Automated Image Segmentation for Synthetic Aperture Radar Feature Extraction

Julie Ann Jackson
Dept. of Electrical and Computer Engineering
Air Force Institute of Technology
Dayton, Ohio, 45433
Email: julie.jackson@afit.edu

Abstract—Automated segmentation routines may be used to extract scattering features in synthetic aperture radar (SAR) images. The watershed transform segments real-valued images into regions associated with local minima. Watershed algorithms suffer from over-segmentation which, for SAR image segmentation, results in many more regions than scatterers. We consider an algorithm called Peak Region Segmentation (PRS). PRS is an inverted version of the watershed transform that seeks to group pixel regions associated with local maxima. We implement the algorithm to segment one, two, and three-dimensional images. We extend PRS to include region merging to avoid over-segmentation. Threshold settings allow the user to strike a balance between region merging and separation of closely-spaced scatterers. Image segmentation examples are shown for 1D, 2D, and 3D SAR images.

Index Terms—image segmentation, watershed transform, synthetic aperture radar

I. INTRODUCTION

The watershed transform is a technique for segmenting a real-valued (gray-scale) image into regions associated with a common local minima. We implement an inverted version of the watershed transform in order to segment synthetic aperture radar (SAR) images into high-energy regions. The proposed segmentation is automated and may be used to extract image features for further analysis or suppression. The algorithm appears in [1]; additional image segmentation examples are shown in this paper.

II. WATERSHED TRANSFORM

Landscape examples are commonly used to explain the concept of the watershed transform for two-dimensional images. Let the magnitude of the image pixels represent the topology of the land. The valleys where rainwater collects are called “catchment basins”. Each catchment basin consists of all of the image pixels whose rainwater flows into that valley. The mountain ridge-lines that separate the catchment basins are the “watershed lines” or “watersheds”. For example, the Continental Divide is a watershed line. Rain that falls west of the Continental Divide flows into the Pacific Ocean; rain that falls east of the divide flows into the Atlantic Ocean.

Depending on the choice of algorithm, the watershed transform finds either the watershed lines or labels the pixels according to catchment basin. We will define the catchment basins and label the pixels. The watershed transform may be extended to higher dimensions.

The watershed transform is developed in the framework of mathematical morphology. We refer the reader to [2]–[4] for a formal mathematical treatment. There are continuous and discrete definitions of the watershed transform, and many algorithmic definitions and variations exist [5]–[7]. We implement and extend an algorithm proposed by Stach and LeBaron [8].

III. PEAK REGION SEGMENTATION

The “Peak Region Segmentation” (PRS) method in [8] is a type of watershed algorithm. In this case, the landscape topology is inverted so that the catchment basins correspond to the high-energy regions of the image. The PRS algorithm is proposed for use in segmenting and removing unwanted radar returns from target supports in compact range measurements. The method is adopted in [13] to segment radar target scattering centers in two dimensions. We extend the algorithm to segment three-dimensional SAR images and define thresholds to balance region merging with scatterer separation.

The PRS algorithm begins by sorting image pixel magnitudes from high to low. Then, each sorted pixel is considered along with its neighbors to determine the appropriate region.
label. If the pixel under test has no labeled neighbors, it is assigned a new label. If all of its neighbors have the same label, the pixel is assigned to the same region as its neighbors. If the pixel under test has labeled neighbors that do not all belong to the same region, it is assigned to the neighbor’s region that contains the largest-magnitude pixel. Labeling of regions continues for all pixels, \( i = 1, \ldots, N \). Note, a threshold (shown as \( \tau_1 \) in Figures 2-3) may be applied to the image so labeling stops at a given magnitude, e.g. the expected noise level. All unlabeled pixels are given the label zero. The algorithm steps are summarized in Algorithm III.1.

As with other implementations of the watershed transform, Algorithm III.1 must be modified to handle over-segmentation due to noise ripple. For a \( d \)-dimensional image, the PRS algorithm defines a pixel’s neighbors to be the surrounding \((2p + 1)^d - 1\) pixels, where \( p \) is a user-defined connectivity parameter that defines how many samples from the center pixel are considered neighbors. See Figure 1 for an example.

Typically, \( p = 1 \) in Algorithm III.1, but the authors of [8] suggest increasing the value of \( p \) to account for noise ripple. For the case of extended scatterers, increasing \( p \) does correctly group more pixels together. However, simply increasing \( p \) does not account for the case of closely-spaced scatterers. We propose another method for handling noise ripple.

We define two regions as adjacent if any pixel from one region is also a neighbor of any pixel from the other region. For extended scatterers, we expect that noise ripple in the response will result in pixels in adjacent regions whose amplitudes are within some threshold \( \tau_1 \) of the peak pixel magnitude in the regions. For closely-spaced scatterers, we expect that the pixel magnitudes will have two closely-spaced peaks with a dip in pixel magnitude below the \( \tau_1 \) threshold at some point between the two scatterer locations. Therefore, we merge adjacent regions if no pixel with value below the \( \tau_1 \) threshold is closer to the adjacent region than the pixels within the threshold. The user must set \( \tau_1 \) to be large enough to account for noise ripple but small enough to distinguish peaks of neighboring scatterers. This region merging method is summarized in Algorithm III.2.

In addition to region merging, we modify Algorithm III.1 to provide additional user control in segmenting high-energy regions which, for our application, correspond to regions in a scene. For each region, we wish to segment mainlobe scattering only. Thus, we keep only those pixels within a user-defined threshold \( \tau_2 \) dB of the region peak. In addition, the image is clipped at a minimum pixel magnitude threshold \( \tau_3 \) to eliminate pixels outside the desired dynamic range. After all regions are labeled, merged, and cropped, we perform renormalizing for bookkeeping purposes. We renumber the region labels to remove gaps in numbering left by the region merging process. Also, we renumber the regions starting with the largest-energy region and limit the number of regions to a user-defined maximum. Note, instead of specifying a maximum number of regions, one may set a threshold on the desired amount of energy contained in the regions. Regions with labels above the maximum number (or whose energy accumulates the total segmented energy above the desired threshold) are removed. These additional region segmentation modifications are also summarized in Algorithm III.2.

![Fig. 1. The shaded pixel’s neighbors in a 2D image defined by connectivity parameter \( p \)](Image 362x611 to 433x681)

\[
P_i \leftarrow \text{Sorted Pixels Magnitudes from High to Low (cut off at } \tau_3)\]
\[
l \leftarrow 0 \{\text{Initialize Region Counter}\}\]
\[
\text{for } i = 1 \text{ to } N \text{ do} \{\text{Assign Region Labels } L_i \text{ to each Pixel}\}\]
\[
\quad \text{Consider Neighbors of Pixel } i \text{ that are } p \text{ pixels away in each direction}\]
\[
\quad \text{if Pixel } i \text{ has no labeled neighbors then}\]
\[
\quad \quad l \leftarrow l + 1\]
\[
\quad \quad \text{Assign pixel } i \text{ a new region label } L_i = l\]
\[
\quad \text{else if ALL labeled neighbors have label } L_i \text{ then}\]
\[
\quad \quad \text{Assign pixel } i \text{ to the same region: } L_i \leftarrow L_i\]
\[
\quad \text{else } \{\text{Labeled Neighbors have different labels}\}\]
\[
\quad \quad \text{Assign pixel } i \text{ to have same label as largest-magnitude neighbor}\]
\[
\text{end if}\]
\[
\text{end for}\]

\textbf{Algorithm III.1: Peak Region Segmentation Algorithm}

IV. IMAGE SEGMENTATION EXAMPLES

In this section we present example region segmentation results. Results for the original PRS (Algorithm III.1) and the extended version (Algorithm III.2) with region merging are shown.

In Figure 2, we demonstrate the region segmentation algorithms for a one-dimensional signal. The 1D signal example represents three extended scatterers in noise, with peak signal to noise ratio SNR = 30dB. Figure 2(a) shows the original signal, colored by the segmentation results of Algorithm III.1. The final segmentation results from Algorithm III.2 are shown in 2(b). We see that Algorithm III.1 is susceptible to local fluctuations and over-segments the signal. The region merging operations in Algorithm III.2 correctly account for the noise ripple in the extended scatterers while still separating the closely spaced scatterers. Although we set a maximum number of five regions to segment, we see that almost all of the signal energy is contained in the first three regions, corresponding to the true number of scatterers.

In Figure 3, we show the region segmentation performance for a one-dimensional signal containing localized scatterers.
Do Algorithm III.1
{Merge adjacent regions whose nearest pixels have amplitudes within $\tau_1$}
\( r = 1 \) Initialize region counter
while \( r \leq \text{max number regions} \) do
  \( \text{repeat} \)
  \( \text{threshold} \leftarrow (\text{peak pixel magnitude in region } r) - \tau_1 \)
  for \( m = 1 \) to \# regions adjacent to region \( r \) do
    \( \text{dist} \leftarrow \text{sorted distance of pixels in region } m \) to peak pixel in region \( r \)
    \( \text{mag} \leftarrow \text{corresponding magnitude of pixels in region } m \)
    if for \( i < j, \text{mag}(i) < \text{threshold} \) and \( \text{mag}(j) > \text{threshold} \) then
      Do NOT merge regions
    else
      Re-assign region \( m \) pixels to region \( r \)
    end if
  \end for
  \{Clip pixels more than \( \tau_2 \) below region peaks\}
  for \( i = 1 \) to \# pixels in region \( r \) do
    if Amplitude of pixel \( i < (\text{Region Peak} - \tau_2) \) then
      Re-assign pixel \( i \) to region \( 0 \)
    end if
  \end for
until No more regions adjacent to region \( r \)
increment region counter: \( r \leftarrow r + 1 \)
\end while
Re-order numbering of regions, from largest to smallest energy contribution
Re-assign regions with region numbers > \max \# of desired segments to region \( 0 \)

**Algorithm III.2:** Extensions to the Peak Region Segmentation in Algorithm III.1.

As in Figure 2, the sub-figures correspond to (a) the Algorithm III.1 segmentation and (b) the Algorithm III.2 segmentation results. We note that the value of threshold $\tau_1$ dictates whether or not the original Segment 2 and Segment 3 are merged. According to the region merging criteria, if the dip in scattering between the two peaks falls below the threshold cutoff determined by $\tau_1$, the peaks are not merged. That is, the two peaks are associated with different scatterers. For the example in Figure 3, $\tau_1 = -1$ dB, the dip falls below the threshold, and the two scatterers are separated. For $\tau_1 = -2$ dB however, the two scatterers would be merged. Thus, the user-defined thresholds in the modified PRS in Algorithm III.2 do account for noise ripple while allowing for the case of closely-spaced scatterers.

Figure 4 shows example segmentation results for a 2D SAR image of simulated radar returns for a construction backhoe. The SAR aperture is a wide-angle, single pass with azimuth $\phi \in [66^\circ, 114^\circ]$ at 1/14th degree increments and elevation angle $\theta = 30^\circ$. Algorithm III.1 locates 147 regions of high energy in the image. Algorithm III.2 merges these regions to group extended scattering responses—see for example, the extended scatterer at the bottom of the image—while still separating closely-spaced scatterers. The resulting number of regions is 85; only the 50 regions corresponding to largest magnitude responses (98.3% of total energy) are shown in Figure 4(c).

Finally, Figure 5 shows example segmentation results for a 3D SAR image of seven canonical shape scatterers. The SAR imaging aperture is a sparse aperture in 3D, resulting in high sidelobes and aliasing with traditional backprojection or Fourier-based imaging [1], [14]. Automated image segmentation will not work on such blurred images. Recently, $\ell_1$-regularized image formation has generated sparse 3D images in which the sidelobes and aliasing are removed or greatly reduced [15], [16]. Image segmentation may be successfully applied to these sparse images, as shown in Figure 5.
Fig. 3. Image segmentation results for Algorithms III.1 and III.2 applied to simulated, closely-spaced localized scatterers in one dimension. Algorithm settings: SNR = 30 dB, $p = 1$, $\tau_1 = -1$ dB, $\tau_2 = -10$ dB, $\tau_3 = -30$ dB, maximum of 5 segments.

Fig. 4. Image segmentation results for Algorithms III.1 and III.2 applied to a 2D SAR image of a construction backhoe with SNR=40 dB. Algorithm settings are SNR = 40 dB, $p = 1$, $\tau_1 = -4$ dB, $\tau_2 = -15$ dB, $\tau_3 = -25$ dB, and a maximum of 50 image segments.
V. Conclusion

Automated SAR image segmentation is necessary for feature extraction. We have demonstrated the watershed method of PRS for 1D, 2D, and 3D scenarios. We have presented user-defined thresholds for region-merging to overcome the over-segmentation that occurs with PRS. The proposed region-merging method attempts to associate regions of extended scattering while separating closely-spaced scatterers. The resulting image segmentation may be used to identify regions for scattering center extraction or other feature extraction methods.

REFERENCES


