Chapter 4

THE SEGMENTATION MODULE

4.1 Introduction

In this chapter, the details of the segmentation module are carefully discussed. There are three steps in the segmentation module: image registration, image smoothing, and image segmentation. The purpose of image registration step is to spatially register all four images so that pixels in different images with the same coordinates represent the same position in the bag. Section 4.2 discusses the image registration algorithm. Much of the discussion in that section focuses on the development of a robust algorithm that can successfully register images even in the presence of strong noise. Due to severe noise appearing in images, the images must be filtered before reasonable segmentation can be done. This operation is called image smoothing and is discussed in Section 4.3.

Section 4.4 discusses image segmentation. The goal of image segmentation is to group x-ray image pixels into meaningful regions, where each region represents an object either partially or in its entirety [JAI95]. Each region should have relatively uniform properties; in this research, uniformity in the gray level is what is important. However, due to the complexity of x-ray bag images, it is difficult to find a proper segmentation algorithm that works on x-ray bag images. Consequently, the author began the development of a segmentation algorithm by trying to characterize x-ray images and determine how those characteristics affect most of the existing segmentation algorithms.

X-ray bag images have very complex image characteristics. The image segmentation algorithm would have been much simpler if each object in an x-ray image has relatively uniform gray levels. Unfortunately, this kind of cartoon-like property does not exist in x-ray images [JIA94]. In x-ray images, objects in a bag can be generally categorized into two classes: *textile* objects and *solid* objects [DRA98, LU98]. *Textile* objects refer to objects that basically do not have shape, e.g. clothes and towels. *Solid* objects refer to objects with constant shapes in a bag, such as Walkman cassette players, shampoo bottles, hair dryers, shoes, etc. Though most types of explosives are in the form of powder, they are usually placed inside small containers when carried. Since containers are *solid* objects, explosives are considered to be *solid* objects as well. Plastic explosives can also be classified as *solid* objects because the shape of an object made of plastic explosive remains constant until a relative strong force is added to extend or change shape of the object. In other words, its shape remains constant in a bag.

As the author pointed out at the beginning of Chapter 1, shape information cannot be used in finding explosives in a bag because plastic explosives, for example, can be molded into any shape. This statement is true in a general sense. For example, a shoe-shape object may mean this object is a shoe. But plastic explosive can also be molded into a shoe-shape object. Although it is a shoe-shape object, it is not a shoe. However, certain object shape properties such as the flexibility of an object can be used in determining an object's material type.

Solid objects like explosives are often placed inside *textile* objects, such as clothes. A *solid* object usually has relatively uniform gray levels across its image region, while a *textile* object usually has large gray level variations. When a *textile* object overlaps with a *solid* object, the gray level variation increases for the image region of the *solid* object.

It is not easy to segment *solid* objects because *textile* objects are usually overlapped with *solid* objects, and the gray level variations of the *solid* objects become larger [LU98]. It is even more difficult to segment *textile* objects because of their inherent large gray level variations. Though perfect segmentation cannot be achieved, reasonable segmentation

can be achieved. The sole purpose of image processing is to identify the *true* gray levels of potentially threatening objects. Since potentially threatening objects are usually *solid* objects, for a reasonable segmentation, a *solid* object must be successfully segmented into one or several large size regions. It is possible that *textile* objects have been segmented into many small regions. Since *textile* objects are always considered to be the background objects, if those background regions can provide enough information so that the *true* gray levels of the foreground object can be computed correctly, the segmentation is considered to be reasonably good.

After a literature review, it was found that there are several types of algorithms can be used for segmenting x-ray bag images. Generally speaking, the simple techniques only work on simple images but run fast [JIA94]. Complex techniques work on both complicated and simple images. However, the time needed to implement such an algorithm is significantly longer, and by its very nature is much more computationally complex. To find a proper algorithm that can segment x-ray images reasonably well, a number of criteria are used to select such an algorithm. The segmentation results should not be influenced by images collected using slightly changed scanning conditions. The algorithm should have the ability to provide necessary results for different processing objectives. Also the algorithm should be able to be applied to different kinds of images without much modification [BAL82, JAI95, JIA94]. Also the speed of the computation should be reasonably fast. This research indicates that there are four possible approaches to segment a gray level x-ray image: thresholding methods, boundary-based methods, region-based methods, and hybrid methods, methods that combine the boundary and region information in doing the segmentation [ADA94].

Assume that all the pixels belonging to a region lie within a certain range of gray levels while other pixels do not, then a thresholding method can be used for segmentation [KOB97, PAN98, SEK96, YAM96, ZHA97]. A thresholding method presets one or several thresholds, or uses local or global histograms to derive those thresholds. The image is examined on a pixel-by-pixel basis. The pixels of an object of interest are separated from the background using the thresholds. A thresholding method is only

useful for simple environments, such as an environment that contains very few objects and where each object has a large contrast relative to the other objects present. The objects must have relatively uniform gray levels. In other words, those objects need to be *solid* objects. If *textile* objects are present in the image, the background gray levels will have significant variations. A thresholding method will not separate the object of interest from its background correctly. In addition, a thresholding method ignores any spatial information that is potentially useful for segmentation, and it does not cope well with noise or blurring at boundaries.

Boundary-based methods are based on the following fact: the gray level difference of two boundary pixels that belong to two different neighboring regions is much larger than the gray level difference of two pixels that belong to the same region [BEV89, JIA99, WAN94, ZHO97]. A Gaussian or Sobel filter can be used to determine which adjacent pixels have markedly different gray levels. A threshold is used to identify edge points. General methods for dynamically setting this threshold do not exist. The edges must completely encompass a region in order for this region to be identified. Edge following methods for linking edge points together to form a complete boundary are computationally complex. It would not work well unless the edge operator can find all the edge points. The boundary-based methods, if used for x-ray image segmentation, may be affected by the appearance of *textile* objects. The interior of a *textile* object will be segmented into many small regions due to the object's large gray level variations. The boundaries between two regions in x-ray bag images are often blurred. This makes the use of boundary-based methods very difficult. Thus, this type of segmentation method is not a good method for segmenting x-ray bag images.

Region-based methods are based on the assumption that the object will have reasonably uniform gray levels [BHA99, IKO98, KIM98, MOS98, SCH98]. Region growing is one of the most often-used region-based segmentation methods [JAI95]. The algorithm starts from a seed region and adds pixels that satisfy certain constraints to the seed region. Eventually it will grow this region into a large region. The split-and-merge method is another often-used method. First, an image is segmented into many regions using the

region-growing method. Regions are then split and merged to refine the spatial structures using boundary, size, orientation, gray level, etc. Using this type of method, a *solid* object will be more likely grown into one region though gray level variance may exist. The size and boundary information can be used to adjust the shape of the labeled regions. Region-based methods are good methods for segmenting x-ray bag images.

The hybrid methods combine boundary and region information to segment an image [CHE98, GAU99, JAI95, MOG98]. An example of a hybrid method is the morphological watershed segmentation [BAL82]. The watershed method is generally applied to the gradient of the image. This gradient image can be viewed as a topography, in which boundaries between regions are considered as ridges. Segmentation is equivalent to flooding the topography from the seed points with region boundaries being erected to keep water from different seed points from meeting each other. Unlike the boundary-based methods, the watershed is guaranteed to produce closed boundaries even if the transitions between regions are of variable strength or sharpness. This technique has difficulties with regions in which both noise and blurred boundaries appear. This segmentation method is entirely data-driven and needs no high-level region knowledge. However, it is very computationally complex. Considering that the image-processing system should only use a segmentation algorithm that can run in a reasonable amount of time, this hybrid method is not a good choice for segmentation.

After this literature review, a candidate segmentation algorithm was found that the author believed might work on x-ray bag images. This candidate algorithm was then implemented and studied. Based on the results of the study it was then modified to meet the specifications of the image-processing system. A detailed discussion of this algorithm is given in Section 4.4.

4.2 Image Registration

As discussed in Section 4.1, the purpose of image registration step is to spatially register all four images so that pixels in different images with the same coordinates represent the

same position in the bag. Figure 4.2-1 illustrates the concept of image registration. To register all four of the gray level images Img^L, Img^H, Img^F, and Img^B, a target image was selected. In this case, Img^L is used as the target image, and the reason was that low-energy transmission image has higher spatial resolution than scatter images, and the dynamic range of gray levels in low-energy images is greater than the dynamic range of gray levels in high-energy images. The other three images are then registered to this target image. The scatter images Img^F and Img^B are collected at the same time as the low-energy transmission Img^L. Because less photons are scattered than are transmitted, the integration time per pixel in the scatter images is twice that of the low-energy transmission image. As a consequence these scatter images contain 1/2 the number of rows as does Img^L. So to register scatter images Img^F and Img^B to low-energy transmission image Img^L, all that needs to be done is to repeat each row twice in each of these images. Doing so puts them in perfect spatial registration with Img^L.

Since the high-energy transmission image Img^H has the same spatial resolution as the low-energy transmission image Img^L, there is no need to adjust the size of Img^H. However, high-energy transmission image Img^H is created after the other images by reversing the direction of the feed belt and backing the bag through the imaging area. Once the bag has cleared the imaging station the belt direction is again reversed and Img^H is collected. Since the belt is not precisely controlled, the region representing the bag in the high-energy transmission image Img^H will typically have a different starting column and ending column than the regions representing the bag in the other three images. The bag position in Img^H needs to be adjusted to the same starting and ending columns as the rest three images. By finding minimum-inscribing rectangles of all four images, the extracted sub-images are registered to each other. This will be explained later in this section.

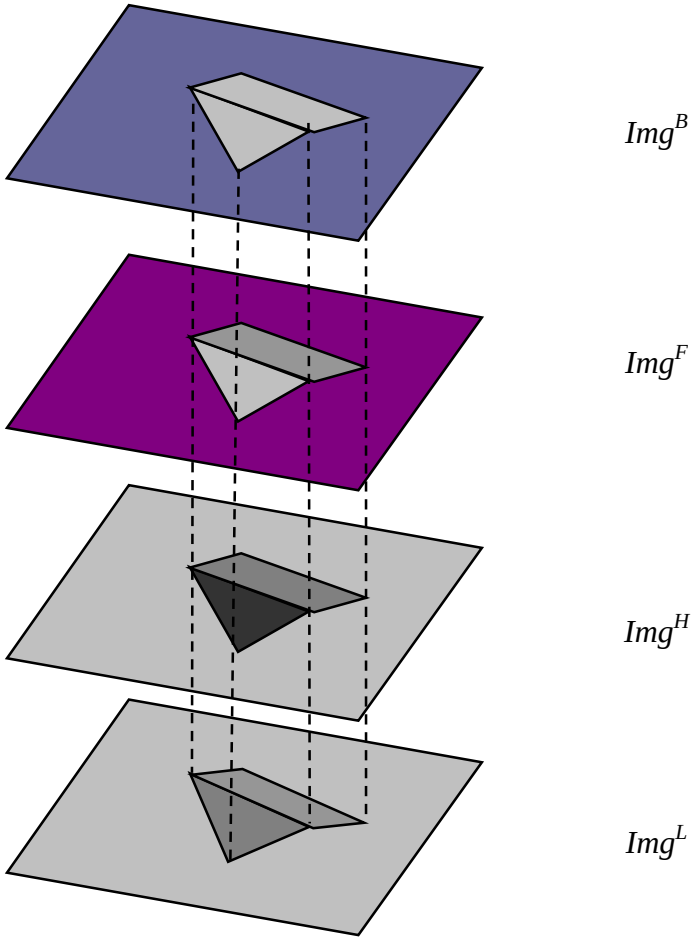


Figure 4.2-1 Illustration of image registration. Gray level images Img^H, Img^F, and Img^B are all registered to gray level image Img^L.

In summary, the gray level image registration needs to be done in two steps:

Procedure *registration()*

1. Size adjustment: adjust scatter images Img^F and Img^B to the same size as Img^L by applying procedure *adjust_size()*.
2. Data reduction: find the minimum-inscribing rectangle of all four images using procedure *find_mir()*.

Adjusting the size of the scatter images Img^F and Img^B is a simple operation. Since the spatial resolution of scatter images is one-half the row-resolution of the transmission image Img^L, and since there is no image distortion, the size adjustment only requires copying row i of a scatter image into rows $2i$ and $2i+1$ of the new image. The two new images will then be perfectly registered to Img^L. The size of a scatter image is $n/2 \times m$. The size of a transmission image is $n \times m$. After adjustment, the size of the new image Img^F_{Adj} or Img^B_{Adj} becomes $n \times m$ as well. More formally the procedure for adjusting the size of the scatter images is given below:

Procedure *adjust_size()*

1. For forward scatter image Img^F, copy row i to row $2i$ and row $(2i+1)$ of the new image Img^F_{Adj}.
2. Repeat step 1 for all $n/2$ rows of Img^F.
3. For backscatter image Img^B, copy row i to row $2i$ and row $(2i+1)$ of the new image Img^B_{Adj}.
4. Repeat step 3 for all $n/2$ rows of Img^B.

During image scanning, the imaging device always collects some extra columns of data before the leading edge of the bag enters the imaging area. This is done in order to assure that an image of the entire bag is created. The device will continue to collect some extra columns of data after the trailing edge of the bag leaves the imagining area for the same

reason. So a bag region is always surrounded by regions of air except the bottom part of the bag region. The bottom of a bag region is also the bottom of the image.

To reduce the computation complexity of a number of analysis steps, a data reduction step was used. This reduction step involved finding the minimum-inscribing rectangle that contains all the pixels of the bag. This operation also serves the purpose of position adjustment for Img^H. This is because the size of a bag remains the same in all four images. The bag in Img^H is extracted along its minimum-inscribing rectangle to form a new image Img^H_{Cut}. The bag in Img^L is also extracted along its minimum-inscribing rectangle to form another image Img^L_{Cut}. Since the newly formed images have the same image size, which is the size of the bag region, both Img^H_{Cut} and Img^L_{Cut} are spatially registered. To register all four images, this *data reduction* operation needs to be applied to the scatter images Img^F_{Adj} and Img^B_{Adj} as well. Since those two scatter images have already spatially registered to Img^L, the minimum-inscribing rectangles of the bag in the scatter images are exactly the same as in Img^L. Using the coordinates of the minimum-inscribing rectangle in Img^L, Img^F_{Cut} is extracted from Img^F_{Adj}, and Img^B_{Cut} is extracted from Img^B_{Adj}. Images Img^L_{Cut}, Img^H_{Cut}, Img^F_{Cut}, and Img^B_{Cut} are spatially registered. The procedure is described as follows:

Procedure *find_mir()*

1. Extract the minimum-inscribing rectangle of the bag region in Img^L using procedure *cut_hist(230)*. Call this new image Img^L_{Cut}.
2. Let $Left^L=Left$, $Right^L=Right$, and $Top^L=Top$.
3. Extract the minimum-inscribing rectangle of the bag region in Img^F_{Adj} using procedure *cut_data($Left^L$, $Right^L$, Top^L)*. Call this new image Img^F_{Cut}.
4. Extract the minimum-inscribing rectangle of the bag region in Img^B_{Adj} using procedure *cut_data($Left^L$, $Right^L$, Top^L)*. Call this new image Img^B_{Cut}.

5. Extract the minimum-inscribing rectangle of the bag region in Img^{H} using procedure *cut_hist(250)*. Call this new image Img_{Cut}^{H}.

Procedure *cut_hist(T)* is used to extract the bag region along its minimum-inscribing rectangle for transmission images. Threshold *T* is used to separate the foreground pixels from the background pixels. Pixels with gray levels less than or equal to *T* are assumed to be foreground pixels while those with gray levels greater than *T* are assumed to be background pixels. For low-energy transmission image, *T* is set to 230. For high-energy transmission image, *T* is set to 250. The method used to select those values will be described later. Procedure *cut_data(left, right, top)* is used to extract the bag region along its minimum-inscribing rectangle for scatter images. This assumes that the coordinates of the boundaries of the minimum-inscribing rectangle, *left*, *right*, and *top* are known after running step 1 of the procedure. Procedure *cut_hist(T)* and *cut_data(left, right, top)* will be described below.

Finding an object's minimum-inscribing rectangle is not typically a difficult process. A single-thresholding method might do the job in some types of images. Unfortunately, due to some gray level distortion in the images generated by the modified AS&E system, a single thresholding method will not consistently find the correct exterior boundaries of bags. This distortion occurs only in those pixels that are located in the bottom part of the images. This is illustrated in Figure 4.2-2. There exists a line that separates the image into two regions. Above this line a single threshold can be used to accurately separate background pixels from the foreground pixels. Below this line, a single thresholding method will not work properly. It should be noted that this distortion line is an artificial line; it does not exist in the real image. After careful observation, it was noticed that this distortion line could usually be drawn at about 70 rows from the bottom of the image. The upper boundary of most bags usually appears at least 100 rows from the bottom of the image. So the distortion line is usually below the upper boundary of a bag. This indicates that the single thresholding method could at least be used to find the upper boundary of a bag.

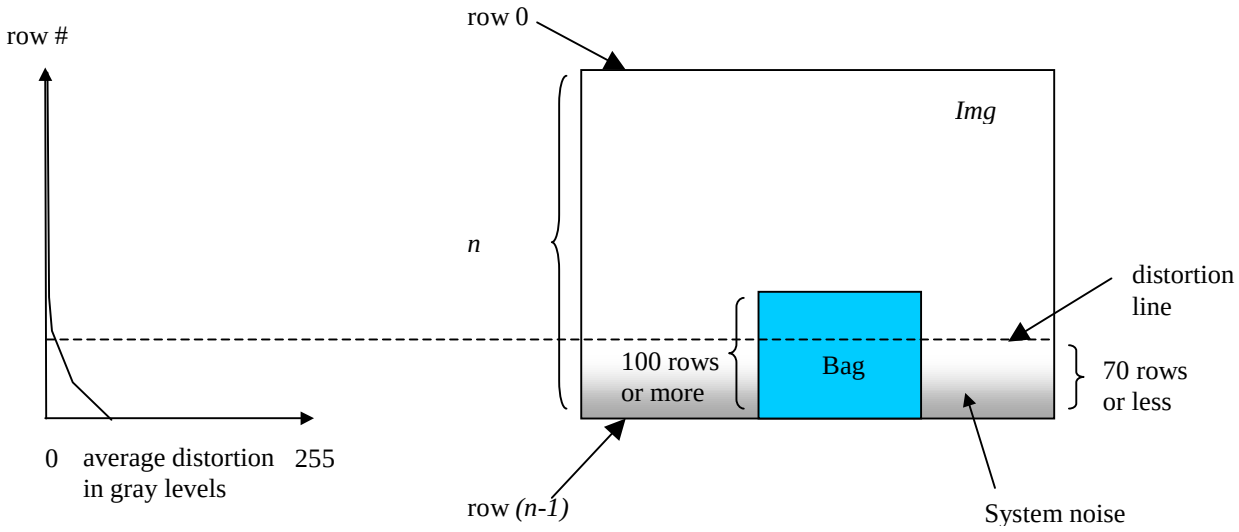


Figure 4.2-2 Illustration of noise in a low-energy transmission image Img^L that is obtained from AS&E system. One may notice that the bottom part of the image has been severely degraded by system noise. On the left, the gray level of the average noise in a row is plotted against the row number. The noise is quite insignificant above the distortion line, a line that is 70 rows from the bottom of the image. The noise becomes significantly large below the distortion line.

Figure 4.2-3 illustrates the process of finding the upper boundary of the bag in a low-energy transmission image Img^L. The procedure starts with row-wise search for the first row – row 0. The row-wise search examines all pixels in a row from left to right. After all pixels in one row have been examined, the procedure moves to the next row and starts the row-wise search again. The row of the first foreground pixel encountered is the upper boundary of the bag. For Img^L, the threshold *T*, which is used to separate the background from foreground, needs to be determined. In order to find this proper threshold, about 40 bag images were collected from a variety of different types of bags. Different threshold values were tested on those images. It was found that when the threshold was set to 230, the results obtained were the most precise. So *T* is usually set to 230 in a low-energy transmission modality.

Assume that the above algorithm finds the upper boundary to be row *Top*. The bottom boundary of the bag is at row *(n-1)*, which is also the bottom of the image. The average gray level of column *i*, *Avg[i]*, is computed using data from row *Top* to row *(n-1)* for all the image columns. The absolute difference between the average value of column *i* and column *i+1*, *Diff[i]*, is computed for all the columns except the last one. *Diff* is plotted against the image as seen in Figure 4.2-4. A peak shows up where there is a transition from the background to the foreground or from the foreground to the background.

To identify the peaks, a threshold value *T'* is used. The level of the gray levels in the bottom part of the image is not predicable. The noise in the bottom part of one image can be extremely severe but much less severe in another image. If the noise gets severe, a small peak in the air may surpass a preset threshold if this threshold is set too low. However, if this threshold is set to a value that is too high, it is possible for the algorithm to miss the peaks of the noise and the bag's left and right boundaries altogether. It is difficult to preset a threshold value that can be used to correctly identify the bag boundaries.

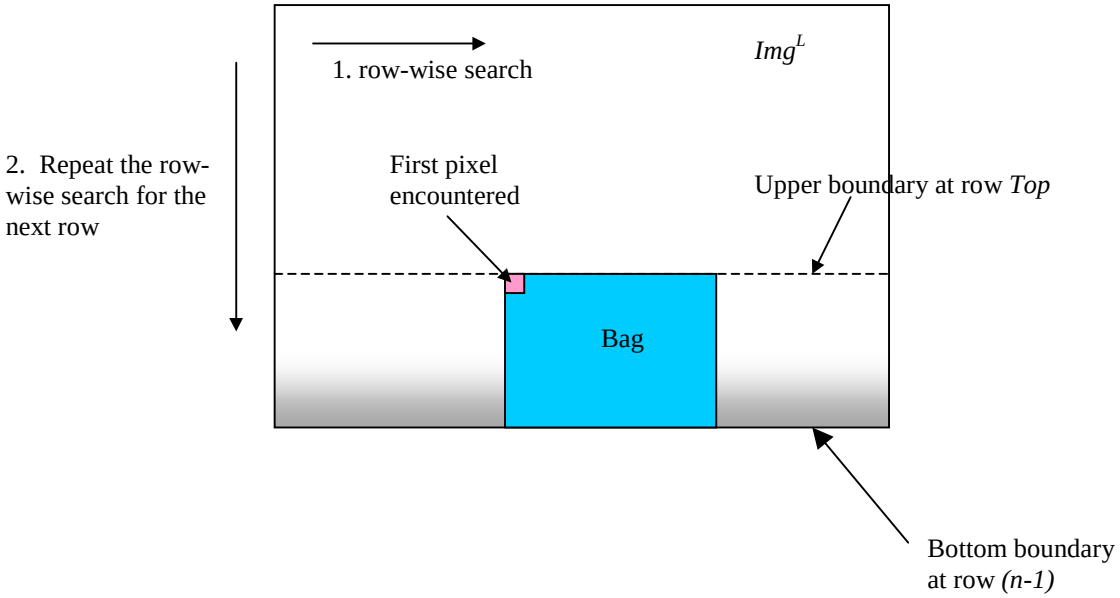


Figure 4.2-3 Illustration of using single-thresholding algorithm to search for the upper boundary of the bag in a low-energy transmission image Img^L. The procedure starts row-wise search first. It examines pixels in a row from left to right. After finishing examining one row, the procedure begins row-wise search with the next row. The row of the first foreground pixel encountered is the upper boundary of the bag.

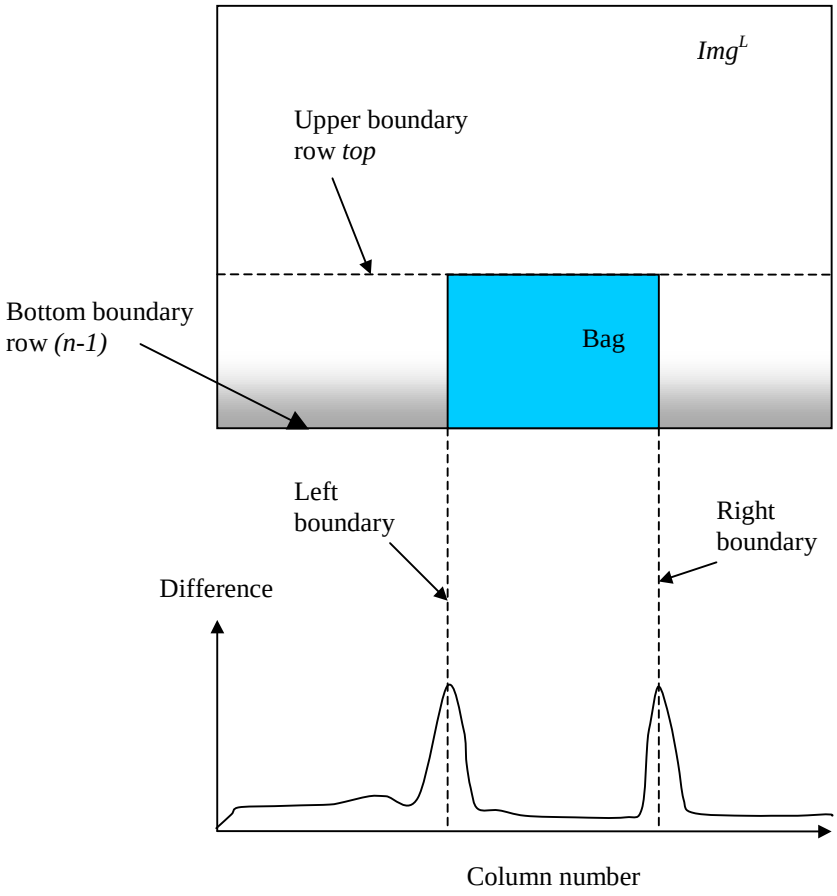


Figure 4.2-4 Illustration of using column difference value *Diff[i]* to identify the left and right boundaries of the bag in a low-energy transmission image Img^L.

An improvement to this is to dynamically compute *T'*. This method requires the AS&E system to collect a number of columns of an image of air to identify the largest noise peak in the background region. *T'* is then set to twice the height of the largest noise peak. This improvement resolves two issues. First, the dynamically computed *T'* will most likely avoid the situation where an image column with a large noise peak is identified as the boundary. Second, twice the height of the largest noise peak in most cases is smaller than the peak caused by the boundary. Using this value, the left and right boundaries of a bag can be correctly identified.

Because of the above AS&E system collects at least 100 columns of the background air image data before starting to collect image data from a bag. The average and difference values of the first 100 columns are then computed. *T'* is set to twice the height of the largest value of *Diff[0...99]*. Note: column 0 is the leftmost column of the image. The peak search begins from left to right on array *Diff[100...(m-2)]*. Assume *Diff[i]* is the first value encountered that is greater than *T'*, this indicates the left boundary of the minimum-inscribing rectangle is at column *i*. The search stops and then restarts from right to left on array *Diff[100...(m-2)]*. Assume *Diff[i']* is the first value encountered that is greater encountered that is greater than *T'*, this indicated the right boundary is at column *i'*. To avoid confusion, the preset threshold *T* used in Img^L is denoted by T^L; the preset threshold used in Img^H is denoted by T^H.

The same procedure can also be applied to the high-energy transmission image Img^H to compute T^H. Since the x-ray beam has higher penetration capability at 150 keV than at 75 keV, the high-energy transmission images are usually brighter than the low-energy transmission images. To determine the threshold T^H, which is used to separate the foreground from background pixels, the same method and the high-energy images of the same 40 bag images that were used to obtain the threshold value as were used to define *T'* used in the low-energy transmission images. Threshold T^H was found to be 250.

Assume that a pixel *p* has coordinates *(i, j)*, where *i* stands for column *i* and *j* stands for row j. Function *g(i, j)* gives the gray level of pixel *p*. Also assume the transmission

image has *n* rows and *m* columns. The threshold *T* is the threshold that is used to separate the foreground pixels from the background pixels. *T* is set to T^L for Img^L, which is 230; and T^H for Img^H, which is 250. Threshold *T'* is the dynamically computed threshold that is used to find the left and right peaks of array *Diff*. Variable *Top* records the row number of the top boundary, variable *Left* records the column number of the left boundary, and variable *Right* records the column of the right boundary. Procedure *cut_hist(T)* for finding the minimum-inscribing rectangle in a transmission image is described as follows:

Procedure *cut_hist(T)*

1. Begin examining pixel *(0, 0)*.
2. For pixel *(i, j)*, if *g(i,j)<T, g(i+1,j)<T, and g(i+2,j)<T*, let *Top=j*.
3. If *Top* has not been found, let *i=i+1*, repeat step 2.
4. If *Top* has not been found, let *j=j+1*, repeat step 2 and 3.
5. For column *i*, compute the average gray level *Avg[i]*, where
$$Avg[i] = \sum_{j=Top}^{n-1} g(i, j) \Big/ [(n-1) - Top + 1].$$
6. Repeat step 5 for column 0 to column *(m-1)*.
7. For column *i*, compute the difference *Diff[i]*, where
$$Diff[i] = |Avg[i+1] - Avg[i]|.$$
8. Repeat step 7 from column 0 to *(m*-2*)*.
9. Let $T' = 2 \cdot \max\{Diff[i] | i = 0,2,\ldots,99\}$.
10. Start from column 100.
11. For column *i*, if *Diff[i]>T'*, let *Left =i*.
12. If *Left* has not been found, let *i=i+1*, repeat step 11.
13. Start from column *(m-2)*.
14. For column *i*, if *Diff[i]>T'*, let *Right =i*.
15. If *Right* has not been found, let *i=i-1*, repeat step 12.
16. Extract the image in the minimum-inscribing rectangle that is bounded by *Left*, *Right*, *Top*, and *(n-1)*.

Since noise is still a big problem above the distortion line, step 2 checks the gray levels of pixel *(i, j)* and its immediate two right neighbors instead of just the gray level of pixel *(i,j)* to determine if pixel *(i, j)* is the foreground pixel.

To extract the minimum-inscribing rectangle of the bag region from scatter images, the boundary coordinates of the low-energy transmission image are used. Routine *cut_data(left, right, top)* is described as follows:

Procedure *cut_data(left, right, top)*

1. Extract the image in the minimum-inscribing rectangle bounded by *left*, *right*, *top*, row *(n-1)*.

where *left*, *right*, and *top* are data obtained from *cut_hist(T)* when this routine is applied to Img^L. Row *(n-1)* is the bottom of the image; it is also the bottom boundary of the bag region.

4.3 X-ray Image Smoothing

X-ray images are inevitably degraded by errors of approximation due to the discrete nature of the image, x-ray source, detector, and electronic device noise [NAG78]. This causes random changes of gray levels at some image pixels. This type of noise is called salt-and-pepper noise.

A pixel with significant noise usually has a much different gray level than its immediate neighboring pixels even though both pixels may have been obtained from the same scanned object. For most segmentation algorithms, the chance of a pixel with significant noise being wrongly identified as a singleton region is quite large. The segmentation will produce a large number of regions with only one or two pixels. It is very difficult to interpret the meaning of such small regions because they do not really represent any objects partially or in their entirety. The large number of those segmented regions may

also cause unnecessary computation burden to the image-processing system that needs to run in real-time.

The purpose of image smoothing is to remove this type of noise as much as possible. By smoothing, the noise in the image data is greatly reduced. The gray level variations of images become more uniform. The probability that this noisy data will cause the image to be segmented into a large number of singleton regions gets smaller. The images will most likely be segmented into fewer regions, each of which is relatively large in size.

There are many types of smoothing filters that have been designed to remove image noise [BEV89], such as a Gaussian filter, a median filter, etc. Most filters have the capability of removing high-frequency salt-and-pepper noise; but they also blur the boundaries between two neighboring regions after smoothing [BEV89]. Boundary information is crucial for accurate image segmentation. An edge-preserving smoothing filter that was developed by Nagao and Matsuyama [NAG78] attempts to remove the image noise without blurring the boundaries between the neighboring regions. To some extent, this smoothing filter even has the capability to enhance blurred edges. It is chosen as the smoothing filter for this application because of its edge-preserving capability.

The algorithm starts by constructing eight masks as seen in Figure 4.3-1. In this figure, each vertex in the mesh represents a pixel. Each mask covers 6 pixels in this mesh. The central pixel p in the mesh is the pixel to be smoothed. Four of the masks are pentagonal masks and four of the masks are hexagonal masks as seen in Figure 4.3-2. The pentagonal mask contains pixels p_1, p_2, p_3, p_4, p_5, and p. The hexagonal mask shown in this figure contains pixels p_6, p_7, p_8, p_9, p_{10}, and p. The number of pixels remains the same for all masks, which is 6. Assume pixel p_k belongs to a mask *Mask*, and the number of pixels in *Mask* is N_{Mask}. The gray level average μ_p and variance σ_p of pixel p when applying *Mask* can be computed using the following equations:

$$\mu_p = \frac{\sum_{p_k \in Mask} g(p_k)}{N_{Mask}} \tag{4.3-1}$$

$$\sigma_p \approx \frac{\sum_{p_k \in Mask} [g(p_k) - \mu_p]^2}{N_{Mask}} \tag{4.3-2}$$

where *g(p)* is the gray level of pixel *p*. For a pixel *p*, the region with the smallest variance σ_p must be the most homogeneous neighborhood region. The gray level of *p* is set to the average gray level μ_p of that region. Once a point has a neighborhood of constant gray levels, its gray level will never change by smoothing again. Therefore, the number of points with changed gray level values by smoothing will gradually decrease to zero. Because the boundary direction is usually the direction with the largest variance, and a pixel *p* is only smoothed towards the direction with the minimum variance, thus the boundary information is preserved.

The algorithm is an iterative process. The ideal converging condition is that none of the pixels of the smoothed image changes their gray levels since the last iteration. When this occurs, the algorithm terminates. An image with the size 300 by 200 usually needs 20 iterations to reach this ideal converging condition. It was observed that after 5 to 6 iterations, the number of pixels changing gray levels for each additional iteration is usually less than 200 pixels, compared to over 1000 pixels changed for earlier iterations. To make the algorithm run faster, the converging condition used in this algorithm is modified.

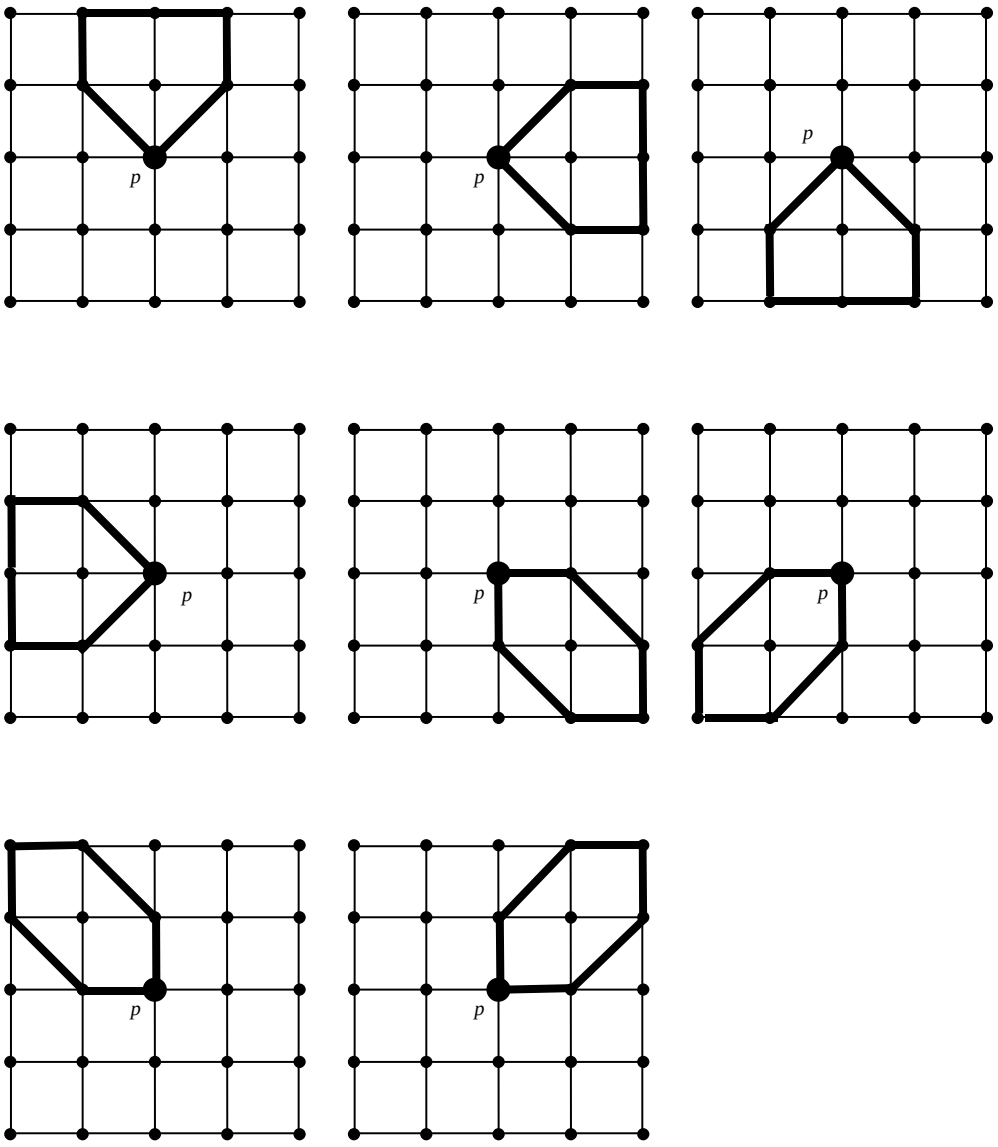


Figure 4.3-1 Diagrams of all 8 masks that are used for smoothing operation.

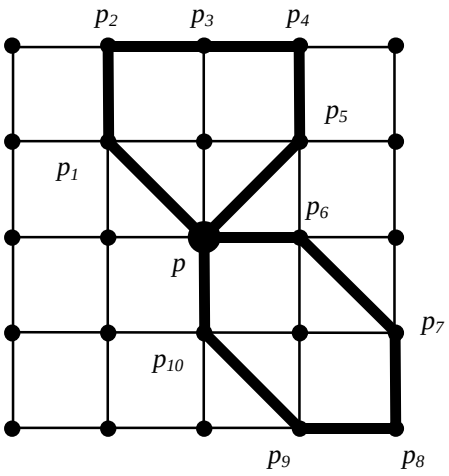


Figure 4.3-2 Diagram of the bar mask used to compute the variance and average gray levels of the neighboring pixels. For the two forms of masks, each would have four copies in four directions. All eight marks are centered around pixel *p*.

The modified converging conditions have two convergence criteria. The first criterion is if the number of pixels changed during an iteration is less than 200, the smoothing operation should terminate. The number 200 was selected because the smoothed image resulting from this termination criteria looks no different from the smoothed image under Nagao and Matsuyama's ideal condition where no pixels change value. Assume that at the n^{th} iteration, the number of pixels changed is *Chng[n]*. The second criterion is if the difference between *Chng[n]* and *Chng[n-1]* is less than 50 pixels, the algorithm should terminate. Both criteria aim at the situation when not many pixels are left that need to be smoothed. The first criterion uses the absolute number as the converging condition, while the second criterion uses the relative numbers as the converging condition. The modified condition has made the smoothing algorithm runs at least three times faster than using the ideal condition. For an image size with 300 by 200, using the original converging condition, the algorithm needs 40 seconds to run on a Pentium 90 MHz PC. Using the modified condition, the running time has reduced to only 10 to 12 seconds. This algorithm was tested on the same set of bag images used for determining the threshold discussed in Section 4.2. The sizes of most image are about 300 by 200. The running time is around 10 to 12 seconds for those images.

The following presents the smoothing algorithm that is similar to [NAG78] with revised converging conditions

Procedure *smoothing()*

1. Apply the 8 masks around pixel *p* as shown in Figure 4.3-1.
2. Compute the variance for all the pixels covered by each mask.
3. Detect the position of the mask with the minimum variance of the gray levels.
4. Set the gray level at *p* to that of the average gray level of the selected mask.
5. Repeat step 1 to 4 for all the pixels that have at least one neighboring pixel with a changed value.
6. Assume this is the n^{th} iteration. Record the number of pixels changed at this iteration as *Chng[n]*.
7. Iterate step 1 through 5 until *Chng[n]<200* or *Chng[n-1]-Chng[n]<50*.

In the original algorithm, pixel *p* is not assigned to average gray level value of a mask that contains edge pixels. This is in order not to blur a sharp edge. In this image-processing system, since no edge-detection algorithm is used on the image to provide the pixel edge information, this condition cannot be used. Even though this condition is not being used, the routine *smoothing()* still preserves the sharp edges. This is because a mask containing a sharp edge pixel usually has a very large gray level variance; in most cases, a mask applied in this direction will not yield a minimum variance. Thus pixel *p* will not be assigned to the average gray level computed using the mask that contains a sharp edge pixel.

The effect of edge-preserved smoothing is obvious. The salt-and-pepper noise is significantly reduced. Most region boundaries are preserved. Since no edge information is provided, some of the region boundaries have been slightly modified. Fortunately, this will not affect the computation of the average gray level of this region. *smoothing()* should be applied Img_{Cut}^{H}, Img_{Cut}^{L}, Img_{Cut}^{F}, and Img_{Cut}^{B}. The resulted images are Img_{Smth}^{H}, Img_{Smth}^{L}, Img_{Smth}^{F}, and Img_{Smth}^{B} respectively.

Besides the method developed by Nagao and Matsuyama, other types of filtering methods can also be used for smoothing, such as the one developed by Overton and Weymouth [OVE79]. No matter what type of smoothing method is used, the algorithm must preserve the edge information while smoothing.

4.4 Image Segmentation

After an extensive literature search, a candidate algorithm that works on hand-wrist radiography developed by Manos [MAN93] was selected for further investigation. Manos's algorithm is a hybrid approach that combines boundary and region criteria to segment x-ray images. The purpose of Manos's algorithm is to segment hand-wrist radiography in order to remove the soft tissue from the image and reveal the bone regions so that bone density measurements can be made. Hand-wrist radiography in a way has

similar characteristics to x-ray bag images. In a hand-wrist radiography image, both bones and soft tissue appear. The gray level variation of soft tissue is quite large, and can be considered as the *textile* objects. The gray level variation of bones is relatively small, and hence are similar to *solid* objects. Bones are always overlapped and surrounded by tissue. The purpose of the algorithm is to find the regions that contain bones only. This algorithm is a five-step process [MAN93]:

1. Edge-preserving smoothing
2. Region growing
3. Region merging
4. Edge detection
5. Region merging by fusion of boundary information

Due to the appearance of image noise, an image filter is needed to remove this image noise. Most filters remove image noise while also blur the boundaries between different regions. The edge-preserving smoothing resolves this conflict. It removes salt-and-pepper noise while preserving important edge information. The next objective is to organize pixels into regions that could then be merged to form larger meaningful regions. The region-growing algorithm performs this task perfectly. Each region may have different gray levels; but each region can be modeled by a function. The region-merging algorithm uses scores to measure the similarities between different regions; the neighboring regions are merged according to their similarities. Although the region-merging algorithm reduces the number of regions, the images can still be further merged. To further merge those image regions, boundary information is used. The edge detection algorithm first uses Gaussian edge-detection filter to detect region boundaries. The boundary information is then used to merge neighboring regions that have weak boundaries.

All five steps in this candidate algorithm were implemented. Manos's algorithm was extensively tested on a number of bag images. The segmentation results are reasonable on those tested x-ray bag images. However, the entire process lasts over 40 minutes for

an image with size 300 by 200 on a Pentium 90 MHz PC. Even though running in real-time is not part of the system design specifications, but 40 minutes is still to long for processing an image. So efforts were made to improve the speed of this algorithm while still maintaining good segmentation results.

Based on Manos's algorithm, a new algorithm is developed. This new algorithm removes redundant steps and steps that may affect the segmentation results negatively. The new algorithm runs much faster than the candidate algorithm and it produces robust segmentation results on x-ray bag images. Among all the activities involved in developing the new segmentation algorithm, the most time-consuming one was to implement and test the candidate algorithm. Please note that the segmentation algorithm needs only to be applied to the smoothed low-energy transmission image Img_{Smth}^{L}, of which much of the image noise that would affect the segmentation has been removed. Again, low-energy transmission image is selected is because of two reasons: the low-energy transmission image has higher spatial resolution than forward scatter and backscatter images; the dynamic range of gray levels in the low energy transmission is greater than the dynamic range of the gray levels in the high-energy transmission image. Assume that the segmented labeled image is Img^{Lbl}. Since the other three images Img_{Smth}^{H}, Img_{Smth}^{F}, and Img_{Smth}^{B} are registered to Img_{Smth}^{L}, they are registered to Img^{Lbl} as well.

In the new algorithm, Step 1 and 2 of the old algorithm were retained. Other steps in the candidate algorithms are removed from the new algorithm. Step 3 region-merging method is removed from the algorithm because the method has increased the gray level variance of a segmented region. Step 4 and 5 are also removed from the algorithm for two reasons. First the edges on an x-ray bag image are not obvious, the edges cannot easily be detected, and this makes the merging results sometimes incorrect. Second, the edge detection is a very time consuming process, and the edge information is not useful for material characterization besides region merging. The tradeoff between the program running time and the improvement of segmentation results is not worthwhile. The new

segmentation algorithm is optimized in detection capability and speed. It runs about 12 seconds to process an image with size 300 by 200 on a 90 Megahertz Pentium. Since step 1 has already been performed in the routine *smoothing()*, the new algorithm *segmentation* needs only to be applied to the smoothed image Img^{L}_{Smth}.

The new algorithm is as follows:

Procedure *segmentation()*

1. Segment Img^{L}_{Smth} using procedure *region_growing(3)*, the resulted image is Img^{Lbl}_{Rgn}.
2. Segment Img^{L}_{Smth} using procedure *region_growing(5)*, the resulted image is Img^{Lbl}_{Obj}.

Routine *region_growing(T)* is the algorithm that segments an image using region growing method. There are many versions of the region growing algorithms [DOR98, HOJ98, IKO98, JAI95, MAN93]. Compare to other version of region-growing algorithms, the algorithm developed by Manos seemed to segment x-ray images more precisely. The Manos's original region-growing algorithm is as follows [MAN93]:

1. Start from an unlabeled pixel and assign a new unused region label.
2. Merge neighboring pixels (4-neighbors) with gray level difference to the newly labeled pixels if the gray level difference is less than or equal to a threshold *T*.
3. Repeat step 2 until no pixels adjacent to the newly labeled region can be merged.
4. Repeat step 1 to 3 until no unlabeled pixels exist.

The result of segmentation is a set of connected components each having its own unique label. Those connected components form a labeled image denoted by Img^{Lbl}. The low-energy transmission image Img^{L}_{Smth} is first segmented using a strict gray level difference

criterion with T=3. The resulted labeled image is Img_{Rgn}^{Lbl}. Regions in Img_{Rgn}^{Lbl} have relatively uniform gray levels. The sizes of those regions are relatively small; each region only represents a small part of an object. Image Img_{Smth}^{L} is then segmented using a less strict gray level difference criterion with T=5. The resulted labeled image is Img_{Obj}^{Lbl}. Regions in image Img_{Obj}^{Lbl} have relatively larger sizes; each region represents an entire object or a large part of an object. The reason that two segmentations need to be performed will be explained later in Chapter 5.

The strict segmentation criterion reflects the conservative approach to growing a region. The purpose of using a strict criterion is to segment an image into regions with relatively uniform gray levels. For this threshold value, a segmented region will have a difference between the maximum gray level and the minimum gray level that is usually less than 15 gray levels. The average gray level of this region is usually less than 3 gray levels away from the gray level that the main peak in the gray level histogram of this region. Typically there is only one large peak in a gray level histogram of a segmented region. So using the average gray level to characterize a region's gray level is meaningful.

The less strict segmentation criterion reflects the desire to segment an image into regions that have shapes closer to object shapes. In a segmented region, the maximum gray level can be 50 gray levels higher than the minimum gray level. The average gray level of this region does not have a particular meaning because there are usually more than one large peak in the gray level histogram of a segmented region. However, the less strict criterion makes it possible to grow parts of the same object into one large region, though each part of the object may have significantly different gray levels than other part. After visually examining the bag images, it was noticed that the gray level difference between two adjacent objects is typically greater than the gray level difference between different parts of the same object. Thus in most cases, this algorithm will not falsely grow two adjacent objects into one region.

To find the exact threshold values for the strict and less strict criteria, a number of tests were performed on the same set of bag images mentioned in Section 4.2. After extensive testing, *T* is found to be 3 for the strict criterion and 5 for the less strict criterion. When those two criteria are used, the segmentation results were considered the best on those 40 images.

The region growing process is a very time-consuming process. The original algorithm needs a running time of 40 seconds to process an image with size 300 by 200 on a Pentium 90MHz PC. The author made some improvement to the original algorithm. The new implementation needs only 6 seconds to process an image with the same size. To reduce the computation complexity, the algorithm can be best implemented by using a region pixel array *M*. Of a region *Rgn* being grown, *M* records all the pixels that have been grown into *Rgn*. The revised algorithm is as follows:

Procedure *region_growing(T)*

1. Clear region array *M*. Start from an unlabeled pixel *p* and assign a new unused region number *k*. Add label *p* to array *M*.
2. For a pixel *p* in region array *M*, find one of its unmarked 4-neighbor pixel *p'*. If the gray level difference between pixel *p* and pixel *p'* is less than or equal to a preset threshold *T*, mark *p'* with label *k*, and append this pixel *p'* to region array *M*.
3. Repeat the examination for all the unmarked 4-neighbor pixels of pixel *p*.
4. Repeat step 2 and 3 for all pixels in *M* including all the newly appended pixels. The region with label *k* is finished growing after no new neighboring pixels can be found that satisfies the gray level difference condition.
5. Repeat step 1 to 4 until all image pixels are marked.

After segmentation, there are six registered images. Four of them are smoothed gray level images: Img_{Smth}^{H}, Img_{Smth}^{L}, Img_{Smth}^{F}, and Img_{Smth}^{B}. Two of them are labeled images: Img_{Rgn}^{Lbl} and Img_{Obj}^{Lbl}. Since in the smoothed images, regions identified by image Img_{Rgn}^{Lbl}

have relatively uniform gray levels, Img_{Rgn}^{Lbl} is used as the primary labeled image to compute the average gray levels, boundary length, region size, and neighboring regions for a region. Image Img_{Obj}^{Lbl} is used as the labeled image to identify regions in Img_{Rgn}^{Lbl} that belong to the same object.

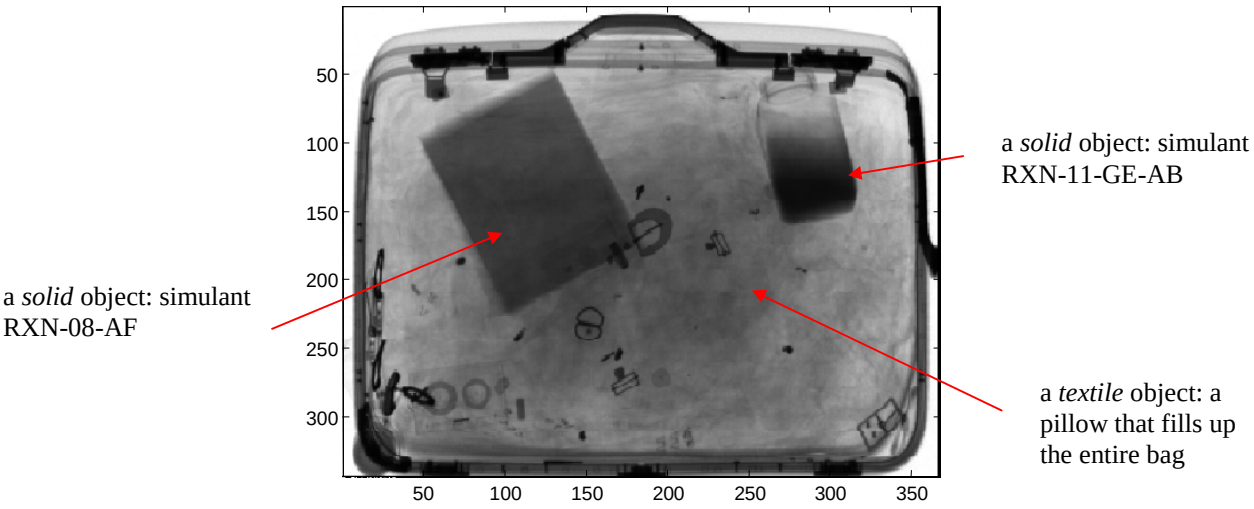
After segmentation, some region merging is needed. The segmentation algorithm still produces regions with extreme small sizes due to the gray level variations and noise. To deal with this, regions with only one or two pixels should be merged into larger regions. There are two reasons for this. First, it is difficult to interpret those one or two pixel regions as objects. Second, there is usually a relatively large number of them; their presence increases the computational complexity of the later stage of processing unnecessarily. The one or two pixel regions must be merged into adjacent regions with which they have the closest gray levels.

Regions with more than ten pixels should not be merged into larger regions because those smaller regions can later be used to compute the *true* gray levels [LU98]. A little bit more discussion is given here regarding to this decision. To compute the *true* gray levels of an object of interest when overlapping occurs, the *true* gray levels of the overlapping object must be known. Since the part being overlapped of the overlapping object is completely hidden behind the object of interest, the only way to know its *true* gray levels is to find a neighboring region of the object of interest, which may be part of the overlapping object. It is possible that the non-overlapped region of this overlapping object has a very small size, but this small size region carries the *true* gray level information of the overlapping object. If somehow it has been merged into a larger neighboring region, this information is lost.

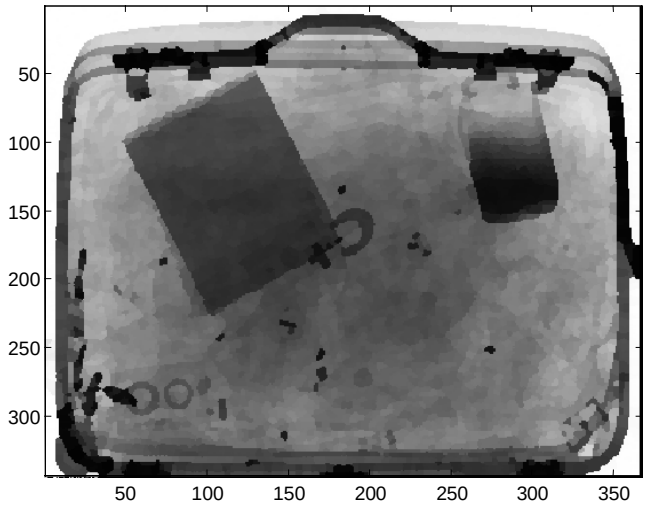
The segmentation result is reasonably accurate. Inside the bag of Figure 4.4-1 (a), there are mainly three objects: two *solid* objects and one *textile* object. The two *solid* objects are two explosive simulants. The *textile* object is a pillow. The descriptions of simulants RXN-11-GE-AB and RXN-08-AF are given in Table 6.1 and Figure 6-3. The *textile* object is a pillow that has roughly the same size as the bag. Using a strict criterion,

RXN-11-GE-AB can be seen segmented into one large region along with many smaller regions as seen in Figure 4.4-1 (c). RXN-08-AF is segmented into one large region. Some parts of the pillow are segmented into very small regions while some other parts of the pillow are segmented into some large regions. The segmentation result is quite satisfied. For any *solid* object, the segmentation procedure generates at least one region with relatively large size. Each segmented region regardless of its size has relatively uniform gray levels. This was verified by examining the histogram of a segmented region. It satisfies the original segmentation goal. Since the focus of the image-processing system is to determine the material type of a *solid* object, how a *textile* object is segmented is not a major concern.

Using a less strict criterion, RXN-11-GE-AB is segmented into regions with larger size as seen in Figure 4.4-1 (d). The pillow is segmented into one large region. The purpose of a second segmentation is to segment a *solid* object into regions closer to the shape of the object, or in other words, to segment a *solid* object into regions with larger sizes. This goal has been achieved in this segmentation.

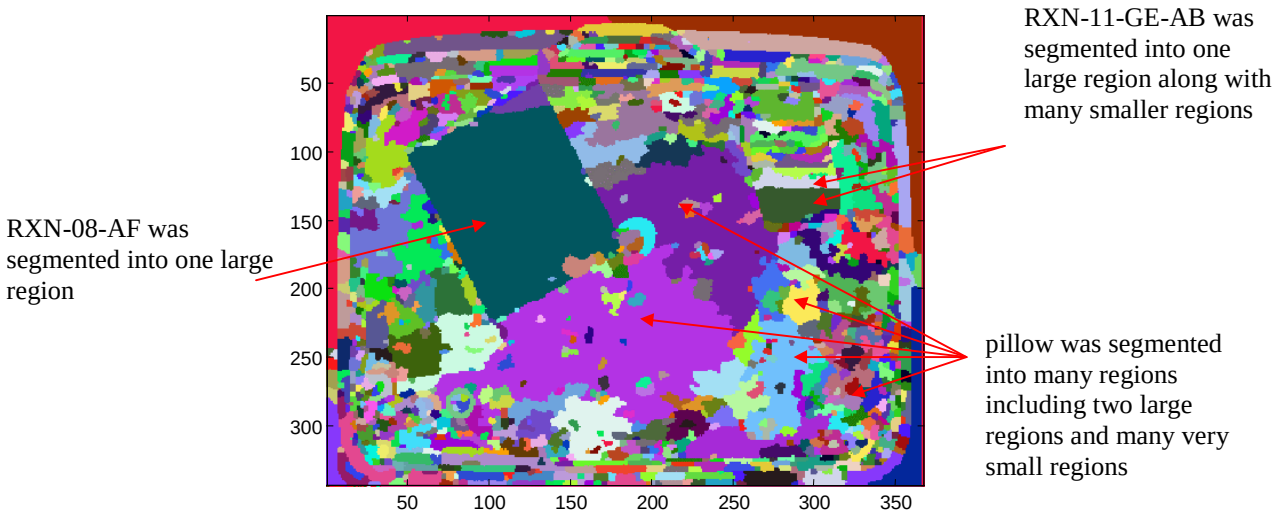


(a)

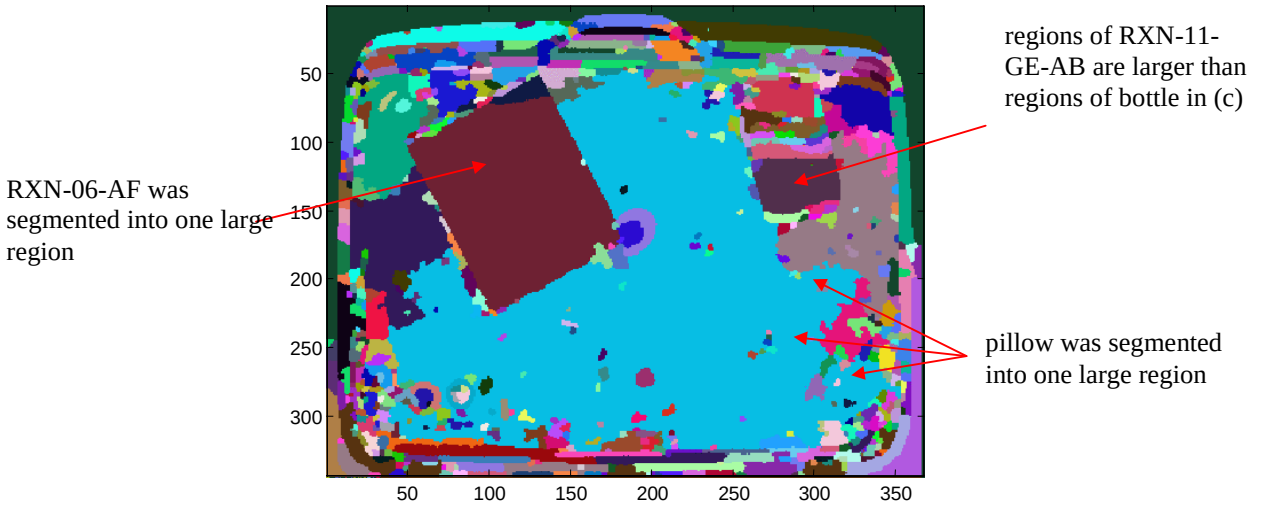


(b)

Figure 4.4-1 Image segmentation results of another luggage bag. a) Low-energy transmission image Img^{L}_{Cut}; b) Smoothed image Img^{L}_{Smth}. (Continued to the next page)



(c)



(d)

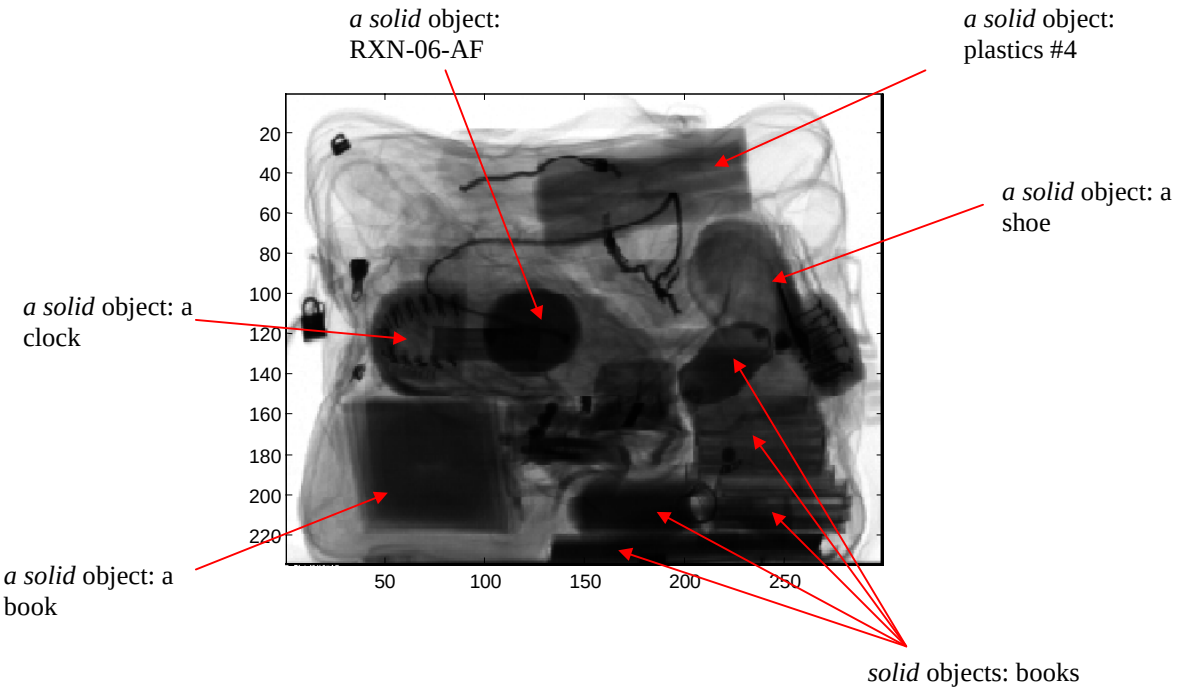
Figure 4.4-1 Image segmentation results of another luggage bag. c) Region label image Img_{Rgn}^{Lbl}; d) Object label image Img_{Obj}^{Lbl}.

Inside the bag of Figure 4.4-2 (a), there are a number of *solid* objects and some *textile* objects. The *solid* objects include simulant plastics #4, RXN-06-AF, a clock, and some books. The descriptions of simulant plastics #4 and RXN-06-AF can be found in Table 6-1 and Figure 6-3. Those *textile* objects are inserted between those *solid* objects, and they are not marked in the Figure 4.4-2 (a). The *textile* objects are basically things like pants, towels, and shirts. Using a strict criterion, RXN-06-AF and books are segmented into large regions as seen in Figure 4.4-2 (a). However, plastics #4 is segmented into many smaller regions. The reason is that this plastic simulant is comprised of three cylinders. The round shapes of the cylinders make the gray level of this object non-uniform. An object with non-uniform gray levels will be segmented into several different regions. When a less strict segmentation criterion is used, plastics #4 is seen being segmented into one large region along with several small regions. The result is what should be expected from this segmentation algorithm.

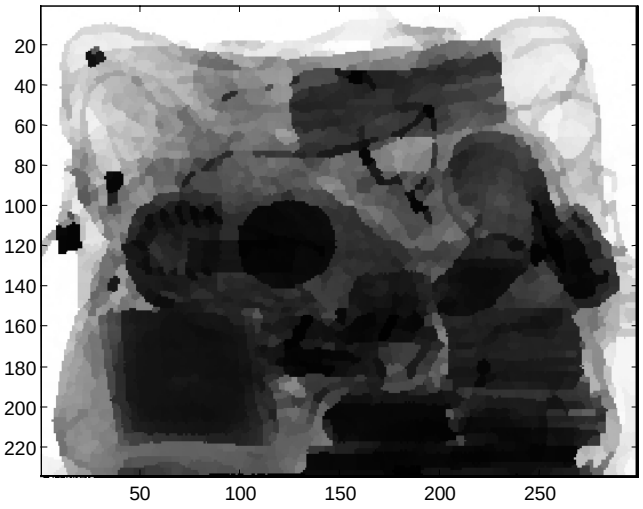
The bag images of Figure 4.4-1 and Figure 4.4-2 represent typical bag images. The algorithm typically takes about 40 to 50 seconds to process an image with size of 300 by 200. The segmentation results are reasonably good. The segmentation algorithm runs reasonably fast.

4.5 Summary

In this chapter, the development of image segmentation module is discussed. The segmentation module contains three steps: image registration, image smoothing, and image segmentation. Images from all four sensing modalities are registered to the low-energy transmission image first. To remove the noise, a smoothing filter is applied to all four images. The low-energy transmission image is then segmented into two different labeled images using the region-growing method under two different segmentation criteria. Some improvement has been made to the original smoothing and segmentation algorithms to make them run more efficiently. The segmentation results are reasonably accurate.



(a)



(b)

Figure 4.4-2 Image segmentation results of yet another luggage bag. a) Low-energy transmission image Img^{L}_{Cut}; b) Smoothed image Img^{L}_{Smth}. (Continued to the next page)

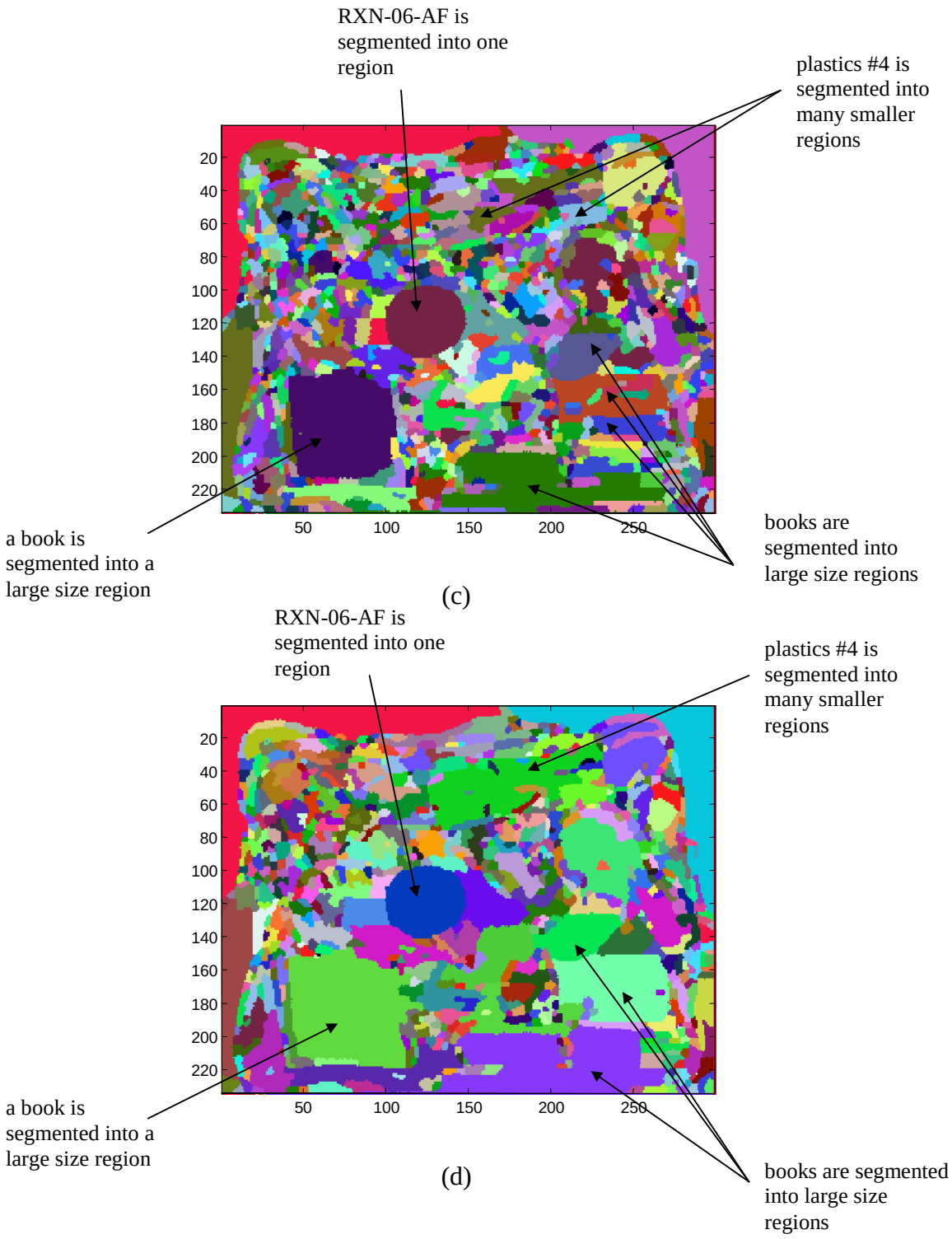


Figure 4.4-2 Image segmentation results of yet another luggage bag. c) Region label image Img_{Rgn}^{Lbl}; d) Object label image Img_{Obj}^{Lbl}.