# Image segmentation by SUP clustering algorithm

Ting-Li Chen*

**Abstract**
Image segmentation plays a very important role in image processing. There are various approaches in this field. One of them is by clustering algorithm. Each pixel in an image can be treated as an individual subject, and the goal of image segmentation is to assign these subjects into clusters with similar pixels grouped in the same cluster. If only intensity information is considered, clustering results are usually poor, especially for noisy data. The location information is another useful variable for segmentation. However, most popular clustering algorithms do not perform well even after the location variable is taken into consideration. We applied SUP clustering algorithm (Chen & Shiu 2007) on the segmentation problem and obtained good results. In this paper, we will present how to choose parameters in SUP algorithm for image segmentation and how to combine the information from intensity and location.

**Key Words:** Image segmentation, clustering algorithm, k-means

## 1. Introduction

Image segmentation is the process of partitioning an image into several parts. It is an important field in image processing. With good segmentation results, it is easier for analysis, recognition or further processing.

There are various approaches for image segmentation. Popular approaches include Snake and Balloon (Kass et al. 1987; Cohen 1991), Region Growing (Zucker 1976). Level-Set methods is another famous approach. Level sets based on contour representations was initially propose by (Osher and Sethian 1988). It became popular for image segmentation after (Caselles et al., 1993; Malladi et al., 1995). (Chan and Vese 2001) introduced a level set formulation of the piecewise constant (Mumford and Shah 1989) which is a popular framework for image segmentation.

Another intuitive approach is by Clustering method. However, most popular clustering algorithms, eg. K-means (McQueen 1967), can not produce good segmentation results. In this paper, we will apply a clustering method, SUP proposed by (Chen and Shiu 2007), on image segmentation.

## 2. Segmentation by Clustering

In this section, we discuss how to use clustering ideas on image segmentation.

Each pixel of an image can be viewed as a subject, and to segment an image is equivalent to cluster pixels. Now the question is: what kind of information of each pixel should we use for clustering? The most useful information of each pixel is probably its color intensity. However, it might not be enough to only use color intensities for clustering.

For example, the artificial image shown in Figure 1(a) is a very simple gray-level image. There are only two color intensities in the whole image. Based on only intensities, any reasonable clustering algorithm will partition the whole image into

*Institute of Statistical Science, Academia Sinica, 128 Academia Road Section 2, Taipei, Taiwan, 115

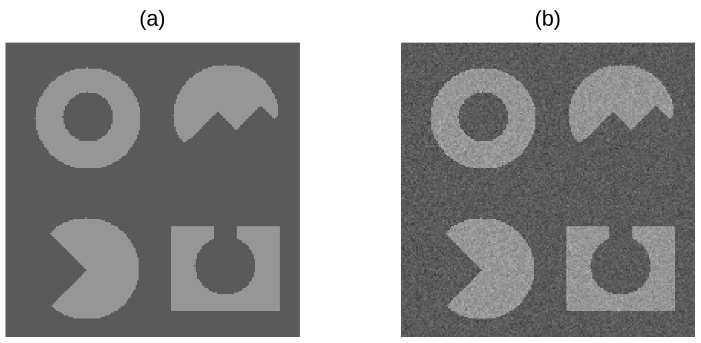

**Figure 1**: (a) artificial gray level image (b) adding i.i.d. Gaussian noise on the image in (a)

two parts, and each of them is of the same color. Since there are four simple objects in the image, an ideal segmentation result should consist of five (or six) parts, the background and four objects (or one more from the background inside the ring). This can be fixed by considering connected components.

For a slightly more complicated image shown in Figure 1(b), it is originally from the image shown in Figure 1(a), and we added i.i.d. Gaussian noise $N(0, 10^2)$ to each pixel. Now the intensities of this image ranges from 48 to 197, and the histogram is shown in Figure 2(a). There are two clear peaks from the histogram, which indicates that there are two groups. To separate these two groups, we picked the valley, intensity value 120, as the threshold. The result image is shown in Figure 2(b). There are many isolated pixels that are assigned to wrong clusters.

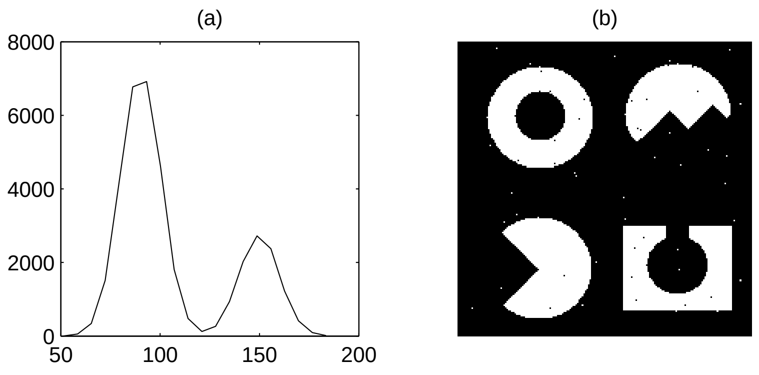

**Figure 2**: (a) The histogram of intensities of the image shown in Figure 1(b) (b) The segmentation result using threshold 120

We can repair those isolated pixels by examining their neighbors. If almost all the neighboring pixels of some pixel belong to the same cluster, but are different from the pixel investigated, this pixel probably should be assigned to the cluster that most of its neighbors belong to. This indicates that the location is also very important. Next question is: how to combine the intensity and the location information?

Each pixel has two variables $(x, y)$ to represent its location. For a gray-level image, the intensity variable is one-dimensional, while that from a color image is three-dimensional. For simplicity, assume that the effect of each variable is linear. We also have to choose the weight between location and intensity. In fact, to estimate the weight is rather difficult. For example, suppose that $I_1$ is an image, and there is a good weight to produce a excellent segmentation. If we scale down $I_1$ by 2 to form a new image $I_2$, the distribution of intensity remains the same while the range of the location variable is reduced by 2. Therefore, the importance of the location variable over the intensity variable is increased by 2 from $I_1$ to $I_2$. Unless there is a good method to estimate the sizes of objects, we can only experiment with various values and choose the best result among them.

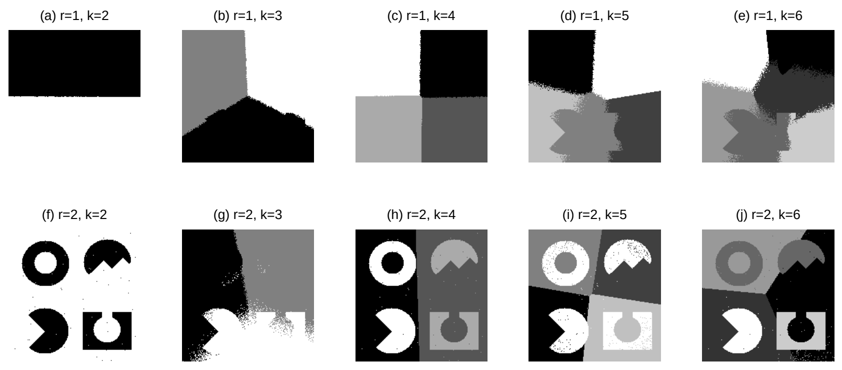

**Figure 3**: Image segmentation results by K-means with different parameters $r$ and $k$

Figure 3 shows the segmentation results by K-means algorithm, where $k$ is the total numbers of clusters and $r$ is the ratio between intensity and location. For example, if $r$=2, we divide the location variable $(x, y)$ by 2. From the results, we find that it is difficult to find a suitable $r$ for K-means. When $r = 2$, there are many isolated pixels, which indicates that the location variable should be more important. However, when we use $r = 1$, not only did the algorithm remove isolated noise, it also destroyed the original shapes of objects.

**Figure 4**: Three image segmentation results with K-means $k = 2$ and $r = 2$

In fact, due to the local minimum problem of most clustering algorithm (including K-means), we might have very different results from the same parameters through experiments. For example, K-means with $k = 2$ and $r = 2$ can produce different results, shown in Figure 4, by chance. Therefore, we propose to use a clustering algorithm that can easily combine distance and location information, and it can avoid the local minimum problem.

## 3. SUP Clustering Algorithm

In this section, we introduce the Self-Updating Process (SUP) clustering algorithm proposed by Chen and Shiu 2007. The central idea of the SUP clustering algorithm can be illustrated by the following example.

Suppose there are a lot of students on the playground. A teacher asks them to form into several groups. What will the students do? Each student will probably move towards others who are closer, with respect to their locations at the playground or to any feature that can characterize the students' relationship. If everyone moves by this rule, the students will gradually form into groups.

SUP is based on the simple and intuitive concept as described above. Suppose there are $N$ subjects to be clustered. For each subject, there are $P$ observations (random variables) representing the subjects' features. Each subject can be viewed as a data point in a $P$-dimensional space. Imitating the aforementioned example, the following mechanism is constructed to move the data points (subjects). The movement of each subject is determined by the between-subject proximity, which can be any measure such as the Euclidean distance or correlations.

The algorithm can be formulated as follows:

1. $x_1^{(0)}, x_2^{(0)}, \cdots, x_N^{(0)} \in R^p$ to be clustered.
2. At time $t+1$, every point is updated according to

$$x_i^{(t+1)} = \frac{\sum_{j=1}^{N} f(x_i^{(t)}, x_j^{(t)}) \cdot x_j^{(t)}}{\sum_{j=1}^{N} f(x_i^{(t)}, x_j^{(t)})}. \tag{1}$$

3. Repeat 2) until every point converges.

$f$ is some statistic that measures the between-subject proximity. Since the role of $f$ is the weight put on each subject to the current updating subject, it is reasonable to have $f$ larger when two subjects are closer. While other $f$ also works fine through experiments, $f$ with a exponential decay with respect to the distance is proposed:

$$f(u,v) = \begin{cases} \exp[-\frac{d}{T}] & d \le r \\ 0 & d > r, \end{cases} \tag{2}$$

where $d$ is the Euclidean distance from $u$ to $v$. $r$ and $T$ are two parameters of this model. It is suggested that SUP with $T = r/5$ can work well.

## 4. SUP on Segmentation

In this section, we propose how to apply SUP clustering algorithm on image segmentation. In addition to choose proper parameters $r$ and $T$ in (2), there is also computation difficulty to overcome.

From (1), we need to calculate $f$ for all possible pairs. Therefore, the complexity of SUP clustering algorithm is $O(N^2)$, where $N$ is the numbers of subjects. When $N$ is large, the computation becomes expensive. For the image shown in Figure 1(b), its size is 192×192. There are $192 \times 192 = 36,684$ pixels. If we apply the original SUP directly, it will take the computer program too much time in computing $f$'s. It may become impossible in practice for larger images, eg. full high definition image of size 1920×1080.

Look at (2), $f = 0$ when the distance between two objects is larger than a threshold $r$. Now if two pixels are far away in location, the overall distance between both will also be large. This suggests that we can only compute $f$ for pixels not too far away each other. Therefore, we modify SUP to update each pixel by the weighted average of pixels that are within distance $R$ to the pixels being updated.

$$x_i^{(t+1)} = \frac{\sum\limits_{j:d(i,j)\leq R} f(x_i^{(t)}, x_j^{(t)}) \cdot x_j^{(t)}}{\sum\limits_{j:d(i,j)\leq R} f(x_i^{(t)}, x_j^{(t)})}. \tag{3}$$

Using this updating scheme, we only have to compute $f$'s for $(2R+1)^2$ times for each pixel. The complexity of the modified algorithm becomes $O(N)$, instead of $O(N^2)$ for the original one.

From our experiment, we found that SUP with $R = 10$ works very well. It is better for larger $R$, but it also takes more time. We also found that SUP with $L^1$ distance works better that with Euclidean distance does. Therefore, we propose to use $L^1$ distance and $R = 10$.

## 5. Simulation Results

In this section, we adopted the scheme discussed in the previous section on image segmentation. First, we apply the algorithm on the simulated image shown in Figure 1(b). Keep the ratio between intensity and location be 1, we experimented SUP with $r$ =10, 20, $\cdots$, 80. The results are shown in Figure 5.

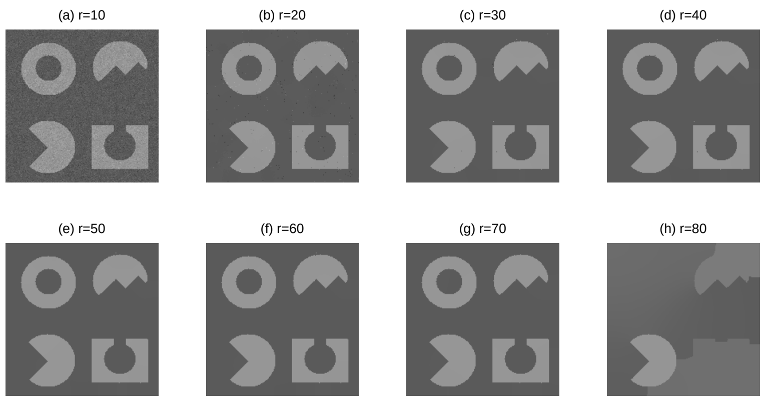

**Figure 5**: Image segmentation results by SUP with the ratio between intensity and location is 1, and $r = 10, 20, \cdots, 80$

For $r = 10$, the image is still noisy. If the overall distance (the sum of the absolute difference in intensity, horizontal and vertical positions) between two pixels is smaller than $r$, two pixels will not affect each other during the updating. Since $r = 10$ is small, there are a lot of clusters in the end. For $r$=20, 30 and 40, the results are very good. There are five main clusters in the end, but there are still a few isolated points. For $r$ =50, 60 and 70, all isolated pixels are removed, and the results are excellent. For $r$ =80, the result is poor. Pixels that should belong to different cluster affected each other due to large $r$.

Next, we experimented SUP by keeping $r$ at 60 and change the ratio from 1 to 4. The results are shown in Figure 6. The four segmentation results are almost identical the same, which tells that the small change in $r$ will not dramatically

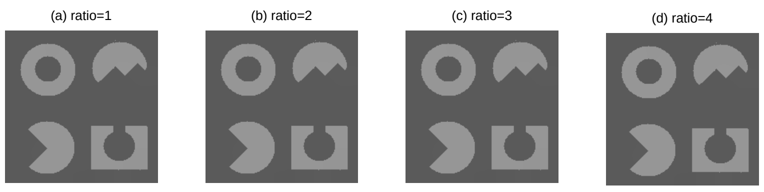

**Figure 6**: Image segmentation results by SUP with $r = 60$ and the ratio between intensity and location is 1, 2, 3, and 4

change the segmentation results. Comparing the segmentation results by K-means shown in Figure 3, the results from SUP are much better.

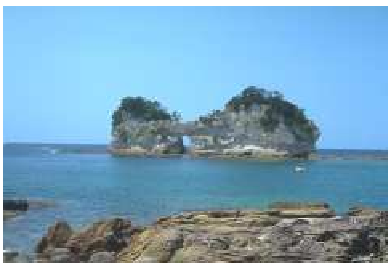

**Figure 7**: Test image

Now we apply SUP on a color image shown in Figure 7. For a color image, the intensity are three dimensional. While RGB representation is widely used, we found that YUV (Y stands for the luma component and U and V are the chrominance components) works better for SUP through our experiments.

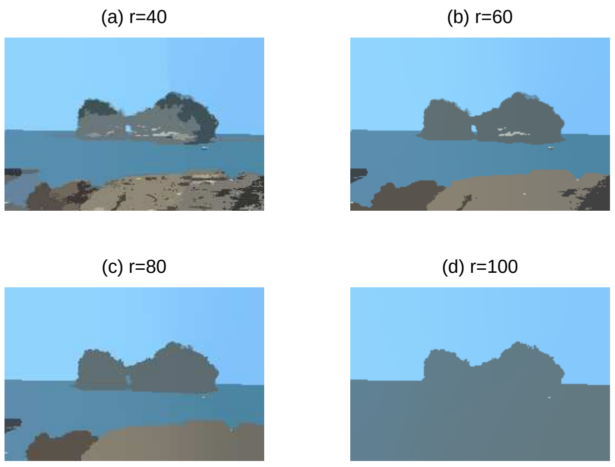

**Figure 8**: Segmentation results by SUP with (a) $r = 40$ (b) $r = 60$ (c) $r = 80$ (d) $r = 100$

Figure 8 shows the results by SUP clustering with $r$ =40, 60, 80, and 100. The ratio between intensity and location is fixed at 3. For $r = 100$, SUP partitioned the whole image into only two parts. For smaller $r$, there are more clusters in the end. Comparing to the results from K-means shown in Figure 9, the results by SUP are again much better than those from K-means.

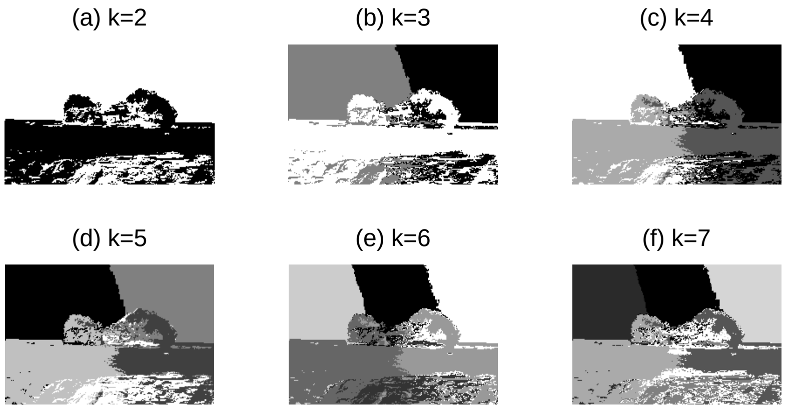

**Figure 9**: Segmentation results by K-means with (a) $k = 2$ (b) $k = 3$ (c) $k = 4$ (d) $k = 5$ (e) $k = 6$ (f) $k = 7$

## 6. Conclusion

We apply SUP clustering algorithm on image segmentation. Our simulation results show that SUP can effectively combine information from intensity and location, and can produce excellent segmentation results.

We modify the original SUP by updating only based on neighboring pixels. This decreases the complexity from $O(N^2)$ to $O(N)$, which largely reduces the computation time.

## REFERENCES

Caselles, V., Catte, F., Coll T., and Dibos, F. (1993), "A geometric model for active contours in image processing," *Numer. Math.*, 66: 1–31.

Chan, T. and Vese, L. (2001). "Active contours without edges." *IEEE Tran. on image processing*, 10(2): 266–277.

Chen, T.-L. and Shiu, S.-Y. (2007). "A New Clustering Algorithm Based on Self-Updating Process," *2007 Proceedings of the American Statistical Association, Statistical Computing Section [CD-ROM]*, Salt Lake City, Utah.

Cohen, L.D. (1991), "On Active Contour Models and Balloons," *CVGIP: Imge Understanding,* 53(2): 211-218.

Kass, M., Witkin, A. and Terzopoulos D. (1987), "Snakes: Active Contour Models," *Proc. lntl Conf. Computer Vision, ICCV87,* London.

Malladi, R., Sethian, J., and Vemuri, B. (1995). "Shape modeling with front propagation: A level set approach", *IEEE trans. on pattern analysis and machine intelligent*, 17(2): 158–175.

McQueen, J. (1967), "Some methods for classification and analysis of multivariate observations," *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 291-297.

Mumford, D. and Shah, J. (1989)," Optimal approximation by piecewise smooth functions and associated variational problems. Comm," *Pure Appl. Math.* 42: 577–685.

Osher, S. and Sethian J. (1988). "Fronts propagating with curvature dependent speed: algorithm based on the Hamilton-Jacobi formulation," *Journal of Computational Physics*, 79: 12–49.

Zucker, S.W. (1976) “Region growing: Childhood and adolescence,” *Comput. Graph. Image Process,* 5: 382-399.