The University of Kansas

Information and Telecommunication Technology Center

A Technical Report of the
Networking and Distributed Systems (NDS) Laboratory

Automated Sea Ice Segmentation Using Spatial Clustering

LeenKiat Soh
Costas Tsatsoulis

ITTC-FY98-TR-11810-01

September 1997

Project Sponsor:

Naval Research Laboratory

1. Introduction

This technical report presents a technique for SAR sea ice segmentation. The objective is to determine the number of classes and the composition of each class in the image automatically. The underlying implementation approach is dynamic local thresholding, divided into regional histogram computation and selection, Gaussian curve approximation, bimodality filtering and optimal threshold determination, significant threshold selection and assignment, regional interpolation, pointwise interpolation, and N-ary decision, as discussed in (Haverkamp *et al.* 1995). However, we have designed a different significant threshold selection strategy, based on multiresolution. Given the significant thresholds and preliminary segmentation, the spatial attribute *variety* is computed. Our clustering method is a spatial clustering technique that we call the Aggregated Population Equalization (APE) that acts upon the attribute *variety*. This method has two modules, one for merging classes, and the other for splitting classes. In this paper, our focus lies on the multiresolution significant threshold selection approach and the APE method.

2. Multiresolution Significant Threshold Selection

Our new significant threshold selection strategy is based on data reduction adapted from (Sezan 1990) and it detects peaks in the histogram. Here, we assume that the peaks are the threshold points where the image should be segmented. Several peak detection algorithms have been proposed in the literature, either for data reduction, segmentation, or enhancement.

One inherent problem in histogram-based peak detection is noise. Histogram tracing algorithms are sensitive to noise and may detect, as a result, spurious peaks. Several techniques have used histogram smoothing prior to processing; for example, Rosenfeld and Torre (1983) proposed smoothing the histogram before computing its convex hull. Also, Horowitz (1977) smoothed the waveform of electrocardiogram using a piecewise linear curve approximation through a split-and-merge approach. The granularity or the closeness of fit of the approximation was controlled by the user: the closer the fit, the less smoothed the waveform was. Others have used embedded smoothing mechanisms within the processing module. Eklundh and Rosenfeld (1979) proposed a peak detection algorithm using difference operators of various sizes applied at all points of the waveform. Our new technique in this paper is also a multiresolution approach.

The basis of our technique is the utilization of the cumulative distribution function (cdf) of the histogram to detect peaks, which has been proposed to eliminate noise effects while processing the histogram. For example, Boukharouba *et al.* (1985) located peaks using the zero-crossings and local extrema of a peak detection signal determined from the curvature of the cdf. To obtain the peak detection signal, the authors fitted the cdf to a Chebyshev polynomial. Subsequently, Sezan (1990) refined the method to lessen computational intensity and reduce the number of user-specified parameters. In what follows, we describe Sezan's method in detail and present our own multiresolution version of the method.

2.1. Cumulative Distribution Function and Peak Detection

As reasoned in (Sezan 1990), the concept of using the cumulative distribution function to detect peaks is as follows. First, generate a peak detection signal from the image histogram. Then, locate the histogram peaks using the zero-crossings of the peak detection signal and the local extrema between the zero-crossings. To obtain the peak detection signal, η_Ω, the histogram, h, is convolved with the peak detection kernel, κ_Ω:

$$\eta_\Omega = \kappa_\Omega \otimes h \tag{1}$$

where $h = H_T(t)$, the histogram, in our application, and

$$\kappa_\Omega = \delta \otimes \Lambda_\Omega \tag{2}$$

where δ is a *differencer* such that

$$\delta(\omega) = \begin{cases} -1, & \omega = -1 \\ 1, & \omega = 0 \end{cases}, \tag{3}$$

and Λ_Ω is a smoother such that

$$\Lambda_\Omega(\omega) = \begin{cases} 1/\Omega, & \omega = -\dfrac{\Omega-1}{2} + 1 \\ \dfrac{(\omega + (\Omega-1)/2)}{\Omega} + \Lambda_\Omega(\omega - 1), & \omega = -\dfrac{\Omega-1}{2} + 2, \ldots, 0 \\ \Lambda_\Omega(-\omega), & \omega = 1, 2, \ldots, -\dfrac{\Omega-1}{2} - 1 \end{cases}, \tag{4}$$

where the parameter Ω is assumed to be odd, and the extent of Λ_Ω is $\Omega - 2$. Combining Equations 2, 3, and 4 yields

$$\kappa_\Omega(\omega) = \delta(\omega) \otimes \Lambda_\Omega(\omega)$$
$$= \begin{cases} -\left(\frac{\Omega-1}{2} + \omega + 1\right)\frac{1}{\Omega}, & -\frac{\Omega-1}{2} \le \omega \le -1 \\ \left(\frac{\Omega-1}{2}\right)\frac{1}{\Omega}, & \omega = 0 \\ \left(\frac{\Omega-1}{2} - \omega\right)\frac{1}{\Omega}, & 1 \le \omega \le \frac{\Omega-1}{2} - 1 \end{cases} \qquad (5)$$

Sezan (1990) referred to the parameter Ω as the *peak-detection parameter.* This parameter is particularly important in determining the sensitivity of the algorithm to noisy peaks in the histogram. Figures 1 and 2 show the smoother and the peak detection kernel, respectively, for $\Omega = 11$.

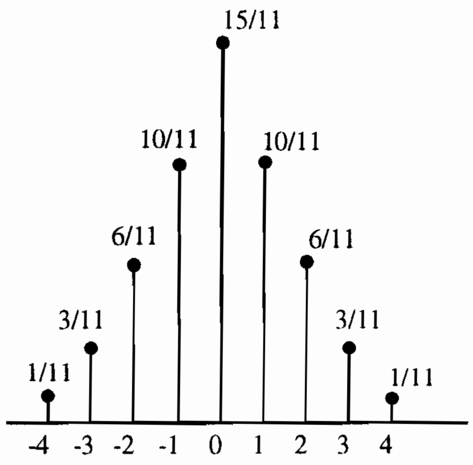

Figure 1. The smoother, $\Lambda_{11}(\omega)$, for $-4 \le \omega \le 4$, and $\Lambda_{11}(\omega) = 0$ otherwise (Equation 4).

According to Sezan (1990), the convolution with the differencer approximates first-order differentiation. For an ideally smooth histogram, the peaks could be located from the sign and the zero-crossings of the convolution of h with δ. However, practically, such histogram does not exist and smoothing must be applied prior to differencing. The smoothing is as follows. At a particular bin, say, ω_0, the convolution of the peak detection kernel with the histogram results in the difference of the weighted sum of histogram values at $(\Omega-1)/2$ bins prior to ω_0 and that of histogram values at $(\Omega-1)/2$ beyond ω_0. Therefore, as the value of Ω decreases, the convolution mask or interval essentially becomes smaller, and thus is more sensitive to local variations. For larger Ωs, the averaging effect is greater and only large peaks give strong responses to the algorithm.

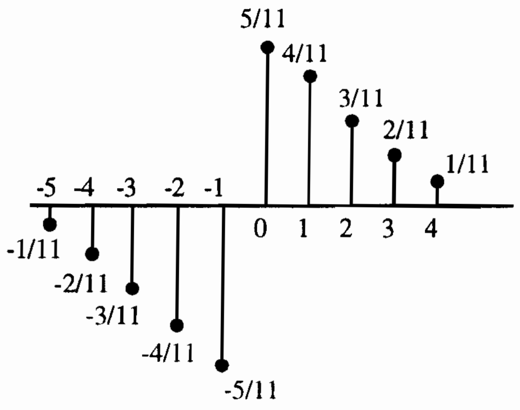

Figure 2. The peak detection kernel, $\kappa_{11}(\omega)$, for $-5 \le \omega \le 4$, and $\kappa_{11}(\omega)=0$ otherwise (Equation 5).

To implement the above strategy, Sezan (1990) formulated that the detection signal in Equation 1 could be generated from the image cdf via a simple averaging operation. He showed that

$$\eta_\Omega = c - \bar{c}_\Omega = \kappa_\Omega \otimes h \tag{6}$$

where c is the cdf of h, and $\bar{c}_\Omega$ is the convolution of the cdf $c(\omega)$ with the uniform rectangular window,

$$w_\Omega(\omega) = 1/\Omega, -(\Omega-1)/2 \le \omega \le (\Omega-1)/2 \tag{7}$$

such that

$$\bar{c}_\Omega(\omega) = \sum_{k=-(\Omega-1)/2}^{(\Omega-1)/2} c(\omega-k)w_\Omega(k). \tag{8}$$

Equation 8 can be separated into three parts:

$$\bar{c}_\Omega(\omega) = \sum_{k=-(\Omega-1)/2}^{-1} c(\omega-k)w_\Omega(k) + c(\omega)w_\Omega(0) + \sum_{k=1}^{(\Omega-1)/2} c(\omega-k)w_\Omega(k). \tag{9}$$

Expressing $c(\omega-k)$ in terms of the histogram, $h(\omega)$, we have

$$c(\omega-k) = \begin{cases} c(\omega) - \sum_{l=\omega-k+1}^{\omega} h(l), & k>0 \\ c(\omega) + \sum_{l=\omega+1}^{\omega-k} h(l), & k<0 \end{cases}. \tag{10}$$

Substituting Equation 10 into Equation 9, we obtain

$$\begin{aligned} \bar{c}_\Omega(\omega) = & \sum_{k=-(\Omega-1)/2}^{-1} \left[c(\omega) + \sum_{l=\omega+1}^{\omega-k} h(l) \right] w_\Omega(k) + c(\omega)w_\Omega(0) \\ & + \sum_{k=1}^{(\Omega-1)/2} \left[c(\omega) + \sum_{l=\omega-k+1}^{\omega} h(l) \right] w_\Omega(k) \end{aligned} \tag{11}$$

Note that since $\sum_{\omega=0}^{\Omega-1} w_\Omega(\omega) = 1$, we have

$$\bar{c}_\Omega(\omega) = c(\omega) + \sum_{k=1}^{(\Omega-1)/2} w_\Omega(k) \sum_{l=\omega+1}^{\omega+k} h(l) - \sum_{k=1}^{(\Omega-1)/2} w_\Omega(k) \sum_{l=\omega-k+1}^{\omega} h(l). \tag{12}$$

Thus, η_Ω can be rewritten as

$$\begin{aligned} \eta_\Omega(\omega) &= c(\omega) - \bar{c}_\Omega(\omega) \\ &= \sum_{k=1}^{(\Omega-1)/2} w_\Omega(\omega) \sum_{l=\omega-k+1}^{\omega} h(l) - \sum_{k=1}^{(\Omega-1)/2} w_\Omega(\omega) \sum_{l=\omega+1}^{\omega+k} h(l). \\ &= \sum_{k=1}^{(\Omega-1)/2} w_\Omega(\omega) \left(\sum_{l=\omega-k+1}^{\omega} h(l) - \sum_{l=\omega+1}^{\omega+k} h(l) \right) \end{aligned} \tag{13}$$

Expanding the above expression for the values $\Omega = 3, 5, 7, \ldots, K_\Omega$, where K_Ω is odd, we have

$$\eta_3(\omega)=\frac{1}{3}\left(h(\omega)-h(\omega+1)\right)$$
$$\eta_5(\omega)=\frac{1}{5}\left(h(\omega-1)+2h(\omega)-2h(\omega+1)-h(\omega+2)\right)$$
$$\eta_7(\omega)=\frac{1}{7}\begin{pmatrix}h(\omega-2)+2h(\omega-1)+3h(\omega)-\\3h(\omega+1)-2h(\omega+2)-h(\omega+3)\end{pmatrix}$$

$$\cdots$$
$$\cdots$$

$$\eta_{K_\Omega}(\omega)=\frac{1}{K_\Omega}\begin{pmatrix}h\left(\omega-\left(\frac{K_\Omega-1}{2}-1\right)\right)+2h\left(\omega-\left(\frac{K_\Omega-1}{2}-2\right)\right)+\cdots\\+\frac{K_\Omega-1}{2}h(\omega)+\cdots-h\left(\omega+\frac{K_\Omega-1}{2}\right)\end{pmatrix}. \qquad (14)$$

From the above equations, we observe that $\eta_\Omega(\omega)$, $\Omega=3,5,7,\ldots,K_\Omega$ are the result of convolving $h(\omega)$ with κ_Ω where κ_Ω is exactly the definition found in Equation 5, with $\Omega=3,5,7,\ldots,K_\Omega$. Therefore, $\eta_\Omega=c-\bar{c}_\Omega=\kappa_\Omega\otimes h$, as stated in Equation 6.

Thus, we have obtained an implementation approach based on cdf of the histogram, which corresponds to convolution of the signal with a kernel. Sezan (1990) further postulated several principles to estimate the *start*, *end*, and *maximum* points of the peaks as the following.

- A zero-crossing (similar to that of Laplacian second order derivative), of the peak detection signal, η_Ω, to negative values—a negative crossover—signifies the *start* of a peak. The bin value, ω, at which the negative crossover occurs is defined to be the estimate of a *start* point. Thus, the start point of the ith peak is ω_i^s; the next negative crossover at the bin ω_{i+1}^s estimates the start of the next peak, and so on.

- A zero-crossing of the peak detection signal, η_Ω, to positive values—a positive crossover—following a negative crossover estimates the bin values at which the peak reaches its maximum. The maximum of the ith peak is ω_i^m.

- The bin value between two successive negative crossovers, ω_i^s and ω_{i+1}^s, at which the detection signal, η_Ω, achieves its local maximum is defined to be the estimate of the end point of the peak. The end point of the ith peak is ω_i^e.

Practically, since the histogram is discrete and so is the detection signal, the zero-crossing of a signal can occur in two situations. If the signal attains the zero value, then the exact point of the zero-crossing can be obtained. For a crossover, the bin value of the leader of the transition becomes the crossover point location. Given these considerations, the ith peak can be represented by a triplet of $\langle \omega_i^s, \omega_i^m, \omega_i^e \rangle$. Figure 3 illustrates an ideal detection signal and the triplet parameters.

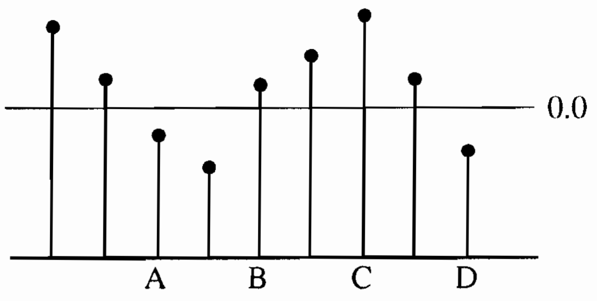

Figure 3. An ideal peak detection signal and the parameters of the peaks. Suppose that we are at the ith peak. A is the first negative crossover point, and it is the start of the ith peak, ω_i^s. D is the second negative crossover point, and it becomes the start of the $i+1$th peak, ω_{i+1}^s. C attains the maximum value between the two start points, A and D, and it thus estimates the end of the ith peak, ω_i^e. Finally, B is the first positive crossover point following the first negative crossover point: the location of the maximum of the ith peak, ω_i^m.

Given the definition and implementation means of peak detection, Sezan (1990) applied them to histogram-based image data reduction. The author defined the peaks as

$$P_\Omega = \left\{ \langle \omega_i^s, \omega_i^m, \omega_i^e \rangle, i = 1, 2, \ldots, N_p \right\}, \tag{15}$$

where N_p is the number of peaks found in the histogram. The author used the peaks to derive a set of thresholds,

$$T_\Omega = \left\{ t_i : t_i = \upsilon e_i + (1 - \upsilon)\omega_{i+1}^s, i = 1, 2, \ldots, N_p - 1; 0 \le \upsilon \le 1 \right\}. \tag{16}$$

The parameter υ was used as an adjuster to finalize the location of a threshold between two peaks. By setting υ to zero, the author declared that each peak location was a threshold and grouped gray level clusters $C_1 = \left(0, \omega_2^s - 1\right)$, $C_2 = \left(\omega_2^s, \omega_3^s - 1\right)$, …, and $C_{N_p} = \left(\omega_{N_p}^s, I_{\max} - 1\right)$, where $I_{\max}$ was the total number of intensity levels. To generate the final quantized image, the author assigned to each cluster the intensity at which the associated peaks attained their maxima, such that

$$L\left(C_i\right) = \omega_i^m. \tag{17}$$

To eliminate neighboring peaks in close proximity, a *closeness* criterion, ε_{close}, was used to merge them. Given the triplets of two peaks, $\left\langle \omega_i^s, \omega_i^m, \omega_i^e \right\rangle$ and $\left\langle \omega_{i+1}^s, \omega_{i+1}^m, \omega_{i+1}^e \right\rangle$, if

$$\omega_{i+1}^s - \omega_i^e \le \varepsilon_{close}, \tag{18}$$

then the two peaks were merged into a single peak with an updated triplet of

$$\left\langle \omega_i^s, \max\left(\omega_i^m, \omega_{i+1}^m\right), \omega_{i+1}^e \right\rangle. \tag{19}$$

2.2. Multiresolution Peak Detection (MRPD)

One inherent disadvantage of Sezan's algorithm is the determination of the peak detection parameter, Ω. As indicated before, if this number is small, then the algorithm responds to local variations and noise. On the other hand, if this number is large, then the algorithm may overlook legitimate peaks because of the averaging effect. We propose a new method called the Multiresolution Peak Detection (MRPD) to address this issue, utilizing the advantages of the multiresolution approach in handling details and noise. The concept underlying our approach is as follows: First, we create a multiresolution map of the histogram peaks. Second, we track each peak through the scale space to assess each peak's significance. Third, we filter out spurious peaks and localize significant peaks. The essential part of this concept lies with the assessment of peak significance—we use a weighted neighborhood to fuse together different scales of information, as will be elaborated in the following. The MRPD algorithm has been designed based on several assumptions:

- Peaks found at a low-level resolution are more significant than the peaks found at a high-level resolution. This concurs with the normal usage of multiresolution in the literature.

- Peaks found in high-level resolution are more accurate in terms of localization than the peaks found in low-level resolution. This also concurs with the concept of multiresolution.

- Neighboring peaks suggest a dominant peak. A peak is more significant if it is surrounded (in a 2- or higher-dimensional space) or flanked (in 1D space) by other peaks than a peak without such neighbors. Viewing this assumption from the noise modeling angle, a single peak is more likely to be a noisy assertion than a cluster of peaks.

- The significance of a peak is proportional to its height.

With the above assumptions, we have designed a multiresolution peak detection algorithm, with its block diagram illustrated in Figure 4. We describe each module in detail in the following subsections. The first module generates the cumulative distributed function from the histogram. Then, the second module computes the maximum peak detection parameter, Ω_{max}, such that a multiresolution map can be created using $\Omega = 3, 5, 7, \ldots, \Omega_{max}$. The fourth module tracks the peaks to assess their significance. The fifth module merges adjacent peaks. Finally, a filter is used to extract a set of peaks to become the set of significant thresholds.

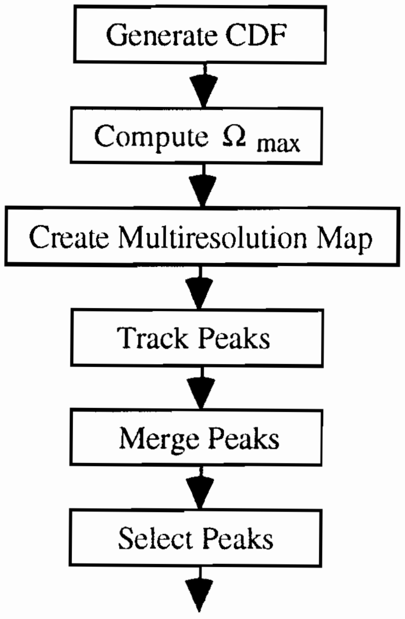

Figure 4. Block diagram of our multiresolution peak detection algorithm.

2.2.1. Generate the Cumulative Distribution Function

After the optimal threshold selection, we obtain a set of thresholds $T = \{t_{r1}, t_{r2}, \cdots, t_{rN_{rt}}\}$ and $0 \le t_{ri} \le 255$, where N_{rt} is the number of regions with a computed threshold value. The objective of the MRPD is to extract from T a set of significant thresholds, T_S. First, a histogram of thresholds is computed:

$$H_T(t) = \underset{t_i \in T}{\#}(t = t_i) \qquad t = 0 \ldots 255. \tag{20}$$

From the histogram, the cumulative distributed function (cdf) is obtained

$$\begin{aligned} cdf_T(0) &= H_T(0) \\ cdf_T(t) &= cdf_T(t-1) + H_T(t) \qquad t = 1 \ldots 255 \end{aligned} \tag{21}$$

2.2.2. Compute the Maximum Scale Size

To create a multiresolution map, we apply the peak detection algorithm to the histogram several times with different values of the peak detection parameter, or scale in our application. Theoretically, the minimum scale size is 1, which involves no averaging and no involvement of neighbors in the signal. Practically, we set the minimum scale size to 3. Thus, the highest resolution level of the map is when the cumulative distributed function has been smoothed with a scale size of 3. To determine the lowest resolution level, we have designed a strategy to compute the maximum scale size, or $\Omega_{\max}$. Suppose we define a subset of T called the *nonzero threshold set*, such that

$$T_{nonzero} = \{t | H_T(t) \ne 0\}. \tag{22}$$

We then further define the range of the threshold set T as

$$R_T = \max(T_{nonzero}) - \min(T_{nonzero}) + 1. \tag{23}$$

Our initial computation for the maximum scale size is the smallest odd number, Ω, that satisfies

$$\Omega > \frac{R_T}{\Psi}. \tag{24}$$

The denominator Ψ, a number greater than 0, is called the *Ideal Domain Class Number* (IDCM). The IDCM is the scientist's assessment of the number of classes in an ideal histogram of a particular domain. At a glance, one might argue that IDCM is simply a transformation of the peak detection parameter used in Sezan (1990). They are *different*. Originally, Sezan (1990) used Ω, the peak detection parameter, to control the averaging effects, and thus this parameter directly and significantly influenced the number of detected peaks. In our case, the key factor in determining the number of potential peaks is the range of the histogram, R_T. Ideally, the maximum scale size should be the range of the signal, which is R_T. Thus, in general, if R_T is large, we have a higher probability of finding more peaks than when R_T is small. However, with the image intensities ranging in 256 levels, the maximum scale size could be 255 if we use R_T as $\Omega_{\max}$. That implies a resolution map of 127 scale levels! Hence, we introduce the use of the IDCM. This parameter improves the efficiency of our algorithm computationally without damaging the accuracy of the scale space. Another advantage of using IDCM is that it allows the incorporation of domain knowledge into the peak detection process. This flexibility allows scientists to customize our new program to segment images when the number of classes is known *a priori* and to estimate the normal number of classes. For example, if the research domain is document processing which involves only two classes of intensity levels, one might want to set the IDCM to 2. Note that by setting the IDCM to 2 does not guarantee one and only one detected peak. However, with such a low Ψ, one can expect a low turnout of detected peaks and number of segmented classes. In addition, if the domain deals with imagery with number of classes varying within a range, one might want to set the IDCM to a number closer to the upper limit of that range. This imposes a slight control on the number of detected peaks. For SAR sea ice imagery, one normally deals with 1 to 8 classes in an image. We thus set Ψ to 6 in our implementation. This reduces the maximum number of scale levels to 21 using the equation:

$$N_{Scale} = \left\lfloor \frac{\lceil R/\Psi \rceil}{2} \right\rfloor, \tag{25}$$

where R is an integer denoting the range of an item, e.g., histogram, image, etc. The denominator 2 is used because we only use odd values for our scale size. A brief run through of our calculation is as follows: R is 255 for maximum image range. Divide 255 with 6 yields 42. Since we are looking for an odd number greater than 42.5 (Equation 24), we use a ceiling function to obtain 43. Divide 43 with 2 yields 21. And the floor function of 21.5 is 21. 21 is thus the maximum number of scale levels for our multiresolution map. Even though the ideal domain class number is six, we have detected as many as 30 peaks in some threshold histograms, showing that Ψ improves the computational efficiency and does not constrain the search for peaks.

The process of obtaining $\Omega_{\max}$ is as follows. First, we obtain the initial $\Omega_{\max}$ using Equation 24. Then, we apply a smoothing operation to the threshold histogram, H_T, to derive a $\Omega_{\max}$-smoothed histogram:

$$H_{T,\Omega_{\max}}(t) = \frac{\sum_{i=t-(\Omega_{\max}-1)/2}^{t+(\Omega_{\max}-1)/2} H_T(i)}{\Omega_{\max}}. \tag{26}$$

From $H_{T,\Omega_{\max}}$, we obtain the nonzero version of T,

$$T_{nonzero,\Omega_{\max}} = \left\{ t \middle| H_{T,\Omega_{\max}}(t) \neq 0 \right\}, \tag{27}$$

and the smoothed range,

$$R_{T,\Omega_{\max}} = \max\left(T_{nonzero,\Omega_{\max}}\right) - \min\left(T_{nonzero,\Omega_{\max}}\right). \tag{28}$$

Then, we compare the ranges, R_T (Equation 23) and $R_{T,\Omega_{\max}}$ (Equation 28), such that if

$$R_{T,\Omega_{\max}} - R_T > \frac{\Omega_{\max}}{2}, \tag{29}$$

then we reduce $\Omega_{\max}$ by 2 and reconsider Equation 29 until the condition is satisfied. If the condition is satisfied, then we have computed the final $\Omega_{\max}$. This condition is important to prevent over-averaging of the histogram that shortens the intensity range of the image considerably. Without this condition to refine $\Omega_{\max}$, we might have *tail-erosion* effects where pixels of intensities at both ends of the spectrum are not segmented correctly. Also, if the initial and final $\Omega_{\max}$ are different, then we note that our best choice (the initial value) of $\Omega_{\max}$ has not qualified, and we set a Boolean variable *range_compress* to *TRUE*. This variable is used later to promote thresholds into the significant threshold set. Given the final $\Omega_{\max}$, we compute the number of scale levels as

$$N_{scale} = \frac{\Omega_{\max} - \Omega_{\min}}{2} + 1. \tag{30}$$

This number is the same as the one of Equation 25 when $\Omega_{\max}$ is 43, and $\Omega_{\min}$ is 3 such that N_{scale} is 21.

2.2.3. Create Multiresolution Map

After obtaining $\Omega_{\max}$, we are ready to generate the multiresolution map using $\Omega = 3,5,7,\ldots,\Omega_{\max}$. Now, for each value of Ω, given the cumulative distribution function (Equation 21), we compute the average cumulative distribution function as follows[1]:

$$\overline{cdf}_{T,\Omega}(t) = \frac{\sum_{i=t-(\Omega-1)/2}^{t+(\Omega-1)/2} cdf_T(i)}{\Omega}. \tag{31}$$

We obtain the peak detection signal from Equations 21 and 31:

$$\eta_\Omega(t) = cdf_T(t) - \overline{cdf}_{T,\Omega}(t). \tag{32}$$

Given the signal, we proceed to find the peaks, and each triplet $\langle \omega_i^s, \omega_i^m, \omega_i^e \rangle$, accordingly. Note that the index i denotes the peaks locally for each resolution level. To be able to propagate the peak information and track the peaks, we must translate the index to a common axis such that detected peaks are not confused throughout the scale space. To do so, we use the intensity or the threshold histogram bin to tag each peak. Specifically, we define ω_i^m as the location where the ith peak occurs. The translation is as follows. We create a quadruple, parallel set of peaks and their descriptions for each resolution level Ω, $\langle P,S,E,W \rangle_\Omega$, that describes a peak's existence, starting point, ending point, and weight. Thus, for a peak located at ω_i^m, we have

$$P_\Omega(\omega_i^m) = 1, \tag{33}$$

[1] In practice, the first Ω and the last Ω bins of the histogram will not be averaged. Instead, the average cumulative distribution function at these bins will be the cumulative distribution function of these bins. Actually, most SAR sea ice images do not occupy the whole range of intensity capacity and thus, most lower and upper ends of the 256-level range have zero-valued frequencies. This simplifies the computational task.

$$S_\Omega(\omega_i^m) = \omega_i^s, \text{ and} \tag{34}$$

$$E_\Omega(\omega_i^m) = \omega_i^e. \tag{35}$$

The local weight of a peak is based on its cdf difference with its neighbor and its height (the frequency of the bin):

$$W_\Omega(\omega_i^m) = \frac{\eta_\Omega(\omega_i^m) - \eta_\Omega(\omega_i^m - 1)}{1 + \eta_\Omega(\omega_i^m) - \eta_\Omega(\omega_i^m - 1)} + \frac{H_T(\omega_i^m)}{\max(H_T) \cdot N_{scale}}. \tag{36}$$

The first term in Equation 36 measures the dominance of the peak—the magnitude of the positive cross over. The higher the magnitude (the greater the difference), the more the term approaches 1. The second term measures the significance of the peak—the frequency of the bin in the original threshold histogram. The denominator is a normalization factor. $\max(H_T)$ is the height of the tallest peak in the histogram and thus $H_T(\omega_i^m)/\max(H_T)$ is always less than or equal to 1. In addition, since we are summing the weights throughout N_{scale} levels, we divide the significance term with N_{scale}. In other words, the first term measures the relative height of a peak while the second the absolute height of a peak. Comparing the two terms, we see that the first term factors more heavily into the equation. This is consistent with our analysis. A peak is better defined when it is higher than its neighbors than when it is simply high. For each $\langle P, S, E, W\rangle_\Omega$ there are two important subsets. First, the set of peaks is defined as:

$$\mathrm{P}_\Omega = \langle P, S, E, W | P(t) = 1, t = 0\ldots255\rangle_\Omega. \tag{37}$$

Second, the set of non-peaks is defined as:

$$\mathrm{N}_\Omega = \langle P, S, E, W | P(t) = 0, t = 0\ldots255\rangle_\Omega. \tag{38}$$

Note that $\mathrm{P}_\Omega \cap \mathrm{N}_\Omega = \varnothing$ and $\mathrm{P}_\Omega \cup \mathrm{N}_\Omega = \langle P, S, E, W\rangle_\Omega$. This completes the translation. The multiresolution map starting from $\Omega_{\min}$ to $\Omega_{\max}$ is

$$\mathrm{M}_{\Omega_{\min},\Omega_{\max}} = \begin{vmatrix} \langle P,S,E,W\rangle_{\Omega_{\min}} \\ \langle P,S,E,W\rangle_{\Omega_{\min}+2} \\ \langle P,S,E,W\rangle_{\Omega_{\min}+4} \\ \cdots \\ \langle P,S,E,W\rangle_{\Omega_{\max}-2} \\ \langle P,S,E,W\rangle_{\Omega_{\max}} \end{vmatrix}. \qquad (39)$$

2.2.4. Track Peaks through the Scale Space

To track peaks, we analyze the multiresolution map, $\mathrm{M}_{\Omega_{\min},\Omega_{\max}}$. At each scale level Ω, our task is to examine the neighbors of a peak to improve the weight or significance of the peak. Thus, for a peak situated at t in the set of P_Ω, we increase its weight:

$$W'_\Omega(t) = W_\Omega(t) + \sum_{j=t-(\Omega-1)/2}^{t-1} \frac{P_\Omega(j)}{|j-t|} + P_\Omega(t) + \sum_{j=t+1}^{t+(\Omega-1)/2} \frac{P_\Omega(j)}{|j-t|}. \qquad (40)$$

As it has been noted in previous section, when $P_\Omega(j) \in \mathrm{P}_\Omega$, it is 1; when $P_\Omega(j) \in \mathrm{N}_\Omega$, it is 0. The two summation terms collect the neighboring peaks as evidence for t as a peak, weighted by the distance of those peaks from t. This satisfies one of our principles in significant threshold selection that a peak is more likely if it is found in a cluster of other peaks than if it is isolated. The single term $P_\Omega(t)$ acts as a self-assurance weight. Across $\mathrm{M}_{\Omega_{\min},\Omega_{\max}}$ we accumulate all weights to obtain

$$W_{\Omega_{\min},\Omega_{\max}}(t) = \sum_{\Omega=\Omega_{\min}}^{\Omega_{\max}} W_\Omega(t) \qquad t = 0\ldots255. \qquad (41)$$

For a same location t, it may not be a peak at all scale levels. Thus, $P_\Omega(t)$ may be 1 or 0. By adding the term into Equation 40, we assure that if an isolated peak at t exists across the scale space often enough, then it will be preserved as a peak. More about this is described in the next sections. Equations 40 and 41 represent our tracking strategy.

We define tracking as a process of collecting or fusing information across the scale space (Witkin 1984). There are three general approaches. First, by following the information at the lowest resolution level to the highest or until such level that the current trail has no way to continue. This approach selectively links the information between each scale level and the decision is local. This is usually employed in edge detection, from detection to localization, and is known as focusing (Hummel *et al.* 1987, Bergholm 1987, Sjoberg and Bergholm 1988). This type of scale space tracking has also been utilized in characterizing and reasoning about the validity of edge pixels (Mallat and Zhong 1992, Lu and Jain 1992, and Kakarala and Hero 1992) and feature extraction (Jackway and Deriche 1996). Another approach is by means of interpolation that links all information points together to form a continuous sheet. Compared to the first, this approach is less selective, and it aims at utilizing and moderating the whole surface of the scale space. This has been used in wavelet representation of multiresolution (Yaou and Chang 1994, Reissell 1996). The third approach also uses the whole scale space but without modifying the information points. This group of tracking techniques is the most popular. For example, Greenspan *et al.* (1994) generated power maps of different resolution levels and computed a set of features for each level. The authors then collected all features and produced 15-dimensional feature vectors as inputs to a K-means clustering algorithm to learn discrimination rules. Goudail *et al.* (1996) fed vectors generated from different resolutions of an image into a neural network to train the network to perform face-recognition. Atiquzzaman (1992) generated several layers of resolution, computed the Hough transform for each layer, and added the results in the accumulator arrays. The author showed that this multiresolution Hough transform, coupled with parameter range reduction, improved the speed of the process significantly.

There are other scale space tracking approaches. Whitten (1993) solved numerically the partial differential equations that delimited the scale space trajectory of the system. Rosin and West (1995) proposed using various saliency factors such as edge strength, length, and curvature to complement the map of multiscale distance transform in edge detection. Whitaker and Pizer (1993) employed a technique of edge-affected diffusion, similar to the focusing approach above, where blurring was limited by the presence of edges as measured at the scale of interest. The authors repeated the process and measured gradients at successively smaller scales to trace a trail through scale space. This allowed them to preserve accurate boundary information and selectively remove objects that fell below a scale of interest. With the same flavor as the previous work, Koplowitz and DeLeone (1996) improved the focusing approach by incorporating the theory of data transmission. The authors designed a progressive system to go from coarse to fine resolution levels such that the same number of bits of transmission was preserved. Gunsel *et al.* (1996) designed a nonlinear multiscale boundary detection method that balanced the tradeoff between the detection and localization goals. The authors utilized multiscale representation of coupled Markov random fields and applied a stochastic regularization scheme based on the Bayesian approach. This allowed a robust integration of boundary information extracted at multiple scales simultaneously. In a different setting, Heitz *et al.* (1994) used a multiresolution relaxation algorithm to minimize global energy function over nested subspaces of the original space of possible scales. Gauch and Pizer (1993) studied comprehensively multiresolution representation of ridges and valleys in an image for image processing. Based on vertex curve and watershed analyses, the authors used Gaussian blurring to create a multiresolution map. To track the critical points across scale levels, the authors employed a fast heuristic. Given an intensity minimum at position $\langle x, y\rangle$ at blurring level l_b, the authors followed the image gradient downhill from position $\langle x, y\rangle$ in level l_b+1 until another intensity minimum was encountered. If there were duplicates arriving at the same position, the extremum with the shortest distance was selected as the normal link, whereas the other links were recorded as annihilation links.

Our tracking strategy is similar to the third general approach in which all information points at each scale are used constructively to generate the final results. The detection factor at low resolution levels is implied when we increase a peak's weight by involving the neighboring peaks. This is because at high resolution levels, a peak has more neighboring peaks than it does at low resolution levels. A peak at low resolution levels prompts a collective effort of its neighboring peaks to enhance its own significance. The localization factor at the high resolution levels is implied when we use the self-assurance term to determine the frequency of a detected peak.

2.2. Merge Adjacent Peaks

Given the tracked peaks and the fused weights, $W_{\Omega_{\min},\Omega_{\max}}$, it is possible to have adjacent peaks due to shifting in locations across the scale space. Suppose while tracing the $W_{\Omega_{\min},\Omega_{\max}}$ for all bins, we locate a pair of adjacent peaks, situated at $t_{c(1)}$ and $t_{c(2)}$, such that $W_{\Omega_{\min},\Omega_{\max}}\left(t_{c(1)}\right)>0$, $W_{\Omega_{\min},\Omega_{\max}}\left(t_{c(2)}\right)>0$, and $t_{c(2)}=t_{c(1)}+1$, where the subscript $c(\bullet)$ denotes a continuum. Upon locating such event, we look ahead from $t_{c(2)}$ to locate the end of $c(\bullet)$, at $t_{c\left(N_{c(\bullet)}\right)}$, such that

$$W_{\Omega_{\min},\Omega_{\max}}(t)>0 \qquad t=t_{c(1)}\ldots t_{c\left(N_{c(\bullet)}\right)}, \tag{42}$$

and

$$W_{\Omega_{\min},\Omega_{\max}}\left(t_{c\left(N_{c(\bullet)}\right)}+1\right)=0, \tag{43}$$

where $N_{c(\bullet)}$ is the number of points in the continuum. Usually, $N_{c(\bullet)}$ is 2. Given the continuum, we compute a new sum of weights

$$W^{\Sigma}_{\Omega_{\min},\Omega_{\max}}=\sum_{i=1}^{N_{c(\bullet)}} W_{\Omega_{\min},\Omega_{\max}}\left(t_{c(i)}\right). \tag{44}$$

We also identify the peak that yields the greatest height:

$$t_{c(\bullet),\max}=t_{c(i)}\left| W_{\Omega_{\min},\Omega_{\max}}\left(t_{c(i)}\right)\geq W_{\Omega_{\min},\Omega_{\max}}\left(t_{c(j)}\right)\quad j=1\ldots N_{c(\bullet)}\right.. \tag{45}$$

To consolidate the continuum and create a new peak, we do the following:

$$\begin{aligned} &W_{\Omega_{\min},\Omega_{\max}}\left(t_{c(i)}\right)=0 \qquad i=1\ldots N_{c(\bullet)}, i\neq t_{c(\bullet),\max},\\ &W_{\Omega_{\min},\Omega_{\max}}\left(t_{c(\bullet),\max}\right)=W^{\Sigma}_{\Omega_{\min},\Omega_{\max}}. \end{aligned} \tag{46}$$

There are two implied goals accomplished by this scheme. First, it enhances the significance of the new peak by absorbing weights of the neighboring peaks. Hence, a cluster of insignificant peaks may be able to become one significant peak after merging. Second, it sharpens the localization of peaks by pinpointing the location of a peak out of a group.

2.2.6. Select Peaks

All nonzero values of $W_{\Omega_{\min},\Omega_{\max}}$ are technically peaks. But, in practice, some of these peaks may be considered noise. Thus, we set a standard to filter out weak peaks. We compute an image-dependent test such that a significant peak must be greater than or equal to the following qualifier:

$$W_{qualify}=\omega_W N_{scale}. \tag{47}$$

The qualifying weight, $W_{qualify}$, is directly proportional to the number of scale levels used. The adjuster ω_W is currently set at 0. This choice of value is based on an ideal view of the multiresolution map. Suppose there is a smooth and obvious modality in the threshold histogram.

Then, the peaks are clearly defined without neighboring peaks. In addition, across scale levels, the peaks, ideally, occupy the same location. Now, we propose that for a peak to be significant in an ideal situation, it must at least exist half of the times in the multiresolution map. If the number of scale levels is 6, then it must exist at least in 3 resolution levels. To have more peaks, one can relax the adjuster by lowering it, and vice versa. This adjuster does not influence the process of peak detection. However, it does affect the number of selected peaks as a final result.

In addition to the selection, we employ one heuristic to promote thresholds to significant peaks. If the Boolean variable *range_compress* is *TRUE*, meaning that the Ω_{max} used has not been the initially chosen scale size, and if $H_T(t) \neq 0$ and $H_{T,\Omega_{max}}(t) = 0$, and t is not adjacent to other peaks, then promote t to a significant peak. If there is such a t, that implies that within the interval with the size of Ω_{max}, there are no other peaks, weak or significant. It also implies that the t must have been eliminated due to some compression in range, that is, *tail-erosion*. So, this second heuristic increases the range of the set of the significant thresholds (peaks) and provides one or two representative peaks for pixels occupying both ends of the histogram.

After selection and promotion, the significant peaks are grouped into a set of significant thresholds $T_S = \left\{T_{S1}, T_{S2}, \ldots, T_{SN_S}\right\}$, where N_S is the number of significant thresholds.

3. Aggregated Population Equalization (APE) Spatial Clustering

Here, we introduce a new spatial clustering algorithm called the Aggregated Population Equalization (APE) to treat imagery data. There are three key components for clustering using the APE: (1) a ranked axis with intervals (RAI), (2) a spatial population, and (3) a set of merging and splitting algorithms. In the following we describe the underlying concept of the APE, the three key components, and the implementation of APE in this application.

3.1. The Underlying Concept of the APE

In a spatial population, each entry is a relationship between a label and another. This relationship could be the average spatial distance, correlation, entropy, conditional probability, and other measures between the two labels. This relationship reflects how a label behaves in the population. A label could populate the image in a centralized, scattered, interspersed, or parasitic manner. For a centralized label, its pixels concentrate at places forming large communities. A pixel at the core of this type of population is usually shielded and has no contact with pixels of other labels. However, a pixel living at the edge of the population has contacts with pixels of other labels. For a scattered label, its pixels populate the image in small, yet noticeable, groups, and these groups are distant from each other. Unlike a centralized label, a pixel within of a scattered population has contacts with pixels of other labels since such a population does not have a strong core. For an interspersed label, it must have another label such that pixels from both labels mix in population; together they are a strong population. In this case, a pixel has contacts with its similarly-labeled pixels and pixels from its counterpart in roughly the same distribution. For a parasitic label, its pixels do not scatter unrestrictedly, but linger along the fringes of another population. These pixels do not form a population, and they actually have more contacts with pixels of other labels then with themselves. Figure 5 depicts different types of population behavior.

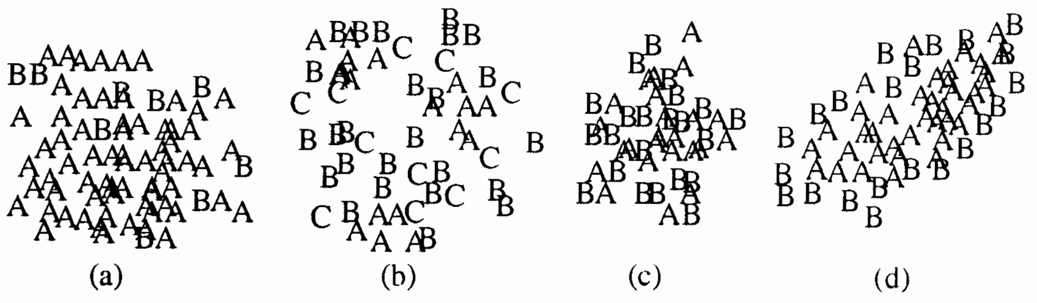

Figure 5. Four types of population behavior. (a) Label A is a centralized population, (b) Label A is scattered among other populations, (c) Labels A and B are interspersed populations, and (d) Label B is a parasitic population occupying the fringes of Label A.

Given the above four types of population behavior, we define several terms in the following.

- A *population* is a label of pixels such that a spatial relationship can be established among its pixels and also with other labels of pixels. The label is an interval on a ranked axis with intervals such that the populations can be ordered along the axis.

- A *population core* is a region of a population that comprises densely of pixels that belong to the population. Note that strictly speaking, every population has a core if we introduce the difference between a weak core and a strong core. A strong population core thus has a high density of within-label pixels; a weak core has a low density.

- An *aggregated population* is a collection of populations.

Given the definitions, we are ready to discuss the underlying concept of the Aggregated Population Equalization. In the space of populations, for members that are not strong, they form alliances and unite to become a strong population. This action is population aggregation, and the product is an aggregated population. On the other hand, a population can be subjected to population disintegration if it is overly dominant or diverse. The result is a group of smaller populations. The Aggregated Population Equalization is the process of obtaining an equilibrium of strong and weak populations such that every aggregated population is similarly strong.

This concept sees some analogies in today's world. For population aggregation, we see that business companies striving to survive or eyeing at greater share of the market either collaborate through joint ventures or mergers. Workers form unions to be able to leverage with their employers. Collegiate basketball teams form conferences to improve competitiveness and revenues. Countries form economic unions or sign trade-related pacts to create bigger business and trade zones. Countries also form military and defense alliances for regional strength and stability. On a smaller scale, people form teams to work on a project, bands to play music, etc. Thus, the behavior of aggregated population is always observed in our society. We can also observe the same behavior in different life forms, animals, plants, bacteria, etc., and such examples are too numerous to count. To list only a few, zebras and giraffes live together to form better defenses against predators. Insects, such as bees and ants, work in groups to build their colonies. Piranhas hunt together so that they can attack preys with size much larger than an individual piranha. On the other hand, we notice that at times a human group, be it a business conglomerate, or a sports team, or a music band, becomes too dominant and disintegrates due to a combination of differences within members of the group and external factors such as the intention of another population in trying to aggregate a member of the group. As a result, a human group, be it academic, artistic, business, cultural, industrial, political, scientific, theological, or social, sometimes becomes too diverse in its operations, opinions, ideologies, practices, or platforms that a division results.

The activity in the population during the Aggregated Population Equalization process depends on the characteristics of the populations, rules of the populations, and the freedom of movements among members of the populations. An aggressive company may launch a hostile takeover to gain control of another company. A cooperative, multi-party political alliance may be able to defeat a strong individual party to gain control of a government. The freedom of movement allows members of a population to switch to another conveniently and maybe even with incentives to do so. To prevent a population from becoming volatile, there are rules—sometimes to slow or restrict the process towards the aforementioned equilibrium—to regulate the movements.

Now we ask the question: why is the concept of the Aggregated Population Equalization useful for spatial clustering? Let us assume that there are N preliminary-segmented classes in the image. However, for these classes to be meaningful and useful, or competitive in our population picture, each should have a strong population. (The strength can be measured in numbers, textures, intensities, or other statistical descriptors.) To do so, some of the N classes form an aggregated class. That aggregated class is what we are looking for in our definition and application of spatial clustering. In our opinion, the spatial relationships within imagery data are important in determining the number of classes and the distribution of each class. They can be used to characterize classes of pixels and gauge the strengths of the classes. With this in mind, we turn to the characteristics of imagery data. Pixels of an aggregated class or population stay together and that is what human observers detect as well. In SAR sea ice analysis, geophysicists observe a patch of roughly homogeneous region more easily than a noisy one. They can also identify a class with higher confidence when the pixels of similar characteristics concentrate at a same place. Thus if there is a population with a weak core, human observers tend to group it to a neighboring population to form a more noticeable aggregated population. If we can capture this process by which sea ice geophysicists analyze sea ice imagery, then we can create meaningful segmentation! This realization of human vision cognition and the observation of the natural behavior of population aggregation and disintegration are the two key ideas behind the concept of the APE.

3.2. Ranked Axis with Intervals

A ranked axis with intervals or RAI is a measuring stick where pixels can be initially labeled in order. Between each label, there is an interval such that there may be more than one members sharing the same label. There is also a precedence in the set of labels such that each pair of labels can be completely ordered. In our case, the set of significant thresholds, T_S, forms the RAI for this implementation. Strictly speaking, the content property of our RAI is intensity, and the intervals are governed by the locations of significant thresholds. To obtain these intervals, we have employed histogram-based algorithms. The MRPD algorithm in this paper can create such intervals. Other existing techniques in 1-D signal processing, and more specifically in histogram processing, can also be used.

Given a ranked axis with intervals, we are able to initially label the image pixels along the axis. This preliminary labeling or segmentation allows the image to be viewed as groups of pixels, reducing noise effects and computational burden. In this application, we have used only one RAI. In general, one may have multiple RAIs. Special care has to be taken to integrate different preliminary segmentations as a result.

The requirement that the axis must be ranked ensures two things: a convenient search implementation for good mergers, and the coherence within merges. One of the difficulties in region merging is to determine which pair of regions to merge. The search for an optimal merger is nonlinearly polynomial. To avoid such combinatorial problem, for a given label, we search only its neighbors for possible merging, and the *neighborhood* is defined by the ranked axis with intervals. So, using T_S as an example, we see that $c(\lambda_j \le p_i < \lambda_k)$ can only be merged with $c(\lambda_{j-1} \le p_m < \lambda_j)$ or $c(\lambda_k \le p_n < \lambda_{k+1})$. The coherence within merges is also achieved by this requirement, i.e., that pixels of non-neighboring labels cannot be merged. Thus, each resulting merge is coherent in terms of a continuum of the axis range. In our case, our axis lies along the intensity: a class that combines $c(\lambda_j \le p_i < \lambda_k)$ and $c(\lambda_k \le p_n < \lambda_{k+1})$ is forbidden. Thus, the within-class intensity is always closed and inclusive.

3.3. Spatial Population

The spatial population captures the spatial relationships among the initially labeled pixels, and it also specifies the neighborhood of each label. This implies that the Cartesian coordinates of the pixels must be utilized. With this implication, our spatial clustering differs from other clustering strategies. We claim that not only are the attributes that describe an entity important, but also the spatial relative location of that entity to others in a space. In normal imagery data analyses, the space is the Cartesian 2-D space.

There are only three requirements for a descriptor used to build the spatial population. First, it should be a function of two labels. Second, the variance of the values of the function of a label and itself should be non-zero. Suppose the descriptor is correlation. Then the function of a label and itself is simply the auto-correlation; and that of a pair of different labels the cross-correlation. Let us say that the descriptor is the average distance between its two elements. Thus, the distance between a label and itself is always zero. As a result, the variance is zero, and this descriptor is not allowed. This second requirement is necessary to ensure activities in the spatial population. Remember that as we characterize a population, we measure its strength and look at its population core. The function of a label and itself reflects the population core. If this value is always a constant, then all populations will be equally strong, and no movements will be observed during the APE. Third, the function should capture the spatial information between the two labels. Here we describe two general approaches: local and global. A local spatial descriptor analyzes each pixel of a population within its neighborhood. With this we introduce locality into the measure, and the spatial information is preserved. For example, suppose we have a *dominance* measure that declares the most dominant class within a neighborhood. For a strong population, this dominance measure will return the label of that population most of the times, and this frequency can be used as an indicator for population strength. A function of a label and itself will thus have a non-zero variance. A function of a pair of different labels will depend on how closely the two classes intersperse. A global spatial descriptor evaluates all pixels of a population within the frame of the image. For example, we can use a more elaborate distance metric than that mentioned above. Suppose the distance metric measures the average distance between each pixel of a label to another pixel of another label. Thus, a function of a label and itself will have a non-zero variance. For a strong population, we have a small distance value. For a weak population—one of those scattered populations for example—the value will be large. Of course, when the neighborhood of a local descriptor gets larger, it approaches globality; and when the working area of a global descriptor is divided into smaller and more manageable regions, it approaches locality. There exist numerous measures such as moments, correlation, and other statistics that can be used to capture spatial information. As for this particular application of the APE, we have chosen to use a local descriptor to build the spatial population.

3.4. Merging and Splitting Algorithms

The merging and splitting algorithms shoulder the task of achieving the equilibrium of the spatial population. The two can work simultaneously or sequentially. In this section, we point out two different approaches for merging and splitting. The first one is *regulatory* in which rules are used to dictate the movements within the populations. So, the decisions for merging and splitting are made by a centralized algorithm. The second approach is *voluntary* in which each population works on its own to achieve strength. Thus, the decisions for merging and splitting are made by each population.

The regulatory approach is the common approach to clustering. We do not discuss it here in detail. However, a few points are worth mentioning. The merging algorithm's objective is population aggregation. Thus, it must know how to measure the strength of each population. Subsequently, it selects the weak populations for aggregation. It must also be able to stop the aggregation process based on some criterion. Usually, an ideal equilibrium is impractical to achieve. Therefore, a relaxed equilibrium should be used. In addition, when merging, only the neighboring labels or populations along the RAI can be merged; and an aggregated population is allowed to merge with neighboring populations of its leader and trailer populations. The splitting algorithm's objective is population disintegration. Thus, it must be able to measure the diversity of each population. For populations with high diversity, the algorithm must know how to divide it into smaller populations. In addition, the algorithm must decide whether to absorb the smaller populations into other populations or let them stand alone.

The voluntary approach has its concept in distributed Artificial Intelligence (DAI). In this case, each population initiates and decides its own movements. This approach assumes that each population has its own drive to achieve strength by looking for cooperations with other populations, and if it becomes too diverse to survive, it disintegrates into two or more populations. In fact, each population is now an agent in a controlled environment. The communications among the agents are restricted to spatial relationships and population strengths. Similarly, a measure for diversity must be designed so that a split is beneficial to the survival of an agent. Again, a provision must be installed to prevent total chaos when a complete equilibrium is impossible to achieve. This approach involves different branches of behavioral sciences and models more closely how the real world (and nature) works: there are certain rules in place, and, at different scales of viewpoint, each population is an agent of its own working to improve itself. Granted that in our spatial clustering, we restrict the options to merging and splitting.

In this application, we have chosen to use the regulatory approach for its ease of implementation. Although the voluntary approach to the APE spatial clustering is an interesting subject and deserves further examinations on multi-agent and distributed Artificial Intelligence, a study of this has not been undertaken in our research.

3.5. Our Implementation of the APE

After the MRPD, we have the set of significant thresholds, $T_S = \left\{T_{S1}, T_{S2}, \ldots, T_{SN_S}\right\}$. Note that the intensity is our ranked axis with intervals and the thresholds are the intervals. Also, since $T_{Si} < T_{S(i+1)}$, the axis is completely ordered, or ranked. With this set of thresholds, we perform regional and pointwise interpolations according to the method outlined in (Haverkamp *et al.* 1995) obtain N_S thresholds for each pixel. Then we segment the image into $N_S + 1$ classes with labels $0 \ldots N_S$. The second component of the APE is the spatial population. Initially, each class is a population, π_i, such that the label of the population,

$$l(\pi_i) = i. \tag{48}$$

Define the spatial descriptor as $SD(l(\pi_i), l(\pi_j))$ or $SD(i,j)$ in our labeling convention. A spatial population, Π_S, is then a matrix of

$$\Pi_S = \begin{bmatrix} SD(0,0) & SD(0,1) & \cdots & SD(0,N_S-1) & SD(0,N_S) \\ SD(1,0) & SD(1,1) & & & SD(1,N_S) \\ \vdots & & \ddots & & \vdots \\ SD(N_S-1,0) & & & & SD(N_S-1,N_S) \\ SD(N_S,0) & & \cdots & & SD(N_S,N_S) \end{bmatrix}. \tag{49}$$

For a commutative function, $SD(i,j) = SD(j,i)$, the matrix is triangular. For a non-commutative function, the whole square matrix is necessary. In our application, we use a local spatial descriptor, using a local neighborhood of a 3×3 window as $\theta(p_i, 1)$ of a pixel p_i. To be consistent, we translate the variety curve in terms of the language of our spatial population:

$$SD(i,j) = V_{T_{S(k-1)}(l-1)} = \frac{\sum_{i=1}^{\#pixels} \sum_{q \in \theta(p_i,1)} \xi(c(p_i), T_{S(k-1)}) * \xi(c(q), l-1)}{\sum_{i=1}^{\#pixels} \xi(c(p_i), T_{S(k-1)})}, \tag{50}$$

where $\xi(a,b) = \{1 \quad if\ a = b; 0 \quad if\ a \neq b\}$, $c(p)$ is the label of pixel p, $i = T_{S(k-1)}$, and $j = l-1$. So the descriptor describes how one label behaves with another label. The function of a label and itself is the probability of that label having itself as a neighbor in a defined neighborhood. The stronger the core of a population, the higher this probability is. Conversely, for a population that has a weak core, the probability of the label having itself as a neighbor is low. For a pair of interspersing populations with labels i and j, $SD(i,j)$ and $SD(j,i)$ might be high. For a parasitic population with a label i at the fringes of another population j, $SD(i,i)$ should be low and $SD(i,j)$ should be high. Note that the descriptor variety is non-commutative.

Now, we are ready for the merging and splitting algorithms. Each aggregated population, $\hat{\pi}_v$, and its member populations

$$\hat{\pi}_v = \{l(\pi_i), l(\pi_{i+1}), \ldots, l(\pi_j)\} \tag{51}$$

can be completely described by the quadruple $\langle S, E, \Sigma, \varepsilon \rangle_{\hat{\pi}_v}$ such that

$$S(\hat{\pi}_v) = l(\pi_i) = i, \tag{52}$$

$$E(\hat{\pi}_v) = l(\pi_j) = j, \tag{53}$$

$$\Sigma(\hat{\pi}_v) = \sum_{i \in \hat{\pi}_v} SD(i,i), \tag{54}$$

$$\varepsilon(\hat{\pi}_v) = \left| SD(l(\pi_{\max}), l(\pi_{\max})) - \Sigma(\hat{\pi}_v) \right|. \tag{55}$$

Equations 52 and 53 denote the starting and ending population labels, respectively, of an aggregated population. Equation 54 describes combined individual strengths of the aggregated population. Our strategy of merging is to first locate the most dominant population, $\pi_{\max}$. To measure the strength of a population, we use the reflexive function value, $SD(i,i)$. Formally,

$$\pi_{\max} = \pi_i | SD(i,i) \geq SD(j,j) \quad j = 0 \ldots N_S. \tag{56}$$

Having located $\pi_{\max}$, we search for aggregated populations that match the strength of the most dominant population. The difference in strength between $\pi_{\max}$ and an aggregated population $\hat{\pi}_v$ is the error of the aggregated population, described by Equation 55.

To relax the ideal equilibrium requirement, we use a Least Over-Commitment Strategy (LOCS). This strategy is borrowed from the planning research in Artificial Intelligence (Rich and Knight 1991). In planning, to avoid combinatorial issues, a commitment can be partially satisfied to generate incomplete solutions for a nearly decomposable problem. Then, these solutions are repaired to improve the final results. The commitment in our case is to produce an aggregated population with a combined individual strength, $\Sigma(\hat{\pi}_v)$, equal to $SD(l(\pi_{\max}), l(\pi_{\max}))$, or $\varepsilon(\hat{\pi}_v) = 0$. Suppose during a current merging process that we have a set of populations $\hat{\pi}_v = \{l(\pi_i), l(\pi_{i+1}), \ldots, l(\pi_j)\}$, and $\Sigma(\hat{\pi}_v) < SD(l(\pi_{\max}), l(\pi_{\max}))$. Under the Least Over-Commitment Strategy, if the inclusion of π_{j+1} into $\hat{\pi}_v$ yields a combined individual strength that is greater than $SD(l(\pi_{\max}), l(\pi_{\max}))$, then it should be the last member of the aggregated population, $E(\hat{\pi}_v) = l(\pi_{j+1}) = j + 1$. This relaxation enables us to design a fast merging algorithm; it also avoids the combinatorial problem of equilibrium optimization.

3.1. Generate Clusters of Aggregated Populations—Merging

To generate a cluster of aggregated population, we move along the ranked axis with intervals in two directions—top-down and bottom-up, creating two clusters $C_\downarrow$ and $C_\uparrow$. Note that $C_\downarrow = \left\{\hat{\pi}_0^\downarrow, \hat{\pi}_1^\downarrow, \ldots, \hat{\pi}_{N_{C\downarrow}-1}^\downarrow\right\}$, and $C_\uparrow = \left\{\hat{\pi}_0^\uparrow, \hat{\pi}_1^\uparrow, \ldots, \hat{\pi}_{N_{C\uparrow}-1}^\uparrow\right\}$, where $N_{C_\downarrow}$ and $N_{C_\uparrow}$ are the number of aggregated populations in the two clusters, respectively. To generate $C_\downarrow$:

1. Move from the top of the ranked axis to the bottom ($0 \rightarrow N_S$).
   A. If the next population is $\pi_{\max}$, then the current aggregated population is complete and initiate a new aggregated population.
   B. If the current population is $\pi_{\max}$, then make it a single-member aggregated population and initiate a new aggregated population.
   C. If the current $\Sigma(\hat{\pi}_v)$ so far has exceeded or equaled to $SD(l(\pi_{\max}), l(\pi_{\max}))$, then the current aggregated population is complete and initiate a new aggregated population.
   D. If the current population is π_{N_S}, then the current aggregated population is complete, and the cluster is also complete.
2. For each completed aggregated population, $\hat{\pi}_v$, compute its quadruplet $\langle S, E, \Sigma, \varepsilon \rangle_{\hat{\pi}_v}$.

A similar algorithm can easily be derived from the above to generate $C_\uparrow(\Pi_S)$, by counting from the bottom to the top of the axis ($N_S \rightarrow 0$), and replacing π_{N_S} in Step 1.D with π_0. From this algorithm, we observe the following. First, $\pi_{\max}$ is made an aggregated population with itself as the only member of the aggregation. This concurs with our APE concept: since it is already the strongest population, it has no drive to improve itself. However, as we shall see later, it is possible for $\pi_{\max}$ to recruit or be recruited by another aggregated population to improve local population coherence. Second, there is no guarantee that every aggregated population will have $\varepsilon(\hat{\pi}_v) \geq 0$. The aggregated populations completed under conditions 1.A and 1.D usually fail to meet the commitment. The degree of the failure will be relieved a little during the refinement process. In general, this merging algorithm is fast and guided by the ranked axis and the commitment criterion. It eliminates optimization issues that accompany any merging algorithm. Now we have to decide which cluster to use, and refine the selected cluster.

3.2. *Select the Best Cluster*

If the two clusters, $C_{\downarrow}$ and $C_{\uparrow}$, are exactly the same, then the merging process is completed and no further refinements or adjustments are needed. However, if they are not, we evaluate the error of each cluster:

$$\varepsilon\left(C_{\downarrow}\right)=\sum_{i=0}^{N_{c\downarrow}-1}\varepsilon\left(\widehat{\pi}_{i}^{\downarrow}\right), \tag{57}$$

and

$$\varepsilon\left(C_{\uparrow}\right)=\sum_{i=0}^{N_{c\uparrow}-1}\varepsilon\left(\widehat{\pi}_{i}^{\uparrow}\right). \tag{58}$$

We select the cluster whose error is the smaller and refine it at a later stage. If the errors are the same, then we use a tie-breaker system:

1. Compute the maximum individual errors,

$$\varepsilon_{\max}^{\downarrow}=\varepsilon\left(\widehat{\pi}_{v}^{\downarrow}\right)\middle|\varepsilon\left(\widehat{\pi}_{v}^{\downarrow}\right)\geq\varepsilon\left(\widehat{\pi}_{u}^{\downarrow}\right)\quad u=0\ldots N_{C_{\downarrow}}, \tag{59}$$

and

$$\varepsilon_{\max}^{\uparrow}=\varepsilon\left(\widehat{\pi}_{v}^{\uparrow}\right)\middle|\varepsilon\left(\widehat{\pi}_{v}^{\uparrow}\right)\geq\varepsilon\left(\widehat{\pi}_{u}^{\uparrow}\right)\quad u=0\ldots N_{C_{\uparrow}}. \tag{60}$$

2. Compare the two errors. If the two errors are different, then the best cluster is the one with the smaller maximum individual errors, and we exit from the tie-breaker system.
3. Compute the maximum individual errors from the reduced sets $\left\{\widehat{\pi}_{u}^{\downarrow}\right\}/\left\{\widehat{\pi}_{v}^{\downarrow}\middle|\varepsilon\left(\widehat{\pi}_{v}^{\downarrow}\right)=\varepsilon_{\max}^{\downarrow}\right\}$ and $\left\{\widehat{\pi}_{u}^{\uparrow}\right\}/\left\{\widehat{\pi}_{v}^{\uparrow}\middle|\varepsilon\left(\widehat{\pi}_{v}^{\uparrow}\right)=\varepsilon_{\max}^{\uparrow}\right\}$, respectively. Designate the new errors as $\varepsilon_{\max}^{\downarrow}$ and $\varepsilon_{\max}^{\uparrow}$. If both errors are defined, repeat step 2. If one of the errors is undefined, that means one of the clusters has a greater number of aggregated populations. The cluster with the less number of aggregated populations is the best cluster and we exit the tie-breaker system.

So, after this stage, we will have selected the best cluster. If it turns out that the two clusters are exactly the same, then we assume that the configuration of aggregated populations is optimal and perform no refinements. Otherwise, the selected cluster has to be refined.

3.3. *Refine the Selected Cluster*

As in planning, this refinement stage is to repair the partial solutions (aggregated populations) to satisfy the commitments better. There are three refinement operations in our application: (1) migrating border populations, (2) solidifying the dominant population, and (3) absorbing insignificant populations. The first operation is always performed while the other two are only carried out when certain conditions are met. These three operations are performed sequentially. Note that in the following discussions, the cluster referred to is always the selected cluster unless specified otherwise; so the superscript that denotes the direction of the merging process is discarded.

A. Migrating Border Populations

This operation is used to reduce the overall amount of errors by locally migrating one population from an aggregated population to another. This operation is applied to all aggregated populations except $\pi_{\max}$.

For an aggregated population $\hat{\pi}_v$, if $v \neq N_C - 1$ and $\pi_{E(\pi_v)+1} \neq \pi_{\max}$ then we have the following assignments:

$$\begin{aligned} S'(\hat{\pi}_v) &= S(\hat{\pi}_v) \\ E'(\hat{\pi}_v) &= E(\hat{\pi}_v) + 1 = S(\hat{\pi}_{v+1}) \\ S'(\hat{\pi}_{v+1}) &= S(\hat{\pi}_{v+1}) + 1 \\ E'(\hat{\pi}_{v+1}) &= E(\hat{\pi}_{v+1}) \end{aligned} \quad . \tag{61}$$

In effect, we are trying to migrate $S(\hat{\pi}_{v+1})$ into $\hat{\pi}_v$. If

$$\varepsilon'(\hat{\pi}_v) + \varepsilon'(\hat{\pi}_{v+1}) < \varepsilon(\hat{\pi}_v) + \varepsilon(\hat{\pi}_{v+1}), \tag{62}$$

then we confirm and execute the migration. On the other hand, given an aggregated population $\hat{\pi}_v$, if $v \neq 0$ and $\pi_{S(\hat{\pi}_v)-1} \neq \pi_{\max}$, then we have the following assignments:

$$\begin{aligned} S'(\hat{\pi}_{v-1}) &= S(\hat{\pi}_{v-1}) \\ E'(\hat{\pi}_{v-1}) &= E(\hat{\pi}_{v-1}) - 1 \\ S'(\hat{\pi}_v) &= S(\hat{\pi}_v) - 1 = E(\hat{\pi}_{v-1}) \\ E'(\hat{\pi}_v) &= E(\hat{\pi}_v) \end{aligned} \tag{63}$$

In effect, we are trying to migrate $E(\hat{\pi}_{v-1})$ into $\hat{\pi}_v$. Similarly, if

$$\varepsilon'(\hat{\pi}_v) + \varepsilon'(\hat{\pi}_{v-1}) < \varepsilon(\hat{\pi}_v) + \varepsilon(\hat{\pi}_{v-1}), \tag{64}$$

then we confirm and execute the migration. For $\hat{\pi}_0$, we apply Equations 61 and 62; for $\hat{\pi}_{N_C-1}$, Equations 63 and 64; for other aggregated populations, we apply all four equations.

B. Solidifying the Dominant Population

It is possible to have a tiny, highly concentrated patch of pixels that ends up as the $\pi_{\max}$. Conceptually, such a population, although extremely strong, needs to absorb other populations to solidify its dominance to survive. Thus, for this operation, we have two tasks. First we have to decide whether $\pi_{\max}$ needs strengthening. During the generation of the spatial descriptor variety, we tally the number of pixels that belong to each population, such that

$$N_{\pi_i} = \# p_j \middle| l(p_j) = l(\pi_i) \quad p_j \in I. \tag{65}$$

Define the number of pixels of the largest population as $\max(N_{\pi_i})$. A $\pi_{\max}$ needs strengthening if

$$N_{\pi_{\max}} < \zeta \max(N_{\pi_i}). \tag{66}$$

The parameter ζ is set at 1% or 0.01. So, in other words, if the strongest population in the spatial population has at least 1% (in terms of numbers) of the largest population in the spatial population, according to our implementation, then the strongest population survives[2]. Otherwise, we solidify it by migrating it into one of its neighboring aggregated populations. The decision is based on the spatial descriptor. Suppose the population ranked before $\pi_{\max}$ is π_{prev} and that ranked after is π_{next}. If

$$SD\big(l(\pi_{\max}),l(\pi_{prev})\big) > SD\big(l(\pi_{\max}),l(\pi_{next})\big), \tag{67}$$

then π_{next} is migrated into the aggregated population with $E(\hat{\pi}_v) = l(\pi_{prev})$. Otherwise, π_{next} is migrated into the aggregated population with $E(\hat{\pi}_v) = l(\pi_{prev})$. As a result, π_{next} has been solidified. It is possible that this merger might introduce a super-strong aggregated population. But there are two things that may offset that super-strong status. First, for this merger to take place, π_{next} must have a small population. So the merger in effect does not produce visually super-strong segmentation classes on a spatial imagery data. Second, note that during our merging process, we always complete an aggregated population if the next population is π_{next}. Often, this results in a population with $\Sigma(\hat{\pi}_v) < SD\big(l(\pi_{\max}),l(\pi_{\max})\big)$, and, thus, again, the merger does not actually produce a super-strong aggregated population. From a reversed angle, we forbid merging of $\pi_{\max}$ during the merging process because if $\pi_{\max}$ has a large population, we do not want any merger with $\pi_{\max}$ to become a super-strong aggregated population, and if $\pi_{\max}$ has a large population, we do not want to contaminate the population with other weak but substantially large populations. As we shall see that during the final stage of refinement, we do allow such contamination from a weak and small population.

C. *Absorbing Insignificant Populations*

If an aggregated population, $\hat{\pi}_v$, satisfies the following conditions:

$$\begin{aligned} &S(\hat{\pi}_v) = E(\hat{\pi}_v) \\ &\pi_i \in \hat{\pi}_v \\ &l(\pi_i) \neq l(\pi_{\max}) \\ &N_{\pi_i} < \zeta \max(N_{\pi_i}) \end{aligned}, \tag{68}$$

[2] This is based on our experience on working with SAR sea ice imagery. For other real world applications, the percentage might vary. Indeed, an extremely dominant population might be less than 1% of the largest population yet it is able to maintain an equilibrium in some environment.

then it is insignificant. Since it has only one member population and is not the $\pi_{\max}$, that implies that it is also a weak population. So, we absorb it into its neighboring aggregated population. There are three possible scenarios. First, if $v = 0$, then it is the remnant of a bottom-up merging process, and it is absorbed into $\hat{\pi}_{v+1}$. The number of aggregated populations and indices within the cluster are adjusted accordingly. Second, if $v = N_C - 1$, then it is the remnant of a top-down merging process, and it is absorbed into $\hat{\pi}_{v-1}$. Third, if it does not match the first two scenarios, then it must be the remnant of a pre-maturely completed population when coming to a neighboring population which is $\pi_{\max}$. So, it is absorbed into $\pi_{\max}$. This is the contamination that we have mentioned above. In our modeling of the real world, we see this absorption of insignificant population into a stronger population as a natural tendency of seeking protection and assistance for growth and survival. In image segmentation, such an insignificant and weak class is actually negligible in terms of human vision capability because the number of pixels belonging to that class is extremely small, compared to the largest population, and the pixels do not form concentrated regions. On the other hand, such an absorption improves visual interpretation of the image because the pixels of the absorbed class enhance the compactness of the absorbing population.

3.4. Disintegrate Diverse Populations—Splitting

To measure diversity within an aggregated population, we first perform the necessary mergings, as a result of above processes, to the image. This is done by re-labeling the pixels of the preliminary segmented image, generated by the dynamic local thresholding module. After merging, we compute the diversity of each aggregated population. In what follows, we first discuss what constitutes a diverse population. Then, we use a non-deterministic assignment strategy to re-label pixels.

A. Diversity Measure

Our diversity measure is based on a local neighborhood of a pixel, $\theta(p_i, 1)$, same as that of the spatial descriptor used in building the spatial population. Supposed a member of the neighborhood is $q \in \theta(p_i, 1)$. We define

$$o(p_i, j) = \#(l(q) = j) \tag{69}$$

such that the number of pixels with the label j within the neighborhood of a pixel p_i is $o(p_i, j)$. From this definition, we can extend it to accumulate the results of all pixels of a same label:

$$H_{i,j,k} = \#\left(o\left(p_n, j\right) = k\right) \quad l\left(p_n\right) = i, p_n \in I. \tag{70}$$

Equation 74 provides us a count of the frequency of all pixels, with a label i, having j as a neighbor k times. Then we compute the probability

$$P_{i,j,k} = \frac{H_{i,j,k}}{\#\left(l\left(p_n\right) = i\right)} \quad p_n \in I. \tag{71}$$

This probability tells us the behavior of the pixels within a label. Our concept behind these definitions is as follows. Suppose there is diversity within a label (or a population), i, such that there are member pixels with close contacts with pixels of another label, j. Then, $P_{i,j,k}$ will be high for at least one value of $k \geq 4$. (We use 4 since it is half of the number of neighboring pixels in a $\theta\left(p_i, 1\right)$ neighborhood.) That indicates that some members of the label i are interspersed with the members of the label j. If there exists such a label, then we perform population disintegration on it by splitting it into two. For a strong population, the regional cores will indicate high $P_{i,i,k}, k \geq 4$. At the edge of these regions, we will have an increase probability of $P_{i,j,k}$. However, the probability is negligible. Hence, no disintegration is carried out. In the modeling of real world, this signifies a very strong, highly concentrated organization.

In this application, we use a threshold called *diversity threshold*, $T_{diversity}$, such that, formally, if a population with the label i has a probability

$$P_{i,j,k} > T_{diversity} \quad j \neq i, k \geq \frac{\left|\theta\left(p_i, 1\right)\right|}{2} \tag{72}$$

then the population is split. The constraint $j \neq i$ is in place to prevent splitting a strong core since as described above, $P_{i,i,k}, k \geq 4$ is normally high. The constraint $k \geq \left|\theta\left(p_i, 1\right)\right|/2$, or $k \geq 4$ in our case, is in place to ensure only high-contact relationships between two labels are analyzed. Note that $P_{i,j,k}$ for $k < 4$ is usually high. The value of $T_{diversity}$ was obtained experimentally. It is now set at 0.17.

B. Non-Deterministic Disintegration

After the identification of a diverse population, we split it into two via a non-deterministic approach. Again, we look at the local neighborhood $\theta(p_i,1)$ to compute $o(p_i,l(p_i)) = \#(l(q)=l(p_i))$ for $l(p_i)$ equal to the label of the diverse population. Then a random number is generated in the range of $0\ldots|\theta(p_i,1)|$. If the random number is greater than $o(p_i,l(p_i))$, then the pixel is flagged for re-labeling. Thus, if a pixel is highly surrounded by its "compatriots", then $o(p_i,l(p_i))$ is high, and the probability of a random number greater than this number is low—indicating that the probability of such a pixel being flagged is low. This is the non-deterministic part of the disintegration process. After all pixels have been examined, the flagged pixels are given a new label. Another alternative is to assign the pixels the label of the next-in-rank population, effectively migrating them into the next population. Our current implementation adopts the former strategy.

4. Conclusion

In this application, we have achieved several objectives. First, we have designed an unsupervised image segmentation technique, combining image processing and spatial clustering, for SAR sea ice image analysis. As a secondary achievement, we have shown that the integration of image processing and machine learning methodologies is practical and viable in unsupervised segmentation. Second, we have improved our dynamic local thresholding in terms of automation and handling of various images. Third, we have designed a peak detection method called the Multiresolution Peak Detection (MRPD) that automatically detects peaks in a histogram. This method can also be viewed as an image quantizer and enhancer. Fourth, we have created an innovative spatial clustering concept, called the Aggregated Population Equalization (APE). This concept is based on a modeling of the real world behavior of weak and strong populations within certain environments. We have provided the three key components of the APE and how one may go about computing them. We have subsequently implemented the APE spatial clustering using a Least Over-Commitment Strategy and aggregation and disintegration procedures. To conclude, we have successfully implemented an unsupervised image segmentation technique. We call this technique ASIS for *A*utomated *S*ea *I*ce *S*egmentation.

Bibliography

Atiquzzaman, M. (1992). Multiresolution Hough Transform—An Efficient Method of Detecting Patterns in Images, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **14**(11), pp. 1090–1095.

Bergholm, F. (1987). Edge Focusing, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **9**(6), pp. 726–741.

Boukharouba, S., J. M. Rebordao, and P. L. Wendel (1985). An Amplitude Segmentation Method Based on the Distribution Function of an Image, *Computer Vision, Graphics, and Image Processing*, **29**, pp. 47–59.

Eklundh, J.-O., T. Elfving, S. Nyberg (1982). Edge Detection Using the Marr-Hildreth Operator with Different Sizes, in *Proceedings of 6th IPCR*, Munich, pp. 1109–1112.

Gauch, J. M. and S. M. Pizer (1993). Multiresolution Analysis of Ridges and Valleys in Grey-Scale Images, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **15**(6), pp. 635–646.

Goudail, F., E. Lange, T. Iwamoto, K. Kyoma, and N. Otsu (1996). Face Recognition System Using Local Autocorrelations and Multiscale Integration, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **18**(10), pp. 1024–1028.

Gunsel, B., A. K. Jain, and E. Panyirai (1996). Reconstruction and Boundary Detection of Range and Intensity Images Using Multiscale MRF Representations, *Computer Vision and Image Understanding*, **63**(2), pp. 353–366.

Haverkamp, D., L.-K. Soh, and C. Tsatsoulis (1995). A Comprehensive, Automated Approach to Determining Sea Ice Thickness from SAR Data, *IEEE Transactions on Geoscience and Remote Sensing*, **33**(1), pp. 46–57.

Heitz, F., P. Perez, and P. Bouthemy (1994). Multiscale Minimization of Global Energy Functions in some Visual Recovery Problems, *CVGIP: Image Understanding*, **59**(1), pp. 125–134.

Horowitz, S. L. (1977). Peak Recognition in Waveforms, in *Waveforms in Syntactic Pattern Recognition, Applications*, Fu, K. S. (ed.), Berlin: Springer-Verlag.

Hummel, R. A., B. Kimia, and S. W. Zucker (1987). Deblurring Gaussian Blur, *Computer Vision, Graphics, and Image Processing*, **38**, pp. 66–80.

Jackway, P. T. and M. Deriche (1996). Scale-Space Projection of the Multiscale Morphological Dilation-Erosion, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **18**(1), pp. 38–51.

Kakarala, R. and A. O. Hero (1992). On Achievable Accuracy in Edge Localization, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **14**(7), pp. 777–781.

Koplowitz, J. and J. DeLeone (1996). Hierarchical Representation of Chain-Encoded Binary Image Contours, *Computer Vision and Image Understanding*, **63**(2), pp. 344–352.

Lu, Y. and R. C. Jain (1992). Reasoning about Edges in Scale Space, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **14**(4), pp. 450–468.

Mallat, S. G. and S.Zhong (1992). Characterization of Signals for Multiscale Edges, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **14**(7), pp. 710–732.

Reissell, L.-M. (1996). Wavelet Multiresolution Representation of Curves and Surfaces, *CVGIP: Graphical Models and Image Processing*, **58**(3), pp. 198–217.

Rich, E. and K. Knight (1991). *Artificial Intelligence*, 2nd. Ed., New York: McGraw-Hill.

Rosenfeld, A. and P. D. Torre (1983). Histogram Concavity Analysis as an Aid in Threshold Selection, *IEEE Transactions on Systems, Man, and Cybernetics*, **13**(3), pp. 231–235.

Rosin, P. L. and G. A. W. West (1995). Salience Distance Transforms, *CVGIP: Graphical Models and Image Processing*, **57**(6), pp. 483–521.

Sezan, M. I. (1990). A Peak Detection Algorithm and Its Application to Histogram-Based Image Data Reduction, *Computer Vision, Graphics, and Image Processing*, **49**(1), pp. 36–51.

Sjoberg, F. and F. Bergholm (1988). Extraction of Diffuse Edge Focusing, *Pattern Recognition Letters*, **7**(3), pp. 181–190.

Whitaker, R. T. and S. M. Pizer (1993). A Multi-Scale Approach to Nonuniform Diffusion, *CVGIP: Image Understanding*, **57**(1), pp. 99–110.

Whitten, G. (1993). Scale Space Tracking and Deformable Sheet Models for Computational Vision, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **15**(7), pp. 697–706.

Witkin, A. P. (1984). Scale Space Filtering: A New Approach to Multi-Scale Description, in *Image Understanding*, S. Ullman and W. Richards (eds.), pp. 79–95, NJ: Ablex Publishing.

Yaou, M. H. and W. T. Chang (1994). Fast Surface Interpolation Using Multiresolution Wavelet Transform, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **16**(7), pp. 673–688.