Gatos *et al.*'s method (k=0.05) and (h) our proposed method (no k).

2. Background and Related Work

A short background on this work is discussed followed by an overview of related work pertaining to both document binarization and interactive computer vision.

2.1. Application Background

This work started when we were approached to develop a software application to make a series of journals from the 1920s more legible for transcription by archivists. Each journal page has an entry for the same day over five years. The original primary source has undergone aging and slight water damage and appears spatially discolored. In addition, the text written for each year often appears in a different intensity. We were given access to high-resolution scans of a Xeroxed version of the journal as shown in Fig. 1(a). It was obvious from the beginning that adaptive thresholding would be necessary. Once it was apparent that this would require the user to tune a parameter *per page*, we opted to find a more meaningful way to engage the user.

2.2. Related Work

Adaptive Thresholding Global thresholding techniques are ineffective to binarize document images exhibiting spatially varying degradation. For example, the widely used Otsu's global method [11] can not give satisfactory result for our images as shown in Fig. 1(b) where the light text in the middle part is removed, while the dark shadow at the top remains. To deal with degraded images, various adaptive methods have been proposed. For example, Niblack [10] calculates a pixelwise threshold by shifting a window across the image. For a pixel (x, y), its local threshold $T(x, y)$ is computed using the local mean $m(x, y)$ and the local standard deviation $s(x, y)$ of the intensities inside a window centered at (x, y), and is written as:

$$T(x, y) = m(x, y) + k \cdot s(x, y), \tag{1}$$

where k is a negative user-tuned parameter. Sauvola and Pietikainen [13] modified Niblack's method and propose the following formula:

$$T(x, y) = m(x, y) \cdot (1 - k \cdot (1 - s(x, y)/R)), \tag{2}$$

where R is usually fixed to 128 and k is a positive parameter that needs be tuned by the user. Gatos *et al.* [5] proposed a technique that first uses Sauvola and Pietikainen's method to estimate rough foreground regions (and therefore needs to tune the k in Eq. 2), then estimates the background after foreground removal. The final thresholds are determined using the estimated background.

Non-intuitive Parameter On careful examination of Eq. 1 and Eq. 2, it is apparent that thresholding is based on the local mean and the local standard deviation (std). The local std serves as a measurement of noise, and the parameter k needs to be tuned to control how much of the std should be used to adjust the local mean. From a user's point of view, there is nothing discernable from the input image to suggest what the exact value of k should be. For example, in Fig. 1(c), Niblack's formula with $k = -0.2$ is used. Although -0.2 is the recommended value [10], Fig. 1(c) is exposed to severe background noise. When k is set to -1.5 we can still observe background noise in Fig. 1(d), while part of the light text is removed. This also happens to Sauvola and Pietikainen's method [13] when using the recommended parameter value $k = 0.5$. This is shown in Fig. 1(e) where light text is discarded, while using a much smaller value 0.05 in Fig. 1(f), light text emerges together with distracting background noise. Based on the result of Sauvola and Pietikainen's method with $k = 0.05$, Gatos *et al.*'s method successfully removes background noise in Fig. 1(f), as shown in Fig. 1(g), but the light text becomes fragmented. Additional noise-removing abilities of Gatos *et al.*'s method can be achieved by tuning three more parameters [5], which we do not attempt.

Contrary to these parameter-tuned methods, our proposed interactive method performs good binarization with no parameter as shown in Fig. 1(h). Note that the local window size about each pixel can be considered as a parameter, but it can be intuitively set to contain 1–2 characters.

Interactive Computer Vision Interactive computer vision algorithms exploit user assistance to provide visual information that is difficult to derive automatically. In many cases, the user only needs to denote the regions for processing such as for inpainting or matting (e.g. [1, 3, 14]). In some cases, e.g. object scene extraction, the interaction takes the form of training-data collection [9, 12].

The work most similar to our approach is the ink-bleed reduction algorithm in [6]. In this approach, the authors also opted for interaction over parameter tuning used by related work. The strategy taken in [6] was to avoid parameter-tuned approaches altogether by recasting the problem into a classification problem where markup provided training samples. Our approach does not rely on training data, but instead has the user identify erroneous regions, which is a simple and meaningful task for the user to perform.

3. Interactive Binarization Framework

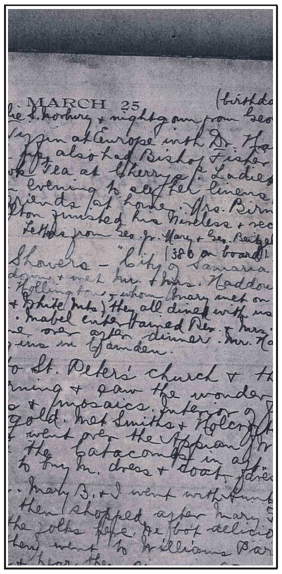
This section discusses our overall framework. Specifically, we discuss how to compute an initial thresholded image based on global image statistics that gives a reasonable, but not optimal, result for the input image. We then show how regional statistics can be used to significantly improve localized regions. User interaction is exploited to detect these local regions to segment them from the initial result.

3.1. Thresholding based on image global/regional statistics

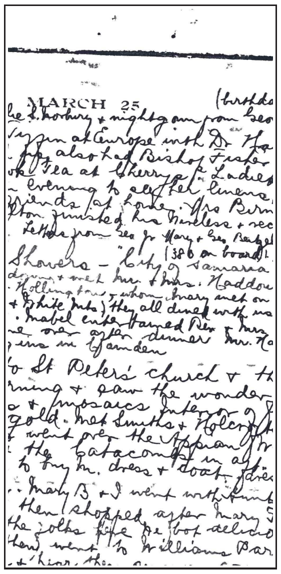
We first show how we can remove the reliance on the parameter k by using global and regional statistics. Not surprising, our formulation is similar format to Eq. 1, however we do not require a parameter.

Globally-Determined Threshold In order to determine a suitable global threshold for an input image, multiple windows were randomly selected across several example images from our document collection. Each window is of size 59×59 and covers at least 1–2 written characters[1]. For each window, we *manually* selected the optimal threshold and recorded the value. For patches where a continuous interval of threshold-values provided good binarization, the median value was recorded. At the same time, the local area statistics used in various adaptive thresholding methods [10, 13, 7], including intensity mean, std and average gradient, were also recorded. For images with a uniform signal-to-noise ratio (e.g. Fig. 2(a)) we observed that there

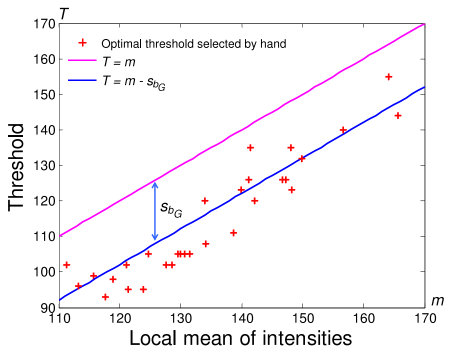
[1]Our images are approximately $2K \times 3K$ in resolution.



(a)



(b)



(c)

Figure 2. Part of a document image and its binarization obtained by the proposed formula in Eq. 3. (a) Image is representative of examples where text intensity changes mildly. (b) Threshold result by Eq. 3 based on the global statistic s_{b_G}. (c) Relationship between the mean of intensities inside a window (x-axis) and the local threshold (y-axis). Symbol '+' is the point for each optimal threshold (selected manually) plotted against the local window's intensity mean. The blue line corresponds to Eq. 3 and shows a good fit to the manually selected thresholds.

is a linear relationship between the optimal thresholds and the local intensity means. This is shown in Fig. 2(c) which plots the manually selected threshold against the local intensity mean of each window. This relationship is not too surprising because existing adaptive thresholding techniques utilize local intensity mean. What is interesting is that we observed the local intensity mean itself (plotted as $T = m$) appears to be globally shifted from the optimal threshold. This suggests that a single global correction could help.

We found that a suitable choice for the global offset is the *overall* global *std* of the background. Referring to Fig. 2(c), we can see that by translating the local intensity means by the image's background *std*, the $T = m$ line shifts to the optimal thresholds. Thus, a reasonable global threshold is:

$$T(x, y) = m(x, y) - s_{b_G}, \tag{3}$$

where s_{b_G} is the *std* of the background intensity of the entire image. To estimate the background, the image is roughly segmented by Otsu's global thresholding method [11] and then s_{b_G} is computed from the background pixels only. Here the subscript G is used to note that term s_{b_G} is a *global* statistic obtained from the entire input image.

While Eq. 3 has the similar form with Niblack's Eq. 1, Niblack's method uses the local *std* and requires the user to tune that by the parameter k. One may wonder if the similar result will be obtained if $k = -1$ in Eq. 1, however, in our experiments we found that the *local std* used in Eq. 1 is not a good estimate of the *global std* and requires k to be tuned. However, unlike Niblack's method, our Eq. 3 is only suitable on images with a uniform signal-to-noise ratio, i.e. the background noise and foreground text (i.e. the signal) are similar throughout the image.

Regionally-Determined Threshold As mentioned, Eq. 3 works for images with mild degradations, but it does not work globally well for images with spatially varying background noise or foreground text intensity. For example, the result of Fig. 1(a) obtained by Eq. 3 is given in Fig. 3(a). The result generates fragmented text for the region with very low foreground-background contrast (i.e. inside the red rectangle). For this low-contrast region, the relationship between the thresholds and the local intensity means are rechecked and displayed in Fig. 3(c). We can see that there is a cluster of $\triangle$ points that represent the optimal threshold in the red rectangular region shown in Fig. 3(c). These $\triangle$ points are clearly far away from the line fitted by Eq. 3 and thus erroneously thresholded. For such regions, we adjusted the thresholds by regional statistics, again trying the *std* of the regional background. This can be expressed as:

$$T(x, y) = m(x, y) - s_{b_R}, \tag{4}$$

where s_{b_R} is the *std* of the *regional* background. The term s_{b_R} is computed from the regional background pixels which are extracted by applying Otsu's method [11] to the region. Updating the result inside the red rectangle in Fig. 3(b) using Eq. 4, we obtain a significantly better result.

Region Extraction via User Interaction Regional adaptive thresholding needs to extract the regions that share similar intensity characteristics. This fits well with our goal, as user-assisted region segmentation is a well known interactive computer vision task. As a result, for our particular application we can reduce the adaptive thresholding problem to one without parameters and that relies on user to only help identify regions where the initial binarization was not successful.

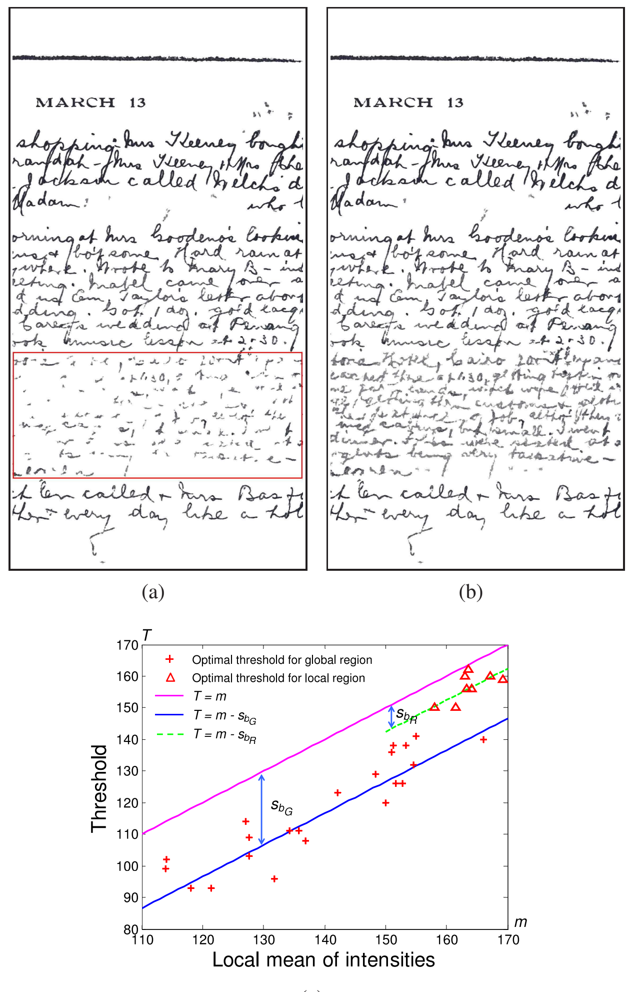


Figure 3. (a) Binary result obtained by using Eq. 3 with the globally-determined parameter s_{b_G} (image is from Fig. 1(a)). (b) The region enclosed by the red rectangle in (a) is updated by using Eq. 4 with the parameter s_{b_R} estimated from the region. (c) Shows the relationship between the intensity mean and optimal thresholds in the image ('+') and in the red rectangle ('△'). The blue line corresponds to Eq. 3. The dashed green line corresponds to Eq. 4.

3.2. User Interaction

Our user interaction is as follows. An global result is first obtained by Eq. 3. In the initial result, the background pixels are set as white and the foreground is in its original colors. If there are regions that are not thresholded correctly, the user can then specify the erroneous region by drawing coarse scribbles on the image in these regions. Based on the user's markup, we extract the erroneous region automatically and then process it using Eq. 4.

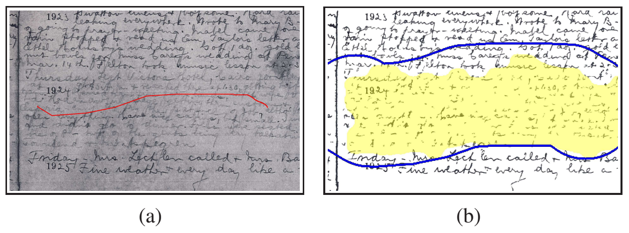


Figure 4. Region segmentation aided by user markup. (a) Original image with markups (red line). (b) Initial large region dilated from user markups (surrounded by blue line) and actual segmented region (highlighted in yellow). Red stroke and blue contour are used as seeds for MRF in order to obtain the segmented region.

3.3. Region Segmentation Based on Markup

Our region segmentation approach is similar to that used in [9] (Lazy Snapping). The significant difference is that we utilize the *results* from the initial binarization in the segmentation process, as opposed to relying only on input pixel intensities. In addition, we only require the user to give examples (i.e. markup) in the "bad region", where Lazy Snapping would require markup in the "good" and "bad" region.

For lack of more descriptive terms, we express the pixels in the region we desire to segment as *Bad* and the region that does not require improvement as *Good*. Their corresponding labels are denoted as $\{\mathcal{B}, \mathcal{G}\}$ respectively. The segmentation problem is formulated as a binary labeling Markov Random Field (MRF) where each pixel p is assigned a label $l_p \in \{\mathcal{G}, \mathcal{B}\}$ (see [8] for details on MRFs). To solve the MRF, the following energy terms are minimized in order to find optimal pixel labels:

$$E = E_d + \lambda E_s, \tag{5}$$

where E_d is the data-cost energy reflecting the likelihood of assigning a l_p to each pixel and E_s is smoothness energy representing the cost of assigning different labels to adjacent pixels. We fixed the weight λ as 0.5.

User-supplied Labels The pixels marked by the user are assigned the label $\mathcal{B}$ and serve as the seeds of the *Bad* set. We do not require the user to provide examples of the *Good* set. Instead, we dilated the user markup and use pixels along this dilated boundary as the seeds points with label $\mathcal{G}$. The rational is that pixels far from the user markups (i.e. the pixels outside the dilated region) are most likely well-thresholded. Here the diameter of the round-shaped structuring element used for dilation is set as $4\times$ the width of the window used for adaptive thresholding. If the initial region does not cover the entire "bad" region, the user can simply supply more markup and generate a bigger region. Fig. 4 shows an example of the seeds of *Bad* set (red-line) and *Good* set (blue-line), together with the initial dilated region. Segmentation is applied to those pixels within the dilated region.

Data Features Two features are used to compute the data and smoothness costs of the MRF. The *background feature*, denoted as BG_p is obtained by averaging all the pixels in the local window used for thresholding that are considered background pixels by Eq. 3. The other feature is computed using the binarization result itself, and is therefore termed the *result feature* (denoted as RE_p). This feature is computed by averaging *all* intensities of the initial binary result using a window $1.5\times$ larger than the window used to average the background. This slightly larger window allows characteristics of the thresholded text to be captured.

Data Cost For each pixel p, the result feature, RE_p is utilized to define the data cost. The average result feature of the user-labeled *Bad* set is denoted as $RE^{\mathcal{B}}$. We assume that the erroneous region we want to segment will have very similar RE_p. However, unlike the *Bad* set, the *Good* set may have more diversity in its characteristics about the boundary. Therefore, the seeds (i.e. boundary pixels around the dilated region) are clustered using K-means clustering [4], where K is set empirically to four and cluster centers are denoted as $RE_n^{\mathcal{G}}, n = 1, 2, 3, 4$. For each pixel p, we define the distance from its RE_p to those of *Bad* seeds as $d_p^{\mathcal{B}} = \|RE_p - RE^{\mathcal{B}}\|$ and define its minimum distance to those of *Good* seeds as $d_p^{\mathcal{G}} = \min_n \|RE_p - RE_n^{\mathcal{G}}\|$. The data cost E_d is then computed as follows:

$$\begin{array}{lll} E_d(l_p = \mathcal{G}) = 0 & E_d(l_p = \mathcal{B}) = \infty & p \in \{\mathcal{G}\} \\ E_d(l_p = \mathcal{G}) = \infty & E_d(l_p = \mathcal{B}) = 0 & p \in \{\mathcal{B}\} \end{array}$$

$$\left.\begin{array}{l} E_d(l_p = \mathcal{G}) = \dfrac{d_p^{\mathcal{G}}}{d_p^{\mathcal{G}} + d_p^{\mathcal{B}}} \\ E_d(l_p = \mathcal{B}) = \dfrac{d_p^{\mathcal{B}}}{d_p^{\mathcal{G}} + d_p^{\mathcal{B}}} \end{array}\right. \quad p \in \{\mathcal{U}\}, \quad (6)$$

where $\{\mathcal{U}\}$ contains all the pixels to be labeled in the dilated region.

Edge Cost For adjacent pixels p and q, the distance between their background features is defined as $d_{pq}^{BG} = \|BG_p - BG_q\|$. We define the edge cost E_s as a function between two pixels p and q, as $E_s(l_p, l_q) = 0$ if $l_p = l_q$, or, $E_s(l_p, l_q) = 1/(1 + (d_{pq}^{BG})^2)$ if $l_p \neq l_q$.

To provide interactive speeds, the MRF described is performed on a 3 : 1 down-sampled version of the image. The optimization result of the MRF is then up-sampled to give the full-size segmentation result. We used the Middlesbury MRF code [15] with the graph-cuts solver [2]. Fig. 4(b) shows a region obtained by using the above approach. Once we have extract the erroneous region, the parameter s_{b_R} is estimated and the new binarization result is computed accordingly by Eq. 4. This entire procedure takes roughly $1s$ to perform and is interactive such that the user can quickly adjust the markup "on the fly" to segment the region they want.

4. Results

We first show results obtained by our application on representative images. This is followed by results from a user study involving 20 participants who used applications based both on our approach and parameter-tuning.

Fig. 5 and Fig. 6 show two representative images from our collection – these also serve as excellent examples of documents that require adaptive thresholding. The top row shows the initial binarization result. Note that faint text is fragmented or even totally removed. The user markup and segmented region are shown in the bottom. The segmented regions are correctly segmented and the new results show a significant improvement.

We performed a user study to substantiate our claims that our interactive approach is desirable over parameter tuning. We asked 20 participants to perform binarization on documents using two different applications: *App1*-[parameter-tuning] (using Sauvola and Pietikainen [13] method) and *App2*-[our interactive markup]. Users were not told which application was ours, instead they were told the purpose of the study was to determine which application was preferred.

Users were given a short training session on how to use both applications. *App1* involved an interface that allowed the user to adjust k (only called "parameter" in the actual application) using either a slider, spin-dial, or manual entry. Due to the authors handwriting style, the text in the journals are very difficult to read, even after binarization, thus we asked the users to only perform the binarization to what they felt was the best quality they could obtain. After a training session, each participant performed binarization on three (3) complete documents.

For each participant we recorded: 1) average time required to obtain a result; 2) which application the user preferred; 3) which result they preferred (i.e. result obtained by *App1* or *App2*). Both apps took user roughly the same amount of time (*App1*=90s) vs. (*App2*=100s) with the parameter-tuning being slightly faster, see Fig. 7. However, even though slightly faster, 90% of the participants (18 out of 20) preferred our approach in *App2*, and 80% felt our results were better (16 out of 20), see Fig. 8. Note that two of the participants who felt the parameter-tuning produced better results still preferred using our application.

In addition, we had feedback from two archivists who are involved with these materials. A formal user study was not performed on the archivists, however, we gave them the opportunity to use both applications. Both archivists agreed, without hesitation, that our interactive tool was significantly better. One archivist commented that the interactive tool also fits how text transcription would be performed, "chunk by chunk" – thus providing the ability of regional processing is inline with the real usage pattern.

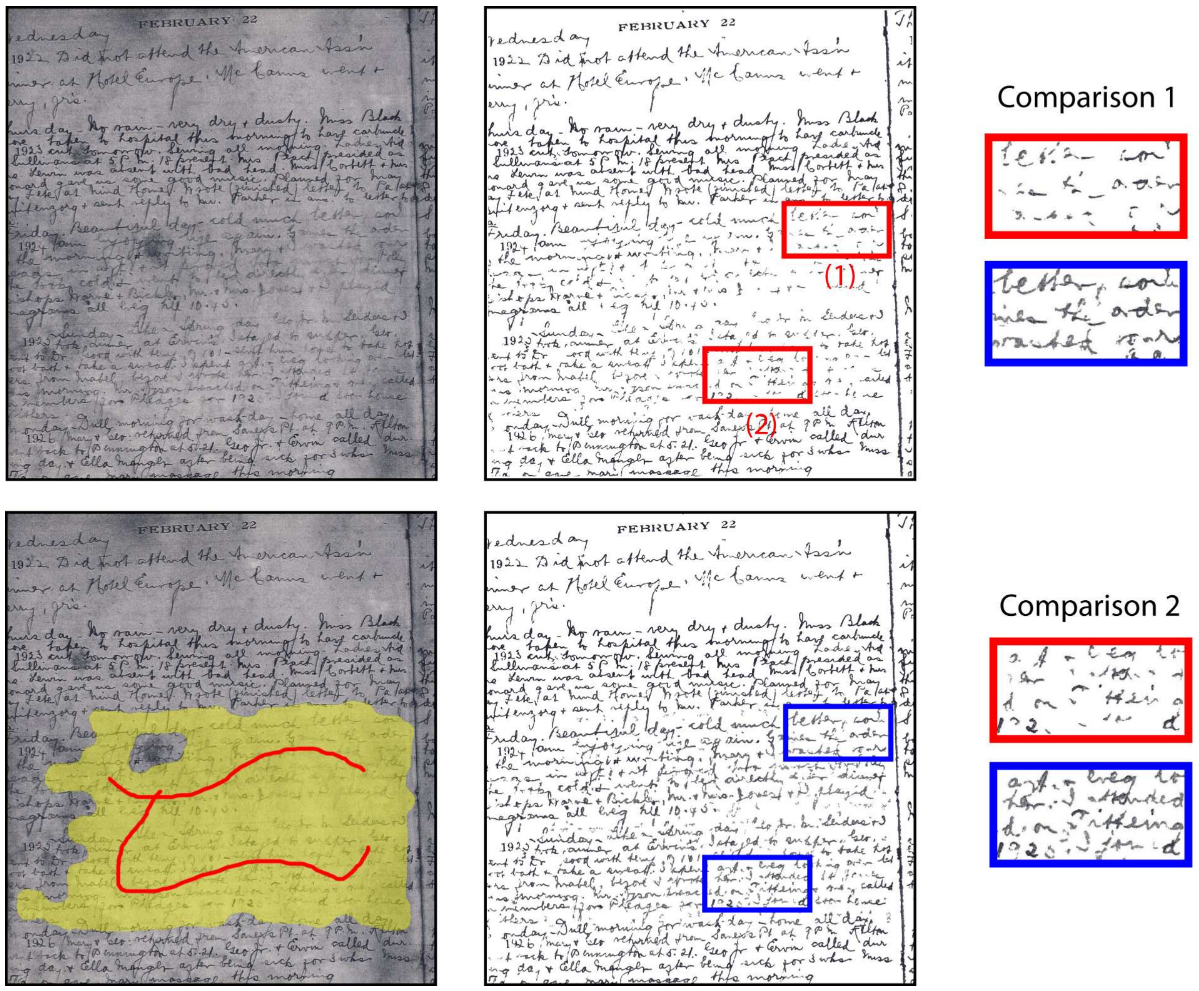


Figure 5. (Top): First image shows the input image. The initial binarization result is shown in the second image. (Bottom) User markup within segmented region, new binarization result. Zoomed comparison are shown on the right.

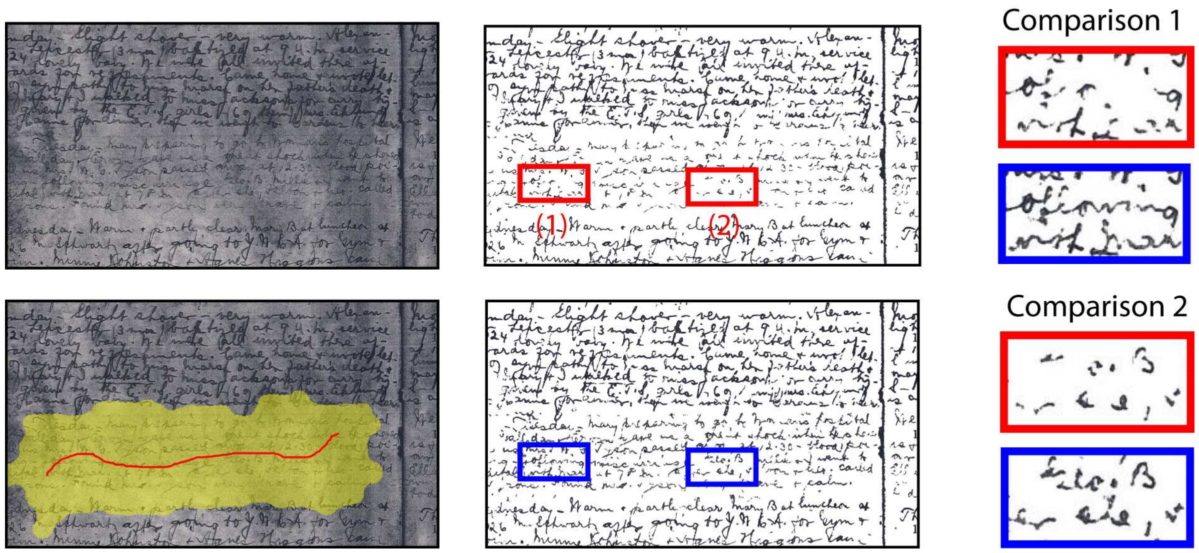


Figure 6. (Top): First image shows the input image. The initial binarization result is shown in the second image. (Bottom) User markup within segmented region, new binarization result. Zoomed comparison are shown on the right.

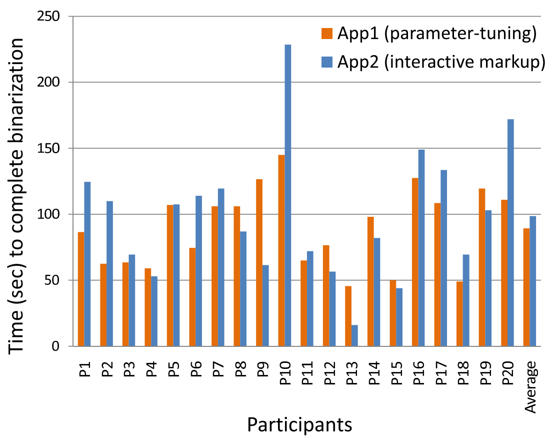


Figure 7. Timing results from 20 users. On average App1 (parameter-tuning, 90s) is slightly faster than our App2 (interactive markup, 100s).

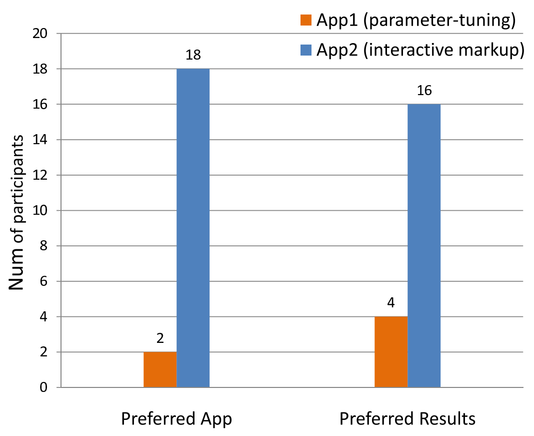


Figure 8. Preference by the user as to which application they would prefer to use, and which result they preferred.

5. Discussion and Summary

We have demonstrated an effective user-assisted approach for document binarization. While some computer vision tasks must be fully automated (e.g. real-time applications), there are many that do not. The degraded nature of documents targeted in our application are not suitable for batch processing and existing parameter-based techniques require fine-tuning for each input. This makes our application an excellent case for exploiting interactive computer vision. As our user study shows, asking the user to perform meaningful mark up on the input image was greatly preferred to tuning a "magic number". In addition, when the user's help is enlisted to localize regions, local processing may be an option to produce better results, as done in our application.

We also point out two potentially non-obvious insights exploited in this paper that may be useful when converting existing parameter-tuned algorithms to ones relying on user interaction. First is the strategy of trying to relate the parameter(s) to the input data even if only *sub-optimal* results can be obtained. This strategy allows the problem to be reduced into one where the user needs to only mark erroneous regions, a task that is generally easy to do. The second insight is the idea of using the erroneous result itself for tasks such as region segmentation as described in Section 3.3. In particular, we were able to utilize the additional feature data from the sub-optimal result to segment this region more effectively than existing user-assisted segmentation routines.

References

[1] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In *SIGGRAPH*, 2000.

[2] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 26(9):1124–1137, September 2004.

[3] Y. Y. Chuang, B. Curless, D. H. Salesin, and R. Szeliski. A bayesian approach to digital matting. In *CVPR*, 2001.

[4] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience Publication, 2000.

[5] B. Gatos, I. Pratikakis, and S. J. Perantonis. Adaptive degraded document image binarization. *Pattern Recognition*, 39:317–327, 2006.

[6] Y. Huang, D. Xu, and M. S. Brown. A framework for removing ink-bleed in old documents. In *CVPR*, 2008.

[7] C. G. Leedham, C. Yan, K. Takru, J. H. N. Tan, and L. Mian. Comparison of some thresholding algorithms for text/background segmentation in difficult document images. In *ICDAR*, 2003.

[8] S. Li. *Markov Random Field Modeling in Image Analysis (2nd Edition)*. Springer-Verlag, 2001.

[9] Y. Li, J. Sun, C.-K. Tang, and H. Y. Shum. Lazy snapping. *ACM Transactions on Graphics (SIGGRAPH)*, 23(3):303–308, 2004.

[10] W. Niblack. *An Introduction to Digital Image Processing*. Prentice-Hall, Englewood Cliffs, NJ, 1986.

[11] N. Otsu. A threshold selection method from gray level histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 9:62–66, 1979.

[12] C. Rother, V. Kolmogorov, and A. Blake. "grabcut" — interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (SIGGRAPH)*, 23(3):309–314, 2004.

[13] J. Sauvola and M. Pietikainen. Adaptive document image binarization. *Pattern Recognition*, 33:225–236, 2000.

[14] J. Sun, J. Jia, C. Tang, and H. Shum. Poisson matting. *ACM Transactions on Graphics (SIGGRAPH)*, 23(3):315–321, 2004.

[15] R. Szeliski, R.Zabih, D. Scharstein, O. Veksler, and V.kolmogorov. A comparative study of energy minimization methods for markov random fields. In *ECCV*, 2006.