ROMANIAN JOURNAL OF INFORMATION SCIENCE AND TECHNOLOGY
Volume **17**, Number 3, 2014, 219–229

Applying Localized Otsu for Watershed Segmented Images

Costin-Anton BOIANGIU, Andrei TIGORA
University "Politehnica" of Bucharest, Romania
E-mail: costin.boiangiu@cs.pub.ro, andrei.tigora@cti.pub.ro

Abstract. This paper describes an image binarization method that applies localized Otsu thresholding to irregular regions of images, determined through watershed segmentation. Traditional localized binarization techniques work on square regions, which group together pixels of different origins. This is a problem, as there may not always be available a comparator for objects from different classes. In order to solve this problem, we use a method that first selects the objects (like shadow and light regions) and then performs binarization on each individual object in order to expose its characteristics. For more accurate results, an inter-scale segmentation and binarization method is proposed.

1. Introduction

Binarization is one of the most used and easily recognizable image manipulation procedures. It classifies the image pixels as belonging to the foreground or background. It can therefore be viewed as a two class segmentation procedure.

Converting a grayscale image to binary may not seem to be a great idea, given the drastic information loss that it implies. However, this loss is important as it creates a simplified image that can be later processed by highly specialized algorithms that could not otherwise efficiently deal with the complexities associated with a colored or even grayscale image.

While for humans it is in most cases quite obvious to which of the two classes a certain pixel should be assigned to, from a computational point of view the things are much more difficult. This is because a pixel value can be part of the background in a region of the image, but in a different region, the same value represents the foreground. It can happen for entire actual areas to be very similar, but they should be interpreted differently. That is why the selection of the windows inside which the statistics are calculated must be optimal.

The research carried in this field, however, considers the most important aspect as what formulas should be applied on a known window, leaving the selection of its size to the user.

This paper continues the previous work of Tigora [1] and attempts to exploit a different schema for binarizing a watershed segmented image. The approach presented here emerges from the proposed ideas in the future work section of the mentioned paper, namely using multiple scales for segmentation and binarization in order to have final clusters that contain both foreground and background elements inside them, in order for Otsu to give correct results. Hopefully, this will minimize the number of white regions marked as black, a problem mentioned in the first paper.

The rest of the article is structured as follows: the following section is dedicated to presenting previous work related to binarization. Section II presents a detailed description of the algorithm, while Section III is reserved for results and performance evaluation. The last part contains the conclusions.

2. Previous Work

The binarization of an image can be handled in two ways [2]. The first solution is based on thresholding, which implies computing a threshold against which the pixel is compared to and cataloged as either black or white. The second approach is dithering, which replaces pixel regions with patterns that best approximate the initial pixel distribution from a "summation" point of view or from the error minimization point of view.

Among these, the widest attention is given to thresholding, as the other produces visually appealing results, for printing purposes, and is less reliable for automatic processing.

Determining the threshold can be treated as a global problem [3], taking into account the entire image, or can be handled locally, determining and applying a threshold on narrow sections of the original image [4]. As expected, either approach has its own advantages and disadvantages. Whereas global methods are usually faster, they tend to handle variations in image lightness poorly. Localized methods, on the other hand, produce better results with irregular illumination, but tend to be much slower and also more susceptible to localized noise.

2.1. The Global Binarization Approach

From the global algorithm's class, Otsu's algorithm [3], created in 1979, is probably the most known and used binarization algorithm, because it has a strong mathematical foundation to why a threshold should be the proposed one: it chooses the threshold that maximizes the inter-class variance or minimize the intra-class variance. The inter-class variance σ_b is defined as

$$\sigma_b^2(t) = \omega_1(t)\omega_2(t)[\mu_1(t) - \mu_2(t)]^2 \tag{1}$$

where $\omega_x(t)$ is the class probability and $\mu_x(t)$ represents the class mean. The class probability is computed as:

$$\omega_1(t) = \sum_{i \leq t} c(i), \quad \omega_2(t) = \sum_{i > t} c(i) \tag{2}$$

whereas the class mean is given by:

$$\mu_1(t) = \sum_{i \leq t} c(i) \cdot i, \quad \mu_2(t) = \sum_{i > t} c(i) \cdot i \tag{3}$$

and $c(i)$ is a function indicating the number of pixels in the image that have the value i.

The threshold will be t^* such that:

$$t^* = \arg\max_t \sigma_b^2(t) \tag{4}$$

2.2. The Local Binarization Approach

The local algorithms received a bit more attention, as the global problem seems to be resolved from the theoretical point of view. The most popular of the class are Niblack [5] with its variations. The main idea is to have a local window slide across the image, inside which statistics are calculated, and based on those statistics a value can be assigned to the center pixel. Niblack considers that the window should be cut at the mean luminosity, but if the contrast is small, the threshold should be lowered and if the contrast is very good, the threshold can be safely raised. This method manages to work well in regions containing both foreground and background, but in background-only windows it produces significant noise.

In order to fix this issue, the NICK [6] algorithm lowers the threshold by using a modified measure of spreadness while Sauvola et al. [7] add a global parameter to the equation, the dynamic range of the entire image, in order to lower the threshold in background only areas. But these methods also lower the threshold for foreground regions, thus causing misclassification of foreground pixels. Wolf and Jolion [8] introduce a standard deviation constructed over all the windows in the image and use this parameter locally, to estimate a relative local contrast. Feng and Tan [9] noticed that a single window cannot offer sufficient information, even when including entire image global statistics, so they decide to use two windows, each with many statistics and parameters. However, the proposed solution introduced an unacceptable problem: a large number of parameters that had to be set manually. In [10] the authors collect various parameters that can be used in binarization for OCR purposes. Aside from the aforementioned ones, they also consider: uniformity, non-uniformity, weighted variance, uniform variance and unbiased weighted variance. Instead of choosing predefined weights, the authors imply a learning algorithm that chooses values from prior experience.

2.2.1. The Window Size

As it can be noticed, there was no mention of locality size or shape until now. The explanation is that computing complex statistics on predefined window sizes is alone very time consuming. Doing this for multiple window sizes is unusable; however it can be approximated.

In [11] the authors simulate different window sizes by rescaling the entire image. Local Sauvola is applied on a scaled-down version of the image, with predefined parameters in order to filter out as much noise as possible. Then progressively larger resolutions are used with parameters that favor precise detection. Only pixels that originally appeared on the previous scales and that are connected are kept. The partial result is relaxed with every scale, resulting in a precise binarization without the usual noise local methods tend to induce. But as Sauvola alone is time consuming, authors also propose an optimization using grid methods: all parameters (not the function itself) are calculated only in some pixels (the grid). For the other pixels the parameters are interpolated and the threshold formula is applied on the interpolated values. Beside the mean and standard deviation needed in the computation of the threshold, they also included the scale as parameter suitable for interpolation.

A more direct approach is presented in [12]. A first step is taken to try to binarize the image, but the binarization also offers a "confusion factor" which is then used to choose from three window sizes: large windows are used for low contrast (probable background) or high probability of confusion (to gather more information); medium windows for normal images and small sizes for regions with high amount of noise. In order to obtain even more region information, a secondary smaller window is also calculated if the region is uncertain. In [4] the problem is taken very far, and for every pixel all the window sizes are tested. It is however not mentioned the complexity of the algorithm, but the result offered proves the importance of a good window-size selection.

2.2.2. The Window Shape

Until now the problem of the window size was taken into consideration, but shape of the locality was mostly considered as unimportant. In order to include shape information, but still work with rectangular images, usually learning algorithms are used, in order to recover shapes from similarly learned regions.

In the case of [13], during the training phase, the algorithm constructs histograms of the shapes of correctly binarized image regions. The shape image is constructed from the binarized image by setting the gray value of a foreground pixel to its distance from a background pixel. A histogram is then computed on this distance image and remembered for future comparisons. When applying the algorithm, localities are no longer constructed as sliding windows. The image is recursively split in half until a predefined size and the binarized tiles with different thresholds are offered to the AI module. From those, only the thresholds offering the highest similarity with the learned histograms are selected. In order not to induce discontinuities at the region borders, the threshold is calculated only for the center pixels of the regions and the threshold values are interpolated for the rest of the pixels.

All the aforementioned methods imply an excessive risk of having multiple different objects in the same region and eliminating some of them. In order to overcome this shortcoming, in [14] the authors don't try to do a binarization, but rather a segmentation that preserves all the objects. This is done with a watershed segmentation that is stopped when water ponds are half filled (in order to separate noise between close-by objects). This process generates ponds or blobs. As a final step, a classification of the blobs is done, considering as metrics the quantity of water in the blobs and average gray level, which are fed to a neural network. The results are very good and don't require manual parameters. This algorithm is the closest to what we propose, but our aim is different: to find the details inside blobs, and thus we will keep the ponds as large as possible.

Because of the generality, simplicity and beautiful explanation of Otsu's algorithm, some adaptive Otsu methods were proposed and work well [15]. That is why we will keep Otsu as the thresholding method for our approach.

2.3. The Watershed-Based Binarization Approach

In [1] Tigora applied the Otsu algorithm not on an entire image, but on individual irregular regions obtained through watershed segmentation. Segmentation can be viewed as a generalization of binarization, as it allows the classification of the image into multiple classes, not just background and foreground. These classes are called segments and are usually continuous regions that have some properties that clearly differentiate them from other regions. Watershed segmentation was developed by Meyer [16] and functions on the flooding principle, with "pools" emerging in low value bits that end up neighboring other pools as the water level rises. This algorithm needs seed points, or rain drops from which to start flooding. The seeds used in the proposed approach are the image edges.

Edge detection is a research domain in itself, with the aim being to discover abrupt changes in an image. The idea behind this is that given two objects, their overlapping regions will present a much stronger discontinuity than any other region inside either of the objects. Canny's edge detection algorithm [17] is one of the more successful algorithms for edge detection. This is a multi-step algorithm that works as follows:

1. The noise is filtered out through a 5 by 5 Gaussian filter with $\sigma = 1.4$
2. An absolute gradient is computed using the Sobel operator, as an estimation of the edge strength
3. The edge direction is determined with the help of axis aligned gradients
4. Each edge is classified under one of 4 possible directions: along the axes or along the two diagonals
5. Non-maximum suppression is applied along the edge and resulting in the removal of border pixels.
6. A hysteresis based processing is applied so as to eliminate streaking. The margins are detected starting from a high value pixel, whose value is greater than

a primary threshold T1, and determining all of its adjacent pixels that are no less than a secondary threshold T2.

Inside each of the watershed segments an Otsu binarization was applied to bring out the details.

Following the computing of the thresholds, a threshold map was created for the regions, which was then smoothed with a Gaussian filter in order to minimize the discontinuities between regions.

For this paper, it was decided to maintain the segmentation approach and compute individual thresholds using Otsu's method for each segment. The difference from the previous method comes from the way this value is adjusted, not by correlating it with results from adjacent segments, but by correlating it with thresholds from segments in a reduced version of the original image in order to benefit from a more global view. In the case of segmentation, this is a larger region of the same object.

3. Algorithm Description

The input of the algorithm is a grayscale image and the end result is a black and white image of the same size. A step by step description of the processing is given in the following paragraphs.

1. The initial image is passed through a Gaussian or median filter. This procedure will eliminate some of the noise, which in turn will reduce the effect of over segmentation that is usually associated with the watershed processing.

2. Canny's algorithm is applied on the image from step 1 to determine the edges. The image has already been smoothed, and is further smoothed by Canny, but that is not an issue. The margins that are to be detected will only serve as initial reference points for segmentation, so extreme accuracy is not necessary, as more edges would produce more segments.

3. The edges are dilated and the resulting image is inverted in order to obtain the regions that are furthest from them; these areas will represent markers for the watershed segmentation. These markers will be expanded into segments that will end up covering the entire image.

4. Starting from the previously determined markers, functioning as hypothetically minimum points, the image is segmented using the watershed algorithm.

5. For each newly determined segment, a threshold is computed using Otsu's algorithm. If there are several values that satisfy the condition, they are averaged.

6. The original image is downsampled to a quarter of its size and steps from 1 to 5 are applied once again, obtaining a new segmentation and new thresholds. The idea behind this step is that the original full scale image has lots of details that are not essential for “understanding" the image as a whole. So, by losing some of the subtleties of the initial image, a new perspective of the image is presented, with a focus on the “macro" features.

7. For each pixel in the original image, the associated region and threshold are determined; next, its corresponding pixel in the downsampled image is identified. Now, the region and threshold associated with this latter pixel are also determined. The two threshold values are then used to compute a third threshold, by averaging the previous two. This newly computed threshold is the one used to classify the current pixel as either foreground or background.

Optionally, the contours obtained using Canny's algorithm can be included, for a better separation of the shapes in the image. This method should lead to a great reduction of the noise in the image, especially specular noise.

4. Results

The algorithm was implemented in C++, using the OpenCV image library. For smoothing the image several filters were tested, such as the median and Gaussian filter. Eventually, the median filter was chosen as the most appropriate because it produced on average fewer segments than the other solutions.

The time required to process an image varies from half a second for a 512×512 pixels image to more than three seconds for a 1024×1024 one. The execution times of the varying components are as follows:

- median blur: on average tens of milliseconds, no more than 20 ms
- Canny edge detection: roughly three to four times larger than the median blur, no more than 50 ms
- dilation: roughly 100 ms
- watershed computation: 50 to 250 ms (512×512 to 1024×1024)
- Otsu thresholding: 300 to 3000 ms

These times are by no means absolute, as the lines between the steps get blurred during the implementation, but one thing is certain: the thresholding is the most time consuming step of the method. This was to be expected, given the irregular form of the segments and the nature of the Otsu algorithm. Furthermore, the time performance is not entirely predictable, as the number of Otsu thresholding operations is dependent on the number of segments found in the image, rather than on the image size.

As can be seen in Fig. 1, the segmentation produces different results in the second reduced picture compared to the original one. The number of segments is in fact reduced from 27 to 22. However, it is not a simple issue of merging of adjacent segments. While some segments do indeed merge to form bigger ones (as is the case with the shoulder and the hair in the picture), other segments split (as is the case of the leftmost vertical segment) or are split and grouped in completely new configurations.

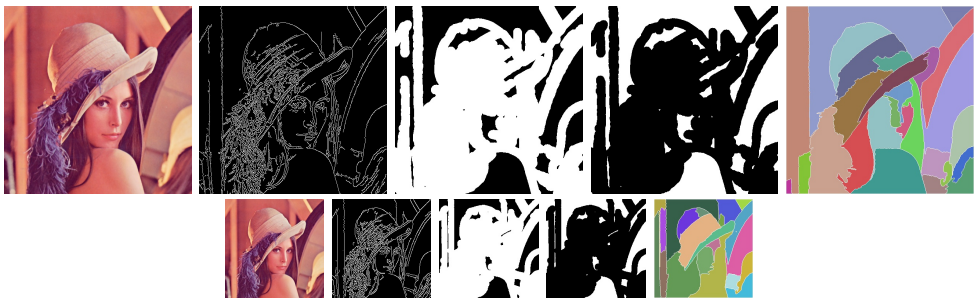

Fig. 1. Intermediary images, from left to right, top to bottom: a. Initial image; b. Canny image; c. Dilated contours; d. Watershed markers; e. Watershed image; f. Canny of reduced image; g. Dilated contours of reduced image; h. Watershed markers of reduced image; i. Watershed segmentation of reduced image.

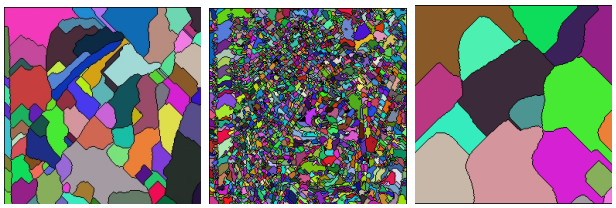

Fig. 2. Segmentation samples as presented in [1]: normal segmentation, oversegmentation, undersegmentation.

Another thing to be noted is that the image can be viewed as under-segmented, as a result of the initial blurring, according to [1] (see Fig. 2). The number and size of segments is clearly closer to the third pattern, than the first one, and nothing like the second segmentation pattern.

The resulting binarized "Lena" can be seen in Fig. 3. The image obtained through the current method is a lot lighter than the old watershed single scale approach, and even than its global Otsu counterpart. And while it seems to better outline some edges, as a result of the Canny based segmentation, it also lacks continuous black blocks filling certain contours (the arched frame behind the character should be black, yet it is not). This method is not useful for natural images as it is not able to select entire objects as one, but it rather tries to find the details from the objects.

Fig. 3. Comparison between various binarization methods: a. Otsu binarization; b. Tigora binarization; c. Current binarization method; d. Current method no Canny.

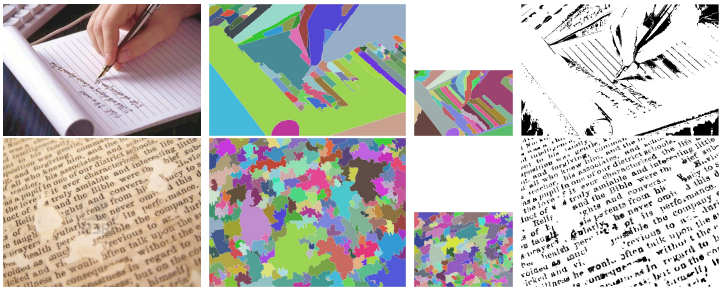

Fig. 4. First column: original image; second column: segmentation of original image; third column: segmentation of downsampled image; fourth column: binarized image.

Figure 4 presents more binarization examples with the segmented counterparts for both the full scale and downsampled images. In the case of the "hand" image, it is quite interesting to notice that there are more segments that correspond to the rows in the notepad in the downsampled version, than there are in the full scale counterpart. Also, some of the small segments that appear in the segmentation of the original image do not show up in the segmentation of the reduced version, which is in line with the prediction that the simplification of the image will focus the "attention" towards larger features. In the binarized result we see a few regions that contained only background (the lines beneath the hand) despite the multi-scale approach and due to the Otsu thresholding many pixels were classified wrongly as foreground. For the second image though, the segmentation results are less intuitive, due partially to the nature of the image contents. The mostly white and black image of the printed paper will generate lots of markers, for both the small and large versions. From the end result perspective, the binarization of the first image still is unable to "fill" black areas, for example, the top of the hand. For the second image, given that there are no significant continuously dark regions, the result is closer to the ideal human binarization.

5. Conclusion

We experimented with a binarization algorithm that tries to pick up the details inside objects instead of binarizing inside rectangle regions with no meaning. We optimized the results by considering two scales for segmentation in order to see both local object segments and the objects in their entirety.

As shown throughout literature, the choice of markers is important for the segmentation of the image. Therefore, having an automated, unbiased mechanism that relies on local image features produces improved results.

Differentiating from existing methods, the main purpose was not to combine letter fragments into binarized letters, but rather find large regions with similar characteristics on which a parameter free good binarization method can be applied safely. That is why we need large segments and try to avoid small ones. In order to counteract the over-segmentation, that usually occurs when performing watershed segmentation, the initial image was smoothed by applying a median filter on the original image.

The proposed method significantly reduces the noise in the binarized image, while retaining much of the information of the original image. This method is not useful for binarizing natural images, for object classification, as it tries to split the actual objects in order to find the textural details. However it is appropriate for binarizing nonuniform lit documents as it tries to select letters from each lighting condition separately. Some artifacts still remain, but the multi-resolution approach improves considerably the previous method on the aforementioned types of images.

References

[1] TIGORA A., *A Watershed-Based Thresholding Approach for Image Binarization*, International Journal of Circuits, Systems and Signal Processing, Vol. **8**, pp. 246–252, 2014.

[2] BOIANGIU C. A., BUCUR I., TIGORA A., *The Image Binarization Problem Revisited: Perspectives and Approaches*, Journal of Information Systems and Operations Management, Vol. **6**, Issue 2, pp. 195.

[3] OTSU N., *A Threshold Selection Method from Gray-Level Histograms*, IEEE Transactions on Systems, Man, and Cybernetics, Vol. **9**, No. 1, pp. 62–66, 1979.

[4] BOIANGIU C. A., OLTEANU A., STEFANESCU A., ROSNER D., TAPUS N., ANDREICA M., *Local Thresholding Algorithm Based on Variable Window Size Statistics*, Proceedings CSCS-18, The 18-th International Conference on Control Systems and Computer Science, Vol. **2**, pp. 647–652, 2011.

[5] NIBLACK W., *An Introduction to Digital Image Processing*, Prentice Hall, Englewood Cliffs, 1986.

[6] KHURSHID K., SIDDIQI I., FAURE C., VINCENT N., *Comparison of niblack inspired binarization methods for ancient documents*, Proc. 16th International Conference on Document Recognition and Retrieval, USA, 2010.

[7] SAUVOLA J., SEPPANEN T., HAAPAKOSKI S., PIETIKAINEN M., *Adaptive Document Binarization*, 4th Int. Conf. On Document Analysis and Recognition, Ulm, Germany, pp. 147–152, 1997.

[8] WOLF C., JOLION J. M., *Extraction and Recognition of Artificial Text in Multimedia Documents*, Pattern Analysis and Applications, pp. 309–326, 2003.

[9] FENG M. L. , TAN Y. P., *Contrast adaptive binarization of low quality document images*, IEICE Electron. Express, Vol. **1**, No. 16, pp. 501–506, 2004.

[10] RAMIREZ-ORTEGO M. A., TAPIA E., ROJAS R., CUEVAS E., *Transition thresholds and transition operators for binarization and edge detection*, Pattern Recognition, Vol. **43**, No. 10, pp. 3243–3254, 2010.

[11] MOGHADDAM R. F., CHERIET M., *A multi-scale framework for adaptive binarization of degraded document images*, Pattern Recognition, Vol. **43**, Issue 6, pp. 2186–2198, June 2010.

[12] BATAINEH B., ABDULLAH S. N. H. S., OMAR K., *An adaptive local binarization method for document images based on a novel thresholding method and dynamic windows*, Pattern Recognition Letters, Vol. **32**, Issue 14, pp. 1805–1813, 15 October 2011.

[13] SHI J., RAY N., ZHANG H., *Shape based local thresholding for binarization of document images*, Pattern Recognition Letters, Vol. **33**, Issue 1, pp. 24–32, 1 January 2012.

[14] VALIZADEH M., KABIR E., *An adaptive water flow model for binarization of degraded document images*, International Journal on Document Analysis and Recognition (IJDAR), Vol. **16**, Issue 2, pp. 165–176, June 2013.

[15] MOGHADDAM R.F., CHERIET M., *AdOtsu: an adaptive and parameter less generalization of Otsu's method for document image binarization*, Pattern Recogn., Vol. **45**, Issue 6, pp. 2419–2431, 2012.

[16] MEYER F., *Un algorithme optimal pour la ligne de partage des eaux*, 8eme Congrès de Reconnaissance des Formes et Intelligence Artificielle, Vol. **2**, 1991.

[17] CANNY J., *A computational approach to edge detection*, IEEE Trans. Pattern Analys and Machine Intelligence, Vol. **8**, pp. 679–714, 1986.