An Iterative Raster Scan Algorithm for Superpixel Segmentation

Inoue, Kohei
Faculty of Design, Kyushu University

Hara, Kenji
Faculty of Design, Kyushu University

Ono, Naoki
Faculty of Design, Kyushu University

Hiraoka, Toru
Faculty of Information Systems, University of Nagasaki

Journal of the Institute of Industrial Applications Engineers Vol.12, No.1, pp.12–17, (2024.1.25)
DOI: 10.12792/JIIAE.12.12 Online edition: ISSN 2187-8811 Print edition: ISSN 2188-1758

Paper

An Iterative Raster Scan Algorithm for Superpixel Segmentation

Kohei Inoue*† Member, Kenji Hara† Member
Naoki Ono† Member, Toru Hiraoka‡ Member

(Received October 1, 2015, revised December 1, 2015)

Abstract: We propose an iterative raster scan algorithm for superpixel segmentation, which is based on the K-means clustering algorithm. The proposed algorithm updates the class label of each pixel only at the boundaries of superpixels in a raster scan order and refers to only two neighboring pixels per pixel for updating the variables. Therefore, the proposed algorithm is computationally efficient compared with existing methods. Experimental results show that the proposed algorithm generates compact and adherent superpixels in a finite number of iterations of the raster scan process.

Keywords: Superpixel segmentation, raster scan, K-means clustering algorithm

1. Introduction

Image segmentation is a fundamental technique in the field of image processing [1]. Superpixel segmentation [2] is a kind of image segmentation, and is also called the image over segmentation [3]. Recently, Achanta et al. have proposed a superpixel algorithm named the *simple linear iterative clustering* (SLIC), which adapts a K-means clustering approach [4]. SLIC has the potential to be used for a large number of applications, e.g., Crommelinck et al. applied it to a remote sensing task [5]. Sveral surveys of superpixel segmentation have also been published, e.g., Stutz et al. presented a comprehensive evaluation of 28 state-of-the-art superpixel algorithms [6], where SLIC placed among the recommended 6 algorithms for use in practice. Currently, a large number of segmentation techniques have been widely used in a variety of fields including industry [7], biomedicine [8], remote sensing [9] and IoV (Internet of Vehicles) [10], that demonstrate researchers' great interest in image segmentation and superpixel segmentation.

In this paper, we propose an iterative raster scan algorithm for superpixel segmentation, and show experimental results which demonstrate that the proposed algorithm generates compact and adherent superpixels in a finite number of iterations of a raster scan process.

The rest of this paper is organized as follows: Section 2 proposes an iterative raster scan algorithm for superpixel segmentation. Section 3 shows experimental results on publicly available image datasets. Finally, Section 4 concludes this paper.

2. Iterative Raster Scan Algorithm

Let $p(i, j)$ be a color vector of a pixel at the position (i, j) in a color image the size of which is $m \times n$ pixels, where m and n denote the numbers of rows and columns, respectively. Then we define a feature vector of a pixel (i, j) as $f(i, j) = [p(i, j), \sqrt{\alpha}i, \sqrt{\alpha}j]$ where α is a positive value which controls the importance of spatial coordinates (i, j) in the feature vector. With these feature vectors, we can calculate the weighted sum of $\|p(i, j) - p(i, j+1)\|^2$ and $\|(i, j) - (i, j+1)\|^2$ by the square of Euclidean distance between $f(i, j)$ and $f(i, j+1)$ as $\|f(i, j) - f(i, j+1)\|^2 = \|p(i, j) - p(i, j+1)\|^2 + \alpha\|(i, j) - (i, j+1)\|^2$.

Let K be the number of superpixels, and let $u(i, j) \in \{1, 2, \ldots, K\}$ be the identity number of superpixel to which a pixel (i, j) belongs. Then we formulate the problem of superpixel segmentation as follows:

$$\min_{u(i,j),\ \bar{f}_k} \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{K} \delta_{u(i,j),k} \left\| f(i, j) - \bar{f}_k \right\|^2, \tag{1}$$

where $\delta_{u(i,j),k}$ denotes the Kronecker delta such that $\delta_{u(i,j),k} = 1$ if $u(i, j) = k$, and $\delta_{u(i,j),k} = 0$ otherwise, and $\bar{f}_k$ is the feature vector for the kth superpixel among K superpixels. If we initialize $u(i, j)$ in an appropriate way such as a regular subdivision of an image plane, then, from the necessary condition for optimality of the above problem (1):

$$\frac{\partial E}{\partial \bar{f}_k} = -2\sum_{i=1}^{m}\sum_{j=1}^{n} \delta_{u(i,j),k} \left[f(i, j) - \bar{f}_k \right] = 0, \tag{2}$$

where E denotes the objective function in (1), we have

$$\bar{f}_k = \frac{\displaystyle\sum_{i=1}^{m}\sum_{j=1}^{n} \delta_{u(i,j),k} f(i, j)}{\displaystyle\sum_{i=1}^{m}\sum_{j=1}^{n} \delta_{u(i,j),k}}. \tag{3}$$

After that, we update $u(i, j)$ for decreasing E as follows:

$$u(i, j) = \arg\min_{k \in \{1,2,\ldots,K\}} \left\| f(i, j) - \bar{f}_k \right\|^2. \tag{4}$$

* Corresponding: k-inoue@design.kyushu-u.ac.jp
† Faculty of Design, Kyushu University
‡ Faculty of Information Systems, University of Nagasaki

The above procedure for updating $\bar{f}_k$ and $u(i,j)$ by (3) and (4), respectively, is iterated until the convergence. However, this naive approach is computationally expensive because every iteration of (3) and (4) includes full search of all pixels in an image. To avoid such an inconvenience, for example, the search region is restricted within a small square in SLIC.

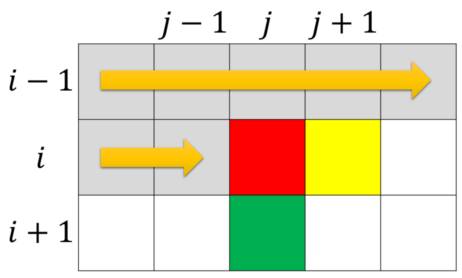


Figure 1: The current pixel (i,j) (red) and its neighboring pixels $(i,j+1)$ (yellow) and $(i+1,j)$ (green) in the raster scanning process.

This section proposes a raster scan algorithm that refers to only two neighboring pixels per pixel for updating $\bar{f}_k$ and $u(i,j)$ at each iteration. Figure 1 illustrates the proposed raster scan process, where the current pixel to be updated is denoted in red, and the neighboring pixels to be referred to are denoted in yellow and green.

The detailed procedure of the proposed algorithm is as follows: Assume that the pixel (i,j) is the current pixel as shown in Figure 1 with a red square. If the red and yellow pixels belong to different superpixels to each other, i.e., $u(i,j) \neq u(i,j+1)$, then we compute the potential effect of change in $u(i,j)$ and $u(i,j+1)$ as follows:

$$d_1 = \left\| f(i,j) - \bar{f}_{u(i,j)} \right\|^2 - \left\| f(i,j) - \bar{f}_{u(i,j+1)} \right\|^2, \tag{5}$$

$$d_2 = \left\| f(i,j+1) - \bar{f}_{u(i,j+1)} \right\|^2 - \left\| f(i,j+1) - \bar{f}_{u(i,j)} \right\|^2. \tag{6}$$

Similarly, if the red and green pixels belong to different superpixels, i.e., $u(i,j) \neq u(i+1,j)$, then we compute the potential effect of change in $u(i,j)$ and $u(i+1,j)$ as follows:

$$d_3 = \left\| f(i,j) - \bar{f}_{u(i,j)} \right\|^2 - \left\| f(i,j) - \bar{f}_{u(i+1,j)} \right\|^2, \tag{7}$$

$$d_4 = \left\| f(i+1,j) - \bar{f}_{u(i+1,j)} \right\|^2 - \left\| f(i+1,j) - \bar{f}_{u(i,j)} \right\|^2. \tag{8}$$

Then we choose the most effective change among the above four cases. Let $l_{\max}$ be the index of the largest value among d_1, d_2, d_3 and d_4, i.e., $l_{\max} = \arg\max_{l\in\{1,2,3,4\}} d_l$. If $l_{\max} = 1$, which indicates that the pixel (i,j) should belong to the $u(i,j+1)$th superpixel, then we update $\bar{f}_{u(i,j)}$ and $\bar{f}_{u(i,j+1)}$ as follows:

$$\bar{f}_{u(i,j)} \leftarrow \frac{\bar{f}_{u(i,j)} N_{u(i,j)} - f(i,j)}{N_{u(i,j)} - 1}, \tag{9}$$

$$\bar{f}_{u(i,j+1)} \leftarrow \frac{\bar{f}_{u(i,j+1)} N_{u(i,j+1)} + f(i,j)}{N_{u(i,j+1)} + 1}, \tag{10}$$

where $N_{u(i,j)}$ denotes the number of pixels belonging to the $u(i,j)$th superpixel as $N_{u(i,j)} = \sum_{i'=1}^{m} \sum_{j'=1}^{n} \delta_{u(i,j),u(i',j')}$. Also, we update $u(i,j)$ as follows:

$$u(i,j) \leftarrow u(i,j+1), \tag{11}$$

that is, we transfer the pixel (i,j) from the $u(i,j)$th superpixel to the $u(i,j+1)$ th one. Accordingly, $N_{u(i,j)}$ and $N_{u(i,j+1)}$ are also updated as follows:

$$N_{u(i,j)} \leftarrow N_{u(i,j)} - 1, \tag{12}$$

$$N_{u(i,j+1)} \leftarrow N_{u(i,j+1)} + 1. \tag{13}$$

On the other hand, if $l_{\max} = 2$, which indicates that the pixel $(i,j+1)$ should belong to the $u(i,j)$th superpixel, then we update $\bar{f}_{u(i,j+1)}$ and $\bar{f}_{u(i,j)}$ as follows:

$$\bar{f}_{u(i,j+1)} \leftarrow \frac{\bar{f}_{u(i,j+1)} N_{u(i,j+1)} - f(i,j+1)}{N_{u(i,j+1)} - 1}, \tag{14}$$

$$\bar{f}_{u(i,j)} \leftarrow \frac{\bar{f}_{u(i,j)} N_{u(i,j)} + f(i,j+1)}{N_{u(i,j)} + 1}. \tag{15}$$

We also update $u(i,j+1)$ as follows:

$$u(i,j+1) \leftarrow u(i,j), \tag{16}$$

that is, we transfer the pixel $(i,j+1)$ from the $u(i,j+1)$th superpixel to the $u(i,j)$ th one. Accordingly, $N_{u(i,j+1)}$ and $N_{u(i,j)}$ are also updated as follows:

$$N_{u(i,j+1)} \leftarrow N_{u(i,j+1)} - 1, \tag{17}$$

$$N_{u(i,j)} \leftarrow N_{u(i,j)} + 1. \tag{18}$$

In a similar manner, if $l_{\max} = 3$, which indicates that the pixel (i,j) should belong to the $u(i+1,j)$th superpixel, then we update the relevant variables as follows:

$$\bar{f}_{u(i,j)} \leftarrow \frac{\bar{f}_{u(i,j)} N_{u(i,j)} - f(i,j)}{N_{u(i,j)} - 1}, \tag{19}$$

$$\bar{f}_{u(i+1,j)} \leftarrow \frac{\bar{f}_{u(i+1,j)} N_{u(i+1,j)} + f(i,j)}{N_{u(i+1,j)} + 1}, \tag{20}$$

$$u(i,j) \leftarrow u(i+1,j), \tag{21}$$

$$N_{u(i,j)} \leftarrow N_{u(i,j)} - 1, \tag{22}$$

$$N_{u(i+1,j)} \leftarrow N_{u(i+1,j)} + 1, \tag{23}$$

and if $l_{\max} = 4$, which indicates that the pixel $(i+1,j)$ should belong to the $u(i,j)$th superpixel, then we update the relevant variables as follows:

$$\bar{f}_{u(i+1,j)} \leftarrow \frac{\bar{f}_{u(i+1,j)} N_{u(i+1,j)} - f(i+1,j)}{N_{u(i+1,j)} - 1}, \tag{24}$$

$$\bar{f}_{u(i,j)} \leftarrow \frac{\bar{f}_{u(i,j)} N_{u(i,j)} + f(i+1,j)}{N_{u(i,j)} + 1}, \tag{25}$$

$$u(i+1,j) \leftarrow u(i,j), \tag{26}$$

$$N_{u(i+1,j)} \leftarrow N_{u(i+1,j)} - 1, \tag{27}$$

$$N_{u(i,j)} \leftarrow N_{u(i,j)} + 1. \tag{28}$$

Algorithm 1 Iterative raster scan algorithm

```
Require: a color image with pixels $p(i, j)$ for $i = 1, 2, \ldots, m$ and
    $j = 1, 2, \ldots, n$, the number of superpixels $K$
Ensure: labels $u(i, j) \in \{1, 2, \ldots, K\}$, mean feature vectors $\bar{f}_k$ for
    $k = 1, 2, \ldots, K$
 1: Initialize $u(i, j)$ for $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$;
 2: while True do
 3:     $u^{\text{old}} := u.\text{copy}()$;
 4:     for $i = 1, 2, \ldots, m$ do
 5:         for $j = 1, 2, \ldots, n$ do
 6:             $d_1 := d_2 := d_3 := d_4 := 0$;
 7:             if $j + 1 \le n$ and $u(i, j) \neq u(i, j + 1)$ then
 8:                 $d_1 := \|f(i, j) - \bar{f}_{u(i,j)}\|^2 - \|f(i, j) - \bar{f}_{u(i,j+1)}\|^2$;
 9:                 $d_2 := \|f(i, j + 1) - \bar{f}_{u(i,j+1)}\|^2 - \|f(i, j + 1) - \bar{f}_{u(i,j)}\|^2$;
10:             end if
11:             if $i + 1 \le m$ and $u(i, j) \neq u(i + 1, j)$ then
12:                 $d_3 := \|f(i, j) - \bar{f}_{u(i,j)}\|^2 - \|f(i, j) - \bar{f}_{u(i+1,j)}\|^2$;
13:                 $d_4 := \|f(i + 1, j) - \bar{f}_{u(i+1,j)}\|^2 - \|f(i + 1, j) - \bar{f}_{u(i,j)}\|^2$;
14:             end if
15:             $l_{\max} := \arg\max_{l \in \{1,2,3,4\}} d_l$;
16:             if $l_{\max} == 1$ then
17:                 $\bar{f}_{u(i,j)} \leftarrow \frac{\bar{f}_{u(i,j)} N_{u(i,j)} - f(i,j)}{N_{u(i,j)} - 1}$;
18:                 $\bar{f}_{u(i,j+1)} \leftarrow \frac{\bar{f}_{u(i,j+1)} N_{u(i,j+1)} + f(i,j)}{N_{u(i,j+1)} + 1}$;
19:                 $u(i, j) \leftarrow u(i, j + 1)$;
20:                 $N_{u(i,j)} \leftarrow N_{u(i,j)} - 1$;
21:                 $N_{u(i,j+1)} \leftarrow N_{u(i,j+1)} + 1$;
22:             else if $l_{\max} == 2$ then
23:                 $\bar{f}_{u(i,j+1)} \leftarrow \frac{\bar{f}_{u(i,j+1)} N_{u(i,j+1)} - f(i,j+1)}{N_{u(i,j+1)} - 1}$;
24:                 $\bar{f}_{u(i,j)} \leftarrow \frac{\bar{f}_{u(i,j)} N_{u(i,j)} + f(i,j+1)}{N_{u(i,j)} + 1}$;
25:                 $u(i, j + 1) \leftarrow u(i, j)$;
26:                 $N_{u(i,j+1)} \leftarrow N_{u(i,j+1)} - 1$;
27:                 $N_{u(i,j)} \leftarrow N_{u(i,j)} + 1$;
28:             else if $l_{\max} == 3$ then
29:                 $\bar{f}_{u(i,j)} \leftarrow \frac{\bar{f}_{u(i,j)} N_{u(i,j)} - f(i,j)}{N_{u(i,j)} - 1}$;
30:                 $\bar{f}_{u(i+1,j)} \leftarrow \frac{\bar{f}_{u(i+1,j)} N_{u(i+1,j)} + f(i,j)}{N_{u(i+1,j)} + 1}$;
31:                 $u(i, j) \leftarrow u(i + 1, j)$;
32:                 $N_{u(i,j)} \leftarrow N_{u(i,j)} - 1$;
33:                 $N_{u(i+1,j)} \leftarrow N_{u(i+1,j)} + 1$;
34:             else
35:                 $\bar{f}_{u(i+1,j)} \leftarrow \frac{\bar{f}_{u(i+1,j)} N_{u(i+1,j)} - f(i+1,j)}{N_{u(i+1,j)} - 1}$;
36:                 $\bar{f}_{u(i,j)} \leftarrow \frac{\bar{f}_{u(i,j)} N_{u(i,j)} + f(i+1,j)}{N_{u(i,j)} + 1}$;
37:                 $u(i + 1, j) \leftarrow u(i, j)$;
38:                 $N_{u(i+1,j)} \leftarrow N_{u(i+1,j)} - 1$;
39:                 $N_{u(i,j)} \leftarrow N_{u(i,j)} + 1$;
40:             end if
41:         end for
42:     end for
43:     if $u == u^{\text{old}}$ then
44:         Break;
45:     end if
46: end while
```

The above procedure is applied to all pixels repeatedly in the iterative raster scan process until a convergence condition is satisfied. After the convergence, each superpixel is given as a set of pixels having the same value of $u(i, j)$, and the mean color vector of the kth superpixel is included in the corresponding feature vector $\bar{f}_k = [\bar{p}_k, \sqrt{\alpha} i_k, \sqrt{\alpha} j_k]$ as $\bar{p}_k$, where (i_k, j_k) denotes the position of the centroid of the superpixel. The pseudocode of the proposed procedure is shown in Algorithm 1.

In Algorithm 1, line 3, the statement "$u^{\text{old}} := u.\text{copy}()$" means to copy the array $u = [u(i, j)]$ to another array u^{old}, which is used at the end of the code to break out of the main while loop.

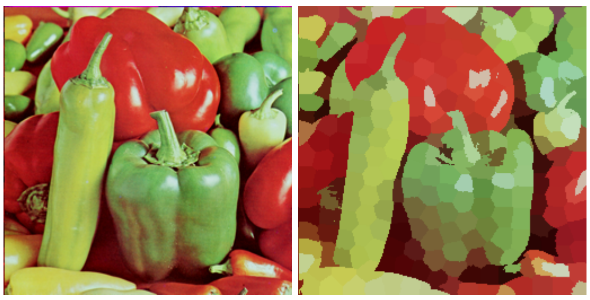

(a) Original image (b) Segmented image

Figure 2: Superpixel segmentation result with $\alpha = 0.001$ and $K = 256$.

3. Experimental Results

This section shows experimental results of superpixel segmentation on the standard image database SIDBA [11]. Figure 2 shows a result where an original color image in Figure 2(a) with 256×256 pixels is segmented by the proposed method with $\alpha = 0.001$ and $K = 256$ as shown in Figure 2(b) where each segment is painted in its mean color $\bar{p}_k$. The boundaries of superpixels adhere well to object boundaries in the image, and the generated superpixels are compact, especially in flat regions.

Figure 3 shows the change in the objective function value in (1), where the vertical and horizontal axes denote the objective function value and the number of iterations, respectively. The objective function value monotonically decreases as the iteration procedure progresses. In this example, the variable $u(i, j)$ exactly converged in 49 iterations to obtain the result in Figure 2(b).

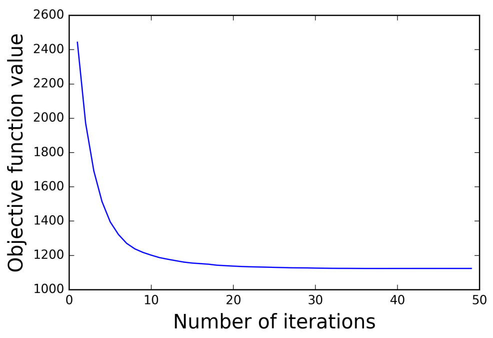


Figure 3: Objective function value.

Figure 4 shows some intermediate images obtained during the iteration procedure. First, we initialized the array of identity numbers, $u(i, j)$, with the square lattice as shown in

Figure 4(a). The obtained images after the 5th to 25th iterations (every 5 iterations) are shown in Figures 4(b) to (f), respectively. The image after the 20th iteration in Figure 4(e) is visually similar to the final result in Figure 2(b).

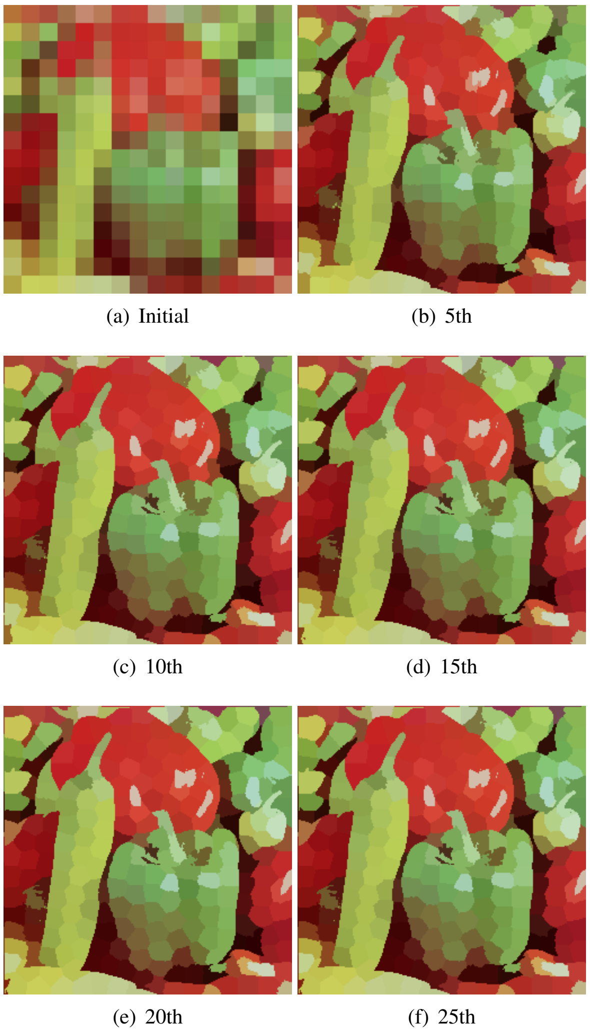

(a) Initial (b) 5th (c) 10th (d) 15th (e) 20th (f) 25th

Figure 4: Intermediate images at 0th (initial) to 25th iterations.

Figure 5 shows the results with different number of superpixels, where Figure 5(a) is obtained with $K = 16$ after 34 iterations, Figure 5(b) is obtained with $K = 64$ after 56 iterations, Figure 5(c) is obtained with $K = 256$ after 49 iterations, and Figure 5(d) is obtained with $K = 1024$ after 29 iterations. These results suggest that the larger the number of superpixels K is, the smaller the number of iterations the procedure needs, and the larger the number of superpixels K is, the more similar to the original image the resultant image becomes.

Figure 6 shows additional results on the Berkeley Segmentation Data Set and Benchmarks 500 (BSDS500) [12], where Figure 6(a) shows five original images, and Figure 6(b) shows the corresponding segmentation results. The size of the top two landscape images is 321 × 481 pixels, and that of the bottom three portrait images is 481 × 321 pixels. The number of superpixels is $K = 384$ for each image, and we set $\alpha = 0.001$.

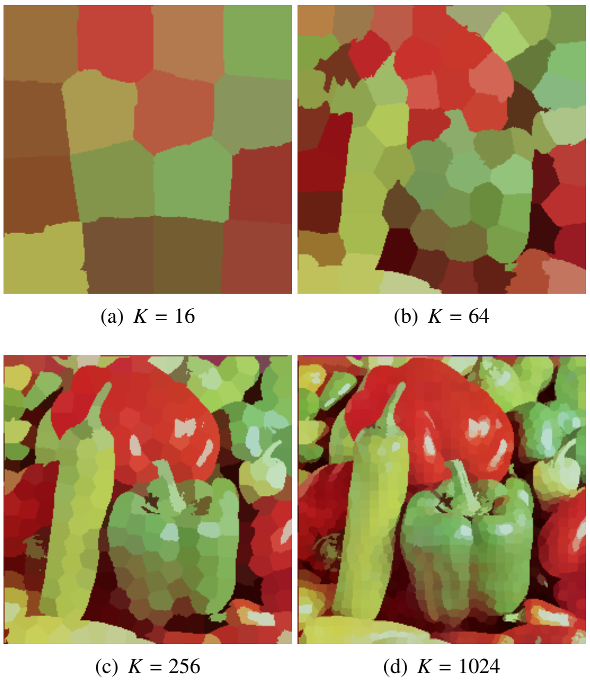

(a) $K = 16$ (b) $K = 64$ (c) $K = 256$ (d) $K = 1024$

Figure 5: Results with the different number of superpixels.

We evaluate the computation cost of the proposed method with the number of distance calculations, and compare it with that of SLIC [4]. The proposed method calculates two distances per pixel as shown in Figure 1, where, for the red pixel, we calculate two distances between red-yellow and red-green pixels. The number of pixels in an image is given by mn in this paper. On the basis of the observation in Figure 3, we choose 20 as a practical number of iterations of the raster scan process. Then we can evaluate the total number of distance calculations as $2 \times mn \times 20 = 40mn$. On the other hand, SLIC searches a limited region, $2S \times 2S$, where $S = \sqrt{mn/K}$, and gives an acceptable result with 10 iterations. Therefore, we can evaluate the number of distance calculations in SLIC as $2S \times 2S \times mn \times 10 = 40mnS^2$. Figure 7 shows the comparison result, where the vertical and horizontal axes denote the numbers of distance calculations and image pixels, respectively. The blue, orange and green dotted lines denote SLIC with $K = 100$, 200 and 300. In SLIC, the search area $2S \times 2S$ becomes smaller when K increases due to the relationship $S = \sqrt{mn/K}$. As a result, the number of distance calculations decreases as K increases in SLIC. On the other hand, the red solid line denotes the proposed method, which requires less number of distance calculations than SLIC.

The above results demonstrate that the proposed algorithm can generate compact superpixels while the boundaries of superpixels adhere well to the object boundaries. The number of distance calculations of the proposed algorithm is smaller than that of SLIC. For further improvement, a hexagonal lattice instead of a square lattice may be adopted in the initialization step of the proposed method.

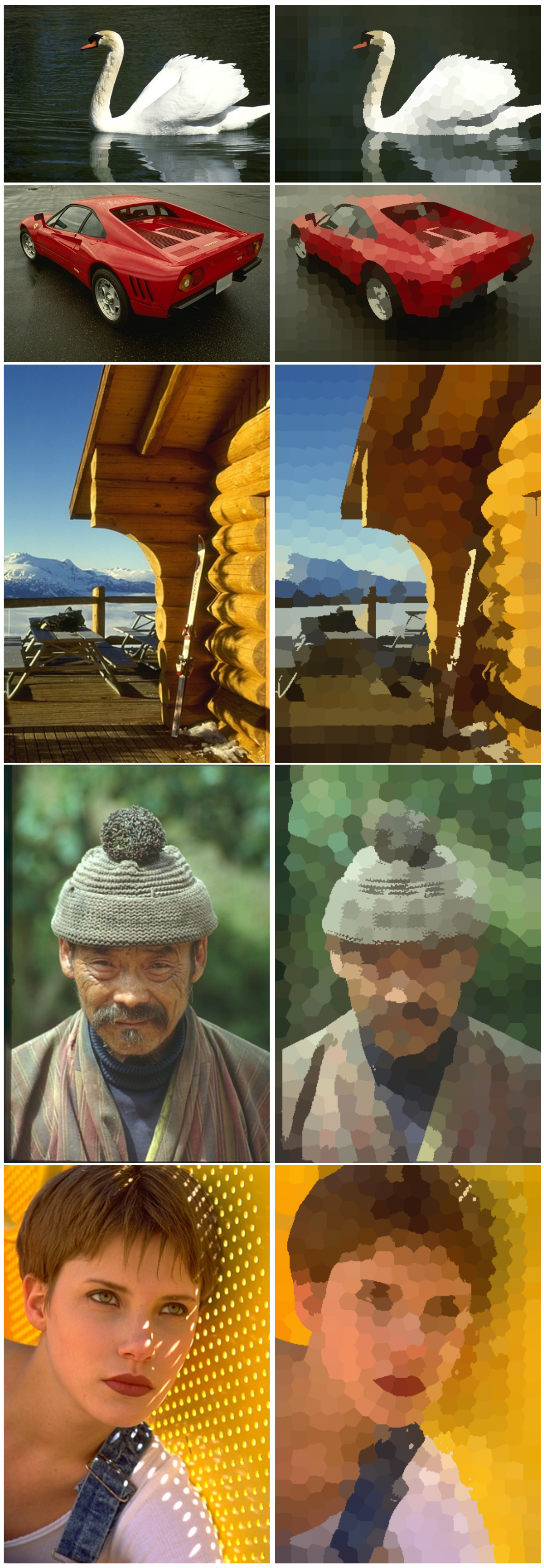

(a) Original (b) Segmented

Figure 6: Results on BSDS500 dataset [12].

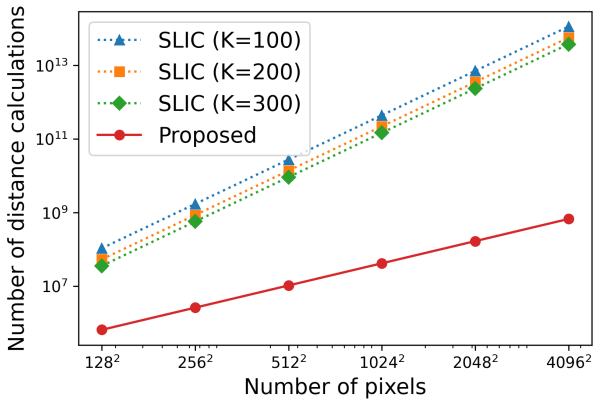


Figure 7: Distance calculation comparison.

4. Conclusions

In this paper, we proposed an iterative raster scan algorithm for superpixel segmentation. Experimental results show that the proposed algorithm generates compact and adherent superpixels from given color images. We demonstrated that the number of distance calculations of the proposed algorithm is smaller than SLIC [4] when the numbers of iterations are set practically. The advantage of the proposed algorithm is that the proposed algorithm has a small number of parameters, and is computationally efficient compared with related methods including SLIC.

In our future work, we would like to apply the proposed superpixel segmentation algorithm to non-photorealistic rendering for image abstraction, and extend the proposed iterative raster scan algorithm to higher dimensions such as supervoxels [13] and hyperpixels [14].

Acknowledgment

This research was supported by JSPS KAKENHI Grant Numbers JP21K11964, JP23K11149 and JP23K11727.

References

[1] T. Lei, A. K. Nandi, "Image Segmentation: Principles, Techniques, and Applications", *Willey*, NJ, 2022.

[2] X. Ren and J. Malik, "Learning a classification model for segmentation", *Proc. IEEE ICCV*, vol. 2, 2003. `https://doi.org/10.1109/ICCV.2003.1238308`

[3] C. L. Zitnick, S. B. Kang, "Stereo for Image-Based Rendering Using Image Over-Segmentation", *International Journal of Computer Vision*, vol. 75, no. 1, pp. 49-65, 2007. `https://doi.org/10.1007/s11263-006-0018-8`

[4] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC Superpixels Compared to State-of-the-Art Superpixel Methods", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2274-2282, 2012. `https://doi.org/10.1109/TPAMI.2012.120`

[5] S. C. Crommelinck, R. M. Bennett, M. Gerke, M. N. Koeva, M. Y. Yang, and G. Vosselman, "SLIC superpixels for object delineation UAV data", *Proc. Interna-*

tional Conference on Unmanned Aerial Vehicles in Geomatics, vol. IV-2/W3, 2017. `https://doi.org/10.5194/isprs-annals-IV-2-W3-9-2017`

[6] D. Stutz, Alexander Hermans, and B. Leibe, " Superpixels: An Evaluation of the State-of-the-Art", *Computer Vision and Image Understanding*, vol. 166, pp. 1-27, 2018. `https://doi.org/10.1016/j.cviu.2017.03.007`

[7] Y. Gao, J. Lin, J. Xie, Z. Ning, "A Real-Time Defect Detection Method for Digital Signal Processing of Industrial Inspection Applications", *IEEE Transactions on Industrial Informatics*, vol. 17, no. 5, pp. 3450-3459, 2021. `https://doi.org/10.1109/TII.2020.3013277`

[8] F. Isensee, P.F. Jaeger, S.A.A. Kohl, et al., "nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation", *Nat Methods*, vol. 18, pp. 203-211, 2021. `https://doi.org/10.1038/s41592-020-01008-z`

[9] S. Hao, W. Wang, M. Salzmann, "Geometry-Aware Deep Recurrent Neural Networks for Hyperspectral Image Classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 59, no. 3, pp. 2448-2460, 2021. `https://doi.org/10.1109/TGRS.2020.3005623`

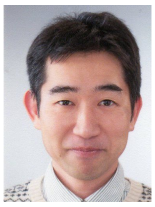
[10] W. Xie, X. Xu, Xin, R. Liu, Y. Jin, W. Bai, Q.Li, "Living in a Simulation? An Empirical Investigation of the Smart Driving Simulation Test System," (December 22, 2020). *Forthcoming at Journal of the Association for Information Systems*, Available at SSRN: `https://ssrn.com/abstract=3753329`

[11] M. Sakauchi, Y. Ohsawa, M. Sone, and M. Onoe, " Management of the Standard Image Database for Image Processing Researches", *ITEJ Technical Report*, vol. 8, no. 38, pp. 7-12, 1984. (in Japanese)

[12] P. Arbelaez, M. Maire, C. Fowlkes, J. Malik, " Contour Detection and Hierarchical Image Segmentation", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898-916, 2011. `https://doi.org/10.1109/TPAMI.2010.161`

[13] C. Xu, J.J. Corso, "LIBSVX: A Supervoxel Library and Benchmark for Early Video Processing", *Int J Comput Vis* vol. 119, pp. 272-290, 2016. `https://doi.org/10.1007/s11263-016-0906-5`

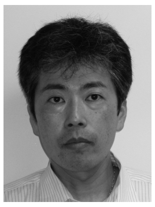
[14] M. Hamad, C. Conti, P. Nunes and L. D. Soares, "Hyperpixels: Flexible 4D Over-Segmentation for Dense and Sparse Light Fields", *IEEE Transactions on Image Processing*, vol. 32, pp. 3790-3805, 2023. `https://doi.org/10.1109/TIP.2023.3290523`



Kohei Inoue (Member) He received B.Des., M.Des. and D.Eng. degrees from Kyushu Institute of Design in 1996, 1998 and 2000, respectively. He is currently an Associate Professor at Kyushu University. His research interests include pattern recognition and image processing.

Kenji Hara (Member) He received the B.E. and M.E. degrees from Kyoto University in 1987 and 1989, respectively, and the Ph.D. degree from Kyushu University in 1999. He is currently a Professor at Kyushu University. His research interests include physics-based vision and geometric modeling.

Naoki Ono (Member) was born at Japan, in 1961. He received a Ph.D. degree in engineering from Kyushu University in 1997, and is presently an associate professor at Kyushu University. He has worked on image processing and pattern recognition. He is a member of IIAE and IEICE.



Toru Hiraoka (Member) He received B.Des., M.Des. and D.Eng. degrees from Kyushu Institute of Design in 1995, 1997 and 2005, respectively. He is currently a Professor at the University of Nagasaki. His research interests include non-photorealistic rendering and disaster prevention.