

Original Research Paper

Local Features Supported by the Complement Feature for Image Segmentation

Salah Ameer

Ontario Colleges, Canada

Article history

Received: 03-06-2020

Revised: 01-07-2020

Accepted: 07-07-2020

Email: salahameer@alumni.uwaterloo.ca

Abstract: An Eigen formulation is proposed for image thresholding/segmentation. A vector composed of local features, normalized intensity of each pixel and that of the neighboring pixels, is used to represent each pixel. A “complement” component is appended to this vector to produce a “unit” vector. The auto-correlation matrix is computed for each pixel in the image using this unit vector. The first component (corresponding to the intensity of the current pixel) from all Eigen vectors, obtained from the auto-correlation matrix, are used as multi-level thresholds. Similar procedure can be adopted using powers of the current pixel intensity value. In general, more than one threshold can be obtained. Results on a wide range of images are demonstrated to show the effectiveness of the proposed schemes.

Keywords: Image Thresholding, Image Segmentation, Eigen Value

Introduction

Multi-level image thresholding is essential in many image segmentation schemes needed by many computer vision tasks. The ultimate goal is to delineate the image in such a way to obtain useful descriptions of the objects comprising the scene. To achieve this goal, many algorithms has been (and still being) developed. Details regarding categorization of these algorithms and the feature space used can be found in many traditional survey papers such as (Sezgin and Sankur, 2004). In fact, the field is so vast and diverse that there are survey papers on a single subcategory e.g., (Ilea and Whelan, 2011; Peng *et al.*, 2013; Unnikrishnan and Hebert, 2005).

In this study, the image segmentation problem is considered as a multilevel thresholding task. A simple but effective Eigen structure is proposed as a solution. The schemes are based on a recent work (Ameer, 2020).

Method

Without loss of generality, the intensity of the original image is normalized to the interval [0,1] (or [-1, 1]) and concatenated to produce a column vector of size $N \times 1$, N is the number of pixels in the image. Each pixel can be represented by any subset from its 8-neighborhood. For ease of notation, a description will be given for using one neighbor, more neighbors can be used and the description can be easily generalized. Each element is then extended to be represented by a “unit” vector type given by:

$$G_k = \begin{bmatrix} g_{i,j} & g_{i-1,j} & \sqrt{1 - w_1 g_{i,j}^2 - w_2 g_{i-1,j}^2} \end{bmatrix} \quad (1)$$

where, w_i is a weighting vector that sums to one and $k = 1, \dots, N$. In this study, the uniform, power dependent and variance weighting have been implemented with minor differences in their performance. The author is not confident that other schemes can provide better performance.

G is now an $N \times n$ vector, n is the size of the vector used in Equation (1), $n = 10$ for a 3×3 neighborhood. An auto correlation matrix (AG) of size $n \times n$ is then constructed from G as:

$$A_G = G^T G \quad (2)$$

Solving the Eigen formula:

$$A_G V = \lambda V \quad (3)$$

The Eigen vectors of A_G represent the axes of inertia for the data set. The largest vector V_{\max} (corresponding to the maximum Eigen value λ_{\max}) points toward the direction of maximum inertia (Ameer, 2020). The first components of each resulting Eigen vector, corresponding to the intensity of the central pixel, is the most suitable for performing the segmentation. Hence, the first component of all Eigen vectors are selected as thresholds. Care should be taken to remove out of range ([0,1] or [-1, 1]) values to overcome unnecessary computations.

An alternative scheme would be to extend Equation (1) to have a complement for each component as shown:

$$G_k \left[g_{i,j} \sqrt{1 - g_{i,j}^2} s(g_{i-1,j}) \sqrt{1 - g_{i-1,j}^2} \right] \quad (4)$$

Effectively, there can be many schemes depending on the number of neighbors included in forming Equation (1) or Equation (4).

Another alternative would be to use powers (not necessarily integers) of the pixel intensity and/or its neighbors. One of the generalization of Equation (4) can be:

$$G_k = \left[g_k \sqrt{1 - g_k^2} s(g_k) g_k^2 \sqrt{1 - g_k^4} s(g_k) g_k^3 \sqrt{1 - g_k^6} \right] \quad (5)$$

where, k is the pixel index and $s()$ is the sign function. Similar argument can be used to generalize Equation (1).

It should be pointed out that in all the schemes described above, the resultant Eigen vectors should be normalized back using the same formulation used, e.g., Equation (1) or Equation (4).

The performance is assessed through the traditional Root Mean Square Error (RMSE) given by:

$$RMSE = \sqrt{\frac{1}{\|x\|} \sum_{m,n} (x_{mm} - y_{mm})^2} \quad (6)$$

where, x and y stand for original and segmented images and $\|x\|$ is the cardinality of the set. Adjustment should be placed when the range of images are different. In addition, it is unfair to compare performance between images having different number of segments.

Another evaluation scheme is the SSIM given by (Wang *et al.*, 2004):

$$SSIM = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \frac{2\sigma_{xy} + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \quad (7)$$

where, μ is the mean, σ^2 is the variance, $C_1 = 0.0001$, $C_2 = 0.0009$ and σ_{xy} is the covariance between x and y .

Experimental Results

Figure 1 shows the test images used. Fig. 2-7 illustrate the results obtained using neighboring intensity values in the same fashion as in Equation (1). On the other hand, Fig. 8-13 illustrate the results using neighbors in the same fashion as in Equation (4). Table 1 and 2 list the values of RMSE and SSIM respectively for images in Fig. 2-13.

It is obvious that more neighbors may result in better performance with increased computational cost. However, some saturation is inherent which means that we may not get better performance when the number of neighbors goes beyond a certain value, its image dependent though. This result is in favor of Equation (4) compared to Equation (1). Normalization to $[-1, 1]$ can be slightly better than $[0, 1]$. However, more tests are needed as it seems more image dependent.

As shown in Table 1 and 2, the measures are encouraging for the proposed schemes. However, care should be taken when using RMSE and SSIM as they have deficiency regarding scale. All the segmented images will have inferior values in these measures if the images were scaled back to $[0,1]$ or $[-1, 1]$ instead of the computed results, i.e., the means of each region. In addition, the higher the number of regions the higher the value of SSIM (lower RMSE) that may not result in higher performance.

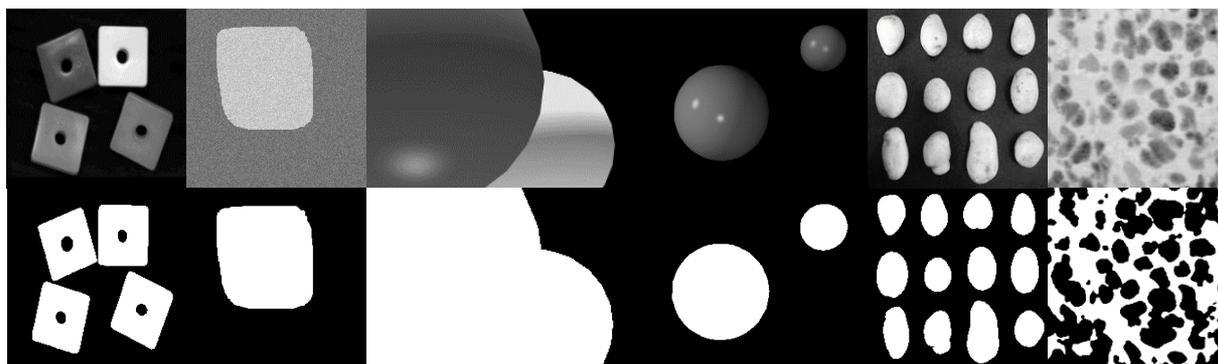


Fig. 1: Test images used and their ground truth



Fig. 2: Resultant images using Equation (1), $[0,1]$ normalization and $(i-1,j)$ neighbor with # of regions (left to right): 3, 3, 2, 3, 3 and 3

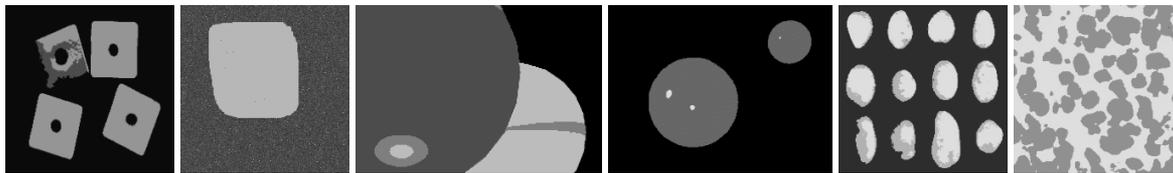


Fig. 3: Resultant images using Equation (1), [0,1] normalization and 4-neighbors with # of regions (left to right): 3, 5, 4, 3, 3 and 3

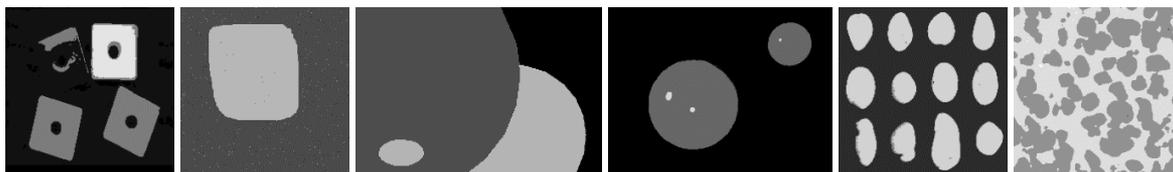


Fig. 4: Resultant images using Equation (1), [0,1] normalization and 8-neighbors with # of regions (left to right): 4, 6, 4, 5, 4 and 7



Fig. 5: Resultant images using Equation (1), [-1,1] normalization and $(i-1,j)$ neighbor with # of regions (left to right): 4, 4, 3, 3, 4 and 4

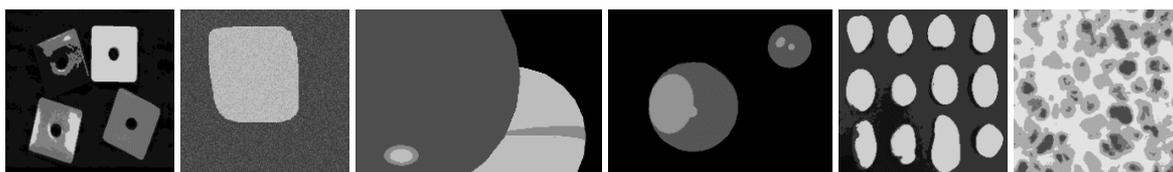


Fig. 6: Resultant images using Equation (1), [-1,1] normalization and 4-neighbors with # of regions (left to right): 6, 6, 6, 6, 6 and 6

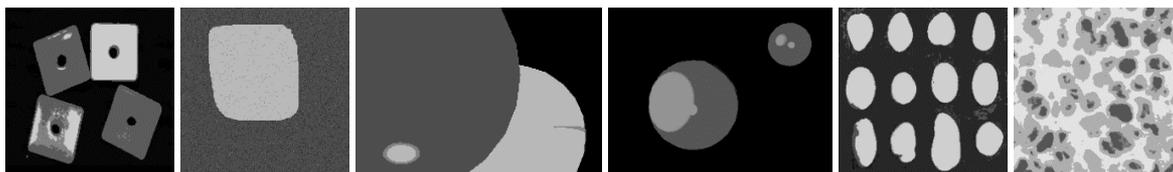


Fig. 7: Resultant images using Equation (1), [-1,1] normalization and 8-neighbors with # of regions(left to right): 8, 8, 7, 8, 8 and 9



Fig. 8: Resultant images using Equation (4), [0,1] normalization and $(i-1,j)$ neighbor with # of regions(left to right): 3, 3, 3, 3, 3 and 3



Fig. 9: Resultant images using Equation (4), [0,1] normalization and 4-neighbors with # of regions (left to right): 5, 5, 6, 6, 5 and 6

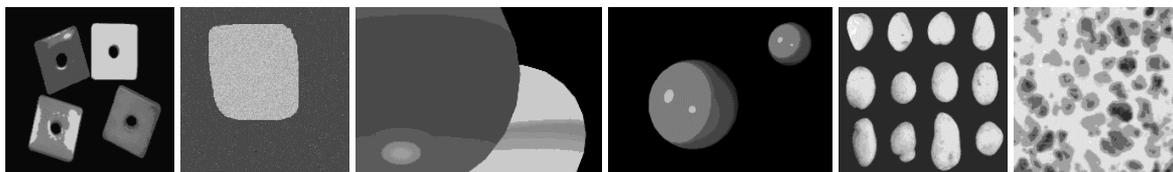


Fig. 10: Resultant images using Equation (4), [0,1] normalization and 8-neighbors with # of regions (left to right): 9, 8, 9, 10, 10 and 11



Fig. 11: Resultant images using Equation (4), [-1,1] normalization and $(i-1,j)$ neighbor with # of regions (left to right): 5, 3, 5, 5, 5 and 5

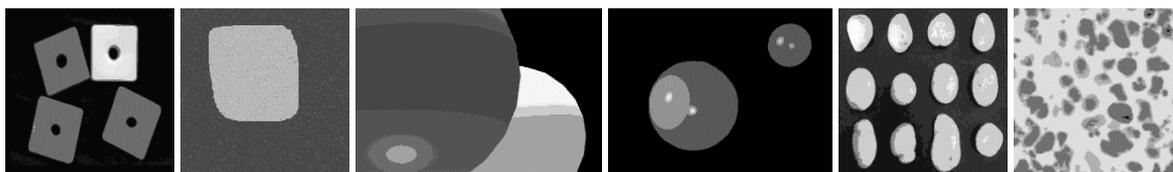


Fig. 12: Resultant images using Equation (4), [-1,1] normalization and 4-neighbors with # of regions (left to right): 11, 8, 11, 9, 10 and 11

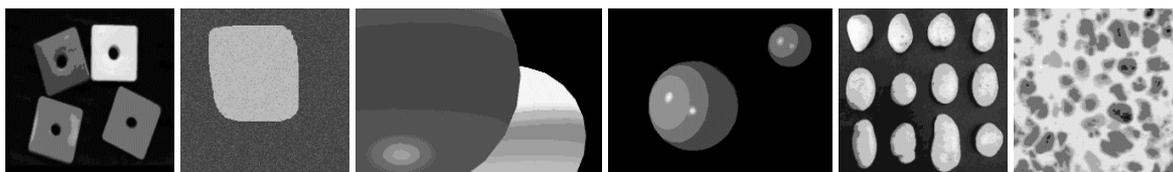


Fig. 13: Resultant images using Equation (4), [-1,1] normalization and 8-neighbors with # of regions (left to right): 19, 14, 18, 16, 19 and 17

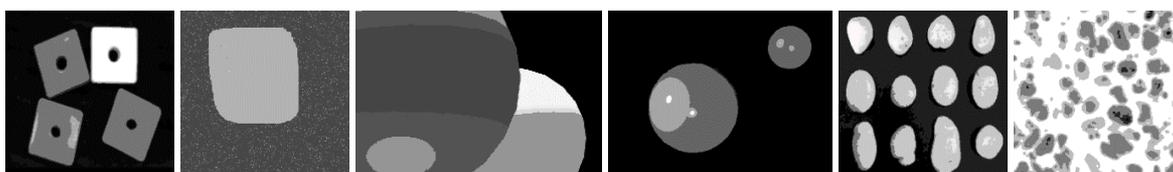


Fig. 14: Iterative implementation of the scheme in Fig. 13 with # of regions (left to right): 9, 7, 10, 7, 8 and 5

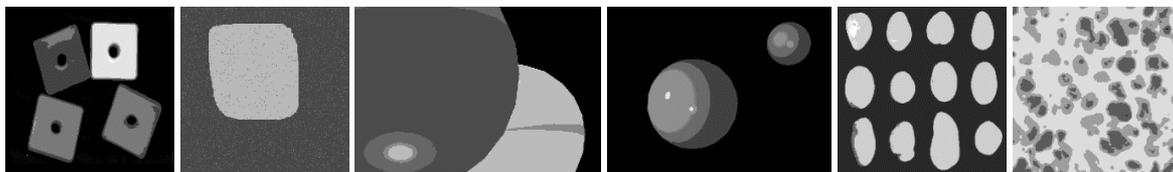


Fig. 15: Resultant images using Equation (5) and [-1,1] normalization with # of regions (left to right): 7, 6, 7, 7, 6 and 7

The scheme used in Fig. 13 was iteratively implemented on each image until no change in the number of resultant regions. Each image is normalized after each iteration. The output images are shown in Fig. 14.

Figure 15 shows the result of using Equation (5) to segment the images in Fig. 1.

To further illustrate the effectiveness of the proposed schemes, some images from Berkley Segmentation Database (BSD) (Martin *et al.*, 2001) are segmented in Fig. 16 using the schemes used in Fig. 13 and 15 respectively. The resultant images are normalized after the segmentation process.

Table 1: RMSE for the results of schemes implemented in Fig. 2-13

Image/Method	1	2	3	4	5	6
Figure 2	0.127	0.074	0.152	0.064	0.085	0.105
Figure 3	0.112	0.054	0.064	0.061	0.075	0.107
Figure 4	0.083	0.070	0.074	0.060	0.079	0.108
Figure 5	0.118	0.111	0.124	0.044	0.098	0.103
Figure 6	0.077	0.036	0.065	0.040	0.069	0.068
Figure 7	0.073	0.056	0.069	0.039	0.075	0.072
Figure 8	0.091	0.066	0.125	0.054	0.075	0.101
Figure 9	0.086	0.044	0.118	0.034	0.072	0.053
Figure 10	0.053	0.056	0.047	0.027	0.052	0.044
Figure 11	0.056	0.094	0.114	0.043	0.076	0.093
Figure 12	0.052	0.057	0.027	0.041	0.043	0.060
Figure 13	0.029	0.042	0.016	0.022	0.033	0.046

Table 2: SSIM for the results of schemes implemented in Fig. 2-13.

Image/Method	1	2	3	4	5	6
Figure 2	0.875	0.929	0.743	0.930	0.961	0.812
Figure 3	0.905	0.963	0.962	0.936	0.969	0.803
Figure 4	0.950	0.937	0.949	0.938	0.967	0.799
Figure 5	0.893	0.826	0.842	0.967	0.947	0.818
Figure 6	0.957	0.983	0.961	0.973	0.975	0.928
Figure 7	0.961	0.960	0.955	0.975	0.970	0.917
Figure 8	0.939	0.944	0.841	0.951	0.970	0.828
Figure 9	0.946	0.976	0.860	0.981	0.972	0.958
Figure 10	0.979	0.960	0.979	0.987	0.985	0.971
Figure 11	0.977	0.881	0.870	0.969	0.969	0.856
Figure 12	0.980	0.958	0.993	0.971	0.990	0.945
Figure 13	0.994	0.977	0.997	0.992	0.994	0.968

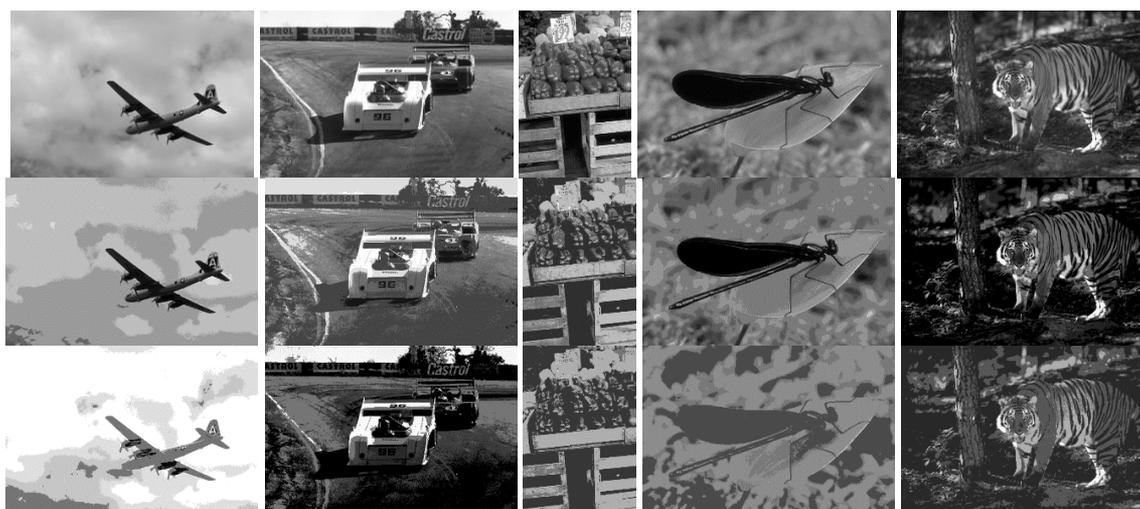


Fig. 16: (Top to bottom) Original images from BSD followed by segmentations using schemes of Fig. 13 and 15 with their respective number of regions: 3096(18, 7), 21077(14, 7), 25098(15, 6), 35070(17, 7) and 108005(18, 7)

It is easily noticed from these images that using the normalization $[-1,1]$ produces higher number of regions compared to $[0,1]$. However, the latter has better performance for thresholding tasks while the former is advantageous for segmentation tasks.

The schemes are easily extended to any feature space using Equation (1), (4) and (5) or any combination of them.

Conclusion and Future Work

Simple algorithms have been proposed in this study to perform multi-level image segmentation. Thresholding is a special case where only V_{max} is used. The proposed schemes are very effective as demonstrated by the values of RMSE and SSIM. However, better evaluation schemes are needed to better distinguish the performances.

More elaboration is needed on the best aggregation used in selecting the thresholds. In particular, how many neighbors are required? Is it beneficial to append the mean or the median? What powers are essential? or even some sort of mixture.

The proposed algorithms can be easily extended to any feature space, e.g., color. However, the optimum selection and/or weighting are still need to be determined. The cost is paid by the extra computation required.

The component added to obtain a unit vector, Equation (1), can be generalized to any fuzzy complement. However, more work is needed to find the best formula and whether improvements can be attained, (Ameer, 2020) for some suggestions.

Ethics

This article is original and contains unpublished material. The corresponding author confirms that all of the other authors have read and approved the manuscript and no ethical issues involved.

References

- Ameer, S., (2020). Image thresholding using the complement feature. *American Journal of Engineering and Applied Sciences*, 13, 311-317.
- Ilea, D. E. and Whelan, P. F. (2011). Image segmentation based on the integration of colour-texture descriptors-A review. *Pattern Recognition*, 44(10-11), 2479-2501.
- Martin, D., Fowlkes, C., Tal, D. and Malik, J. (2001, July). A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001 (Vol. 2, pp. 416-423)*. IEEE.
- Peng, B., Zhang, L. and Zhang, D. (2013). A survey of graph theoretical approaches to image segmentation. *Pattern recognition*, 46(3), 1020-1038.
- Sezgin, M. and Sankur, B. (2004). Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic imaging*, 13(1), 146-166.
- Unnikrishnan, R. and Hebert, M. (2005). Measures of similarity. In *Proc. of the IEEE Workshop on Applications of Computer Vision and the IEEE Workshop on Motion and Video Computing (Vol. 1, pp. 394-400)*.
- Wang, Z. A., C Bovik, H., Sheikh, R. and Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4), 600-612.