1. Region Merging.

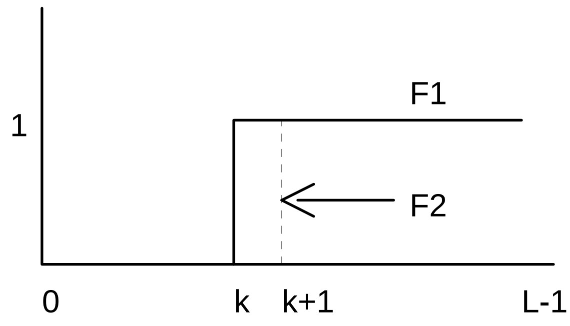
Region based segmentation methods use similarities between pixels and adjacent regions to form regions that constitutes a valid segmentation of the image into a partition with regions that satisfy a uniformity predicate. Gray-level intensities are most often used to define the uniformity predicate but could also use color and texture features.

In attempting to obtain a segmentation of an image one may a.) start with large regions and split them, b.) start with small, atomic, regions and merge or grow them, or c.) alternately split and merge regions. In this chapter, we consider the merging or growing or regions. In creating atomic regions many false boundaries may be created by shadows, uneven illumination, reflections, and non uniformities in the surface of objects. This will give a large number of small regions that must be merged to correspond to the objects. This approach to segmentation is sometimes called bottom-up because one starts with small regions and seeks to enlarge them to form the correct regions corresponding to the objects in the image.

One basic strategy is to compute the first-order statistics of the gray-levels in the regions, namely $p_1(k)$ from R_1 and $p_2(k)$ and R_2, and merge R_1 and R_2 if the statistics are from the same distribution. One needs to develop a search strategy such that arbitrary decisions on selecting the regions to merge do not affect the final result. For example, consider the case of four regions. Suppose one chooses to consider merging R_1 and R_2 to form R_5. Suppose the statistics of R_5 then permit the merging with R_3 but not R_4. Consider an alternate strategy where R_1 and R_3 are considered for merger first to form R_6. In this case, R_6 has statistics that permit it to merge with R_4 but not R_3. One obtains a different result depending upon an arbitrary decision on the region chosen first for merging. The following figure indicates the situation.

Figure-1. Merging Order of Regions

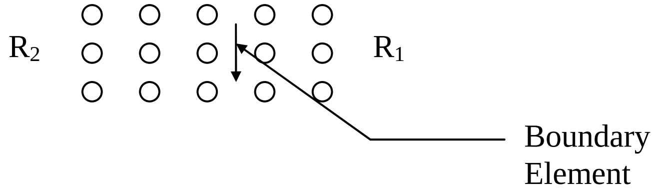
It is desired to merge regions if two regions are uniform or homogeneous. Therefore, we need measures of uniformity. The simplest measure of uniformity would use mean values [Levine, 1985, pp. 390]. A rule would be to merge the regions if their mean values are close. Let μ_1 be the mean of R_1 and μ_2 be the mean of R_2, then merge R_1 and R_2 if $|\mu_1 - \mu_2| < \varepsilon$ where ε is a parameter to the algorithm. A variation would be to compare their cumulative distribution functions. Let F_1 be the distribution of gray-levels computed on R_1 and F_2 the distribution computed on R_2. Then merge the two regions if $||F_1 - F_2|| < \varepsilon$ where the norm of a function F might be defined as $||F|| = \max\{|f(k)|, 0 \le k \le L-1\}$ where there are L gray-levels. This particular norm does not work well since a difference of 1 gray-level can give a large difference in the norm. The following figure shows this case.

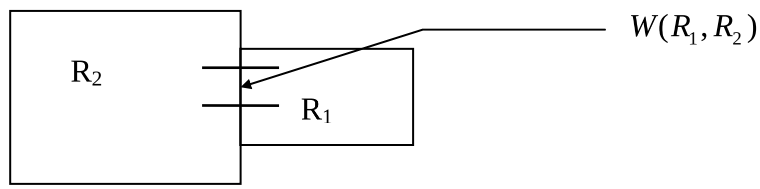


Maximum Difference of Norms

1.1 Region Merging Based on Boundary Measures

Uniformity is a measure obtained from regions that indicates if the regions could be combined. In addition to region properties, the properties of the boundary between two regions can indicate if the regions should be merged. One basic idea is to merge adjacent regions if the common boundary is weak and the resultant region has a shorter boundary than the original regions [Brice and Fennema, 1970, Levine, 1985, pp. 393]. Let R_1 and R_2 be the regions under consideration to be merged. Let p_1 and p_2 be pixels in R_1 and R_2 which are also neighbors. This is a boundary pair [Sonka, Hlavac, and Boyle, 1999, pp. 178-179] and the strength of the boundary is given by $st(p_1, p_2) = \| g(p_1) - g(p_2) \|$. Let $W(R_1, R_2) = \#\{(p_1, p_2) \mid st(p_1, p_2) \le \varepsilon_1\}$ where (p_1, p_2) is a boundary pair. Let P_1 be the perimeter length of R_1, P_2 be the perimeter length of R_2, and $P = \min\{P_1, P_2\}$. Then merge the regions if $\frac{W(R_1, R_2)}{P} > \varepsilon_2$. This says they share a long weak boundary. Another criterion one can use is to merge the regions if $\frac{W(R_1, R_2)}{B(R_1, R_2)} > \varepsilon_3$ where $B(R_1, R_2)$ is the total number of boundary pairs between R_1 and R_2. This says the weak boundary is a significant part of their common boundary.





1.2 Region Growing and Edge Detection.

Assume an initial region segmentation has been obtained. The regions are then to be modified by examining the boundaries between regions. The combined approach should be better than one bases only upon region similarity measures. Region growing methods at times produce false boundaries because it is hard to find a uniformity measure to cover situations such as the gray-levels vary in a linear fashion between regions. Regions may form boundaries that are not true edge points.

Let us now consider some issues and measures that could be used in the analysis of boundaries between regions. There are many different ways that boundary information can be used to form regions. The following discussion indicates some general issues that must be addressed by any method that considers analyzing and removing boundaries.

Let b be a boundary between two regions and $f(b) = \frac{sum\_of\_contrast(b)}{length(b)} + \beta \frac{number\_of\_direction\_changes\_on\_b}{Length(b)}$. The term β is a parameter. The first term measures the strength of the boundary while the second term penalizes long straight lines [Pavlidis and Liow, 1990]. If $f(b) \le \varepsilon$ which indicates it is small, then the boundary is a candidate for removal. One may choose to use only the first term in which case the parameter ε could be the expected contrast across edges. The second term is used to remove artifacts which often appear as straight lines.

There are several constraints that must be considered before boundaries are removed. In the following figure, if boundary b1 is removed then boundary b2 must also be removed. This is called the region constraint. If b3 is removed then b1, b4, b2 must be merged to form a new boundary between region R3 and R1∪R2. In subsequent steps one cannot remove any of the single boundaries b1,b4,b2. This is called the boundary constraint.

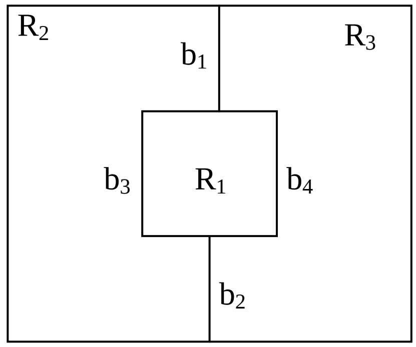


Figure 2. Constraint on Boundary Removal

The following algorithm gives the method for merging the regions.

An algorithm for removing boundaries and merging regions follows.

Repeat the following steps.
set fmin to ε
for every boundary b_i
  do;
    if f(b_i) < fmin and better(b_i)
    then set fmin to f(b_i), imin to i
  enddo;
if fmin ≥ ε then exit
eliminate $b_{i\,\mathrm{min}}$ and rearrange the list of boundaries taking
  into account the edge mergers resulting from the removal of $b_{i\,\mathrm{min}}$.

At each iteration the minimum boundary is eliminated until the minimum exceeds ε.

The function better in the above algorithm is used to test conditions for the removal of a boundary. Consider the following figure and the definition of the function better. One cannot remove both weak boundaries because AK would no longer be a region boundary. If BK is removed then AK and CK must be merged. Since AK is a long strong boundary and CK is a long weak boundary this is not a good strategy. The function better(BK) should be set to 0 in this case. If CK is removed, then AK and BK can be merged since BK is a short boundary. Hence better(CK) could be set to 1.

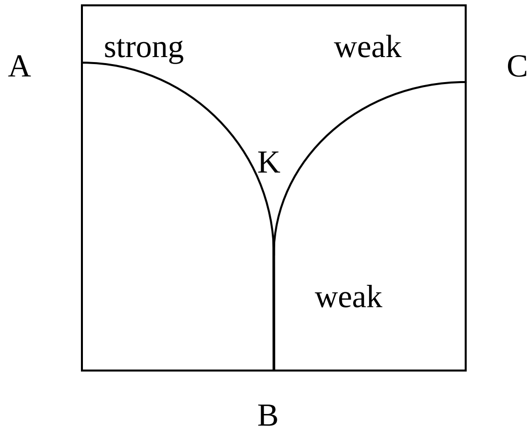


Figure 3 Weak Boundary Considerations

The following example demonstrates this idea.

C

A

K

	3	3	3	3	3	3	3	3
	3	3	3	3	3	3	3	3
	3	3	3	3	3	6	6	6
	3	3	3	3	3	6	6	6
	9	9	9	9	3	6	6	6
	9	9	9	9	3	6	6	6
	9	9	9	9	6	6	6	6
	9	9	9	9	6	6	6	6

B

The following boundary would be obtained.

A

	3	3	3	3	3	3	3	3
	3	3	3	3	3	3	3	3
	3	3	3	3	3	6	6	6
	3	3	3	3	3	6	6	6
	9	9	9	9	3	6	6	6
	9	9	9	9	3	6	6	6
	9	9	9	9	6	6	6	6
	9	9	9	9	6	6	6	6

B

1.3 Seeded Region Growing

This method assumes that atomic regions have been selected and then considers how the regions might be grown from the atomic regions [Adams and Bischof, 1994]. The atomic regions should be selected so that each be interior to one of the regions in the correct final segmentation. If the atomic regions are not interior to the final regions then the final segmentation will be incorrect.

One first considers the pixels that are neighbors to the atomic regions. Let T be the set of pixels that are neighbors to some atomic region. This is expressed in the following expression.

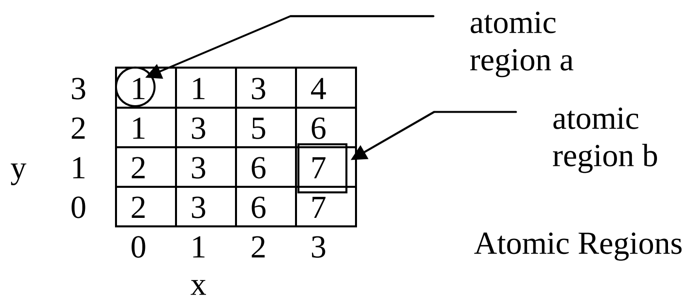
$T = \left\{ p \mid p \notin \bigcup_i A_i \quad and \quad \left(N(p) \bigcap \left(\bigcup_i A_i \right) \right) \neq \varphi \right\}$. Here N(p) is the 8-neighbor region of pixel p. The set of A_i are initially the atomic regions. If pixel p is in T, then for every atomic region A_i such that $A_i \cap N(p) \neq \varphi$, let $\delta_i(p) = | g(p) - mean\{g(p_1) \mid p_1 \varepsilon A_i\} |$. This is the difference between p and A_i which is computed as the difference between the gray-level of p and the average gray-level in A_i. Then select the minimum of these differences over A_i as $\delta(p) = \min\{\delta_i(p) \mid N(p) \cap A_i \neq \varphi\}$. The point we are interested in is $p' = \{p \varepsilon T \mid \delta(p) \text{ is a minimum}\}$. This is a candidate point to be added to a region. A sorted list is useful in the calculations of the algorithm. Let SSL be a linear list of the pixels p'. The points in the SSL list are sorted in order of increasing values of $\delta(p')$ The ones with the smallest values are in the front of the list.

The algorithm is given below.

algorithm
1. Label the pixels in the atomic regions with their initial labeling.
2. Compute the pixels in T and put these pixels in SSL a list of pixels which are sorted according to $\delta(p')$. The p with the smallest values of $\delta(p')$ are first on the list.
3. Remove the first pixel p from SSL while the SSL list is not empty.
4. If all neighbors of p that have already been labeled have the same label, then set p to this label. Next update the mean of the affected region and add the neighbors of p that are not already labeled or already on the SSL to the SSL list according to their δ values.
5. else label p as a boundary pixel.
6. iterate the process by going back to step 3.

In step 4 one may want to update the distances in the SSL list according to the new point that has been added to the region.

The next example illustrates the method.



The atomic regions are {(0,3)} with a mean of 1 and label a and {(3,1)} with a mean of 7 and label b. The boundary pixels are:
T = { (0,2),(1,2),(1,3),(2,0),(3,0),(2,1),(2,2),(3,2) }
The distances are

T	(0,2),	(1,2),	(1,3),	(2,0),	(3,0),	(2,1),	(2,2),	(3,2)
δ =	0	2	0	1	0	1	2	1

and the sorted list is:

SSL=

T	(0,2)	(1,3)	(3,0)	(2,0)	(2,1)	(3,2)	(1,2)	(2,2)
δ =	0	0	0	1	1	1	2	2

Consider the first pixel (0,2), its label becomes "a". The updated atomic regions now become {(0,3),(0,2) } and {(3,1)}. The new neighbors to be put on T are (0,1),(1,1) with δ's of 1 and 2 respectively. The SSL now becomes

SSL=

T	(1,3)	(3,0)	(2,0)	(2,1)	(3,2)	(0,1)	(1,2)	(2,2)	(1,1)
δ =	0	0	1	1	1	1	2	2	2

The arrows indicate the new items on the list. One could at this step have updated all these distances. The new first pixel is (1,3) whose label becomes "a". The updated atomic regions are now {(0,3),(0,2),(1,3) } and {(3,1)} with means of 1 and 7 respectively. The new neighbor is (2,3) with a δ of 2. The SSL becomes

SSL=

T	(3,0)	(2,0)	(2,1)	(3,2)	(0,1)	(1,2)	(2,2)	(1,1)	(2,3)
δ =	0	1	1	1	1	2	2	2	2

One continues in this process and arrives at a first pixel of (0,1) which is labeled "a". The updated atomic regions become {(0,3),(0,2),(1,3),(0,1)}
and {(3,1),(3,0),(2,0),(2,1),(3,2)} with means of 1.25 and 6.4. The new neighbor is (0,0) with a δ of .75. This pixel immediately comes off the SSL as the new first pixel that is labeled "a". The updated atomic regions become
{(0,3),(0,2),(1,3),(0,1),(0,0)}
and {(3,1),(3,0),(2,0),(2,1),(3,2)} with means of 1.8 and 6.4. There are no new neighbors. The SSL is

SSL=	T	(1,2)	(2,2)	(1,1)	(2,3)	(3,3)	(1,0)
	δ =	2	2	2	2	2.4	3

The region labeling at this point is given by the following figure.

	3	1	1	3	4
	2	1	3	5	$\boxed{6}$
y	1	2	3	$\boxed{6}$	$\boxed{7}$
	0	2	3	$\boxed{6}$	$\boxed{7}$
		0	1	2	3
			x		

region a

region b

first pixel

The first pixel is (1,2) which is labeled boundary. The next first pixel is (2,2) which is label b boundary. The updated atomic regions become
{(0,3),(0,2),(1,3),(0,1),(0,0)}
and {(3,1),(3,0),(2,0),(2,1),(3,2)} with means 1.8 and 6.4. There are no new neighbors. The first pixel is (1,1) which is label boundary. The next first pixel is (2,3) which is also boundary. The first pixel now becomes (3,3) which is labeled b. The updated atomic regions become
{(0,3),(0,2),(1,3),(0,1),(0,0)}
and {(3,1),(3,0),(2,0),(2,1),(3,2),(3,3)} with means 1.8 and 6. There are no new neighbors. The SSL is

SSL=	T	(1,0)
	δ =	3

The pixel (1,0) is labeled boundary.

	3	1	1	3	$\boxed{4}$
	2	1	3	5	$\boxed{6}$
y	1	2	3	$\boxed{6}$	$\boxed{7}$
	0	2	3	$\boxed{6}$	$\boxed{7}$
		0	1	2	3
			x		

region a

region b

Final Segmentation

1.4 Graphical Methods.

Graph structures can be useful in region merging methods. One can represent the regions as nodes on a graph. There is an arc between nodes if they are adjacent. The weights on the arc can be the uniformity measure between the two regions. If the uniformity measure is high of an arc between two regions, this means the combined regions would be uniform. The nodes could be individual pixels if one is starting at a very low level. They could be atomic regions if one has obtained an initial segmentation. For each node, calculate the uniformity measure for each arc. If the highest uniformity measure for each arc is above a threshold then merge the adjacent regions. Compute the uniformity measure and the links for the newly created node [Eggleston, 1998]. This is demonstrated in the following figure.

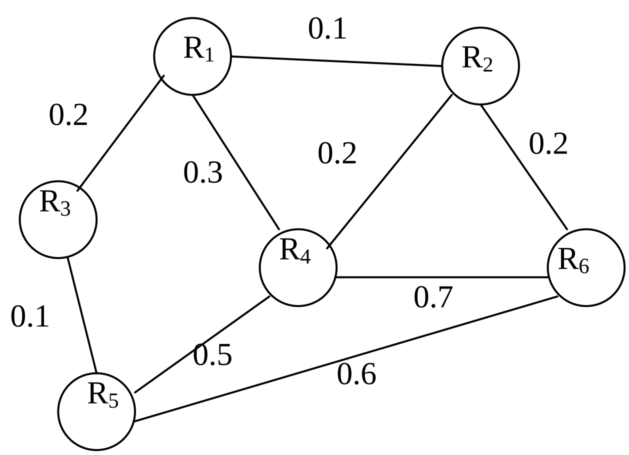


Figure 4. Graph Representation of Regions

Merge the regions if the uniformity function is greater than .45.

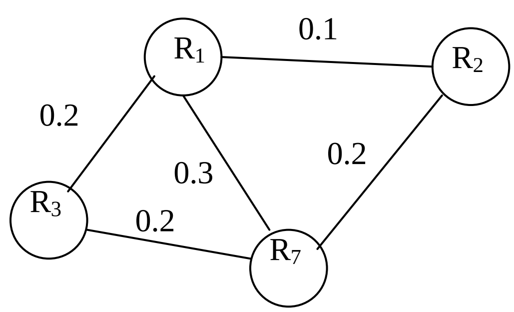


Figure 5. Graph of Merged Regions

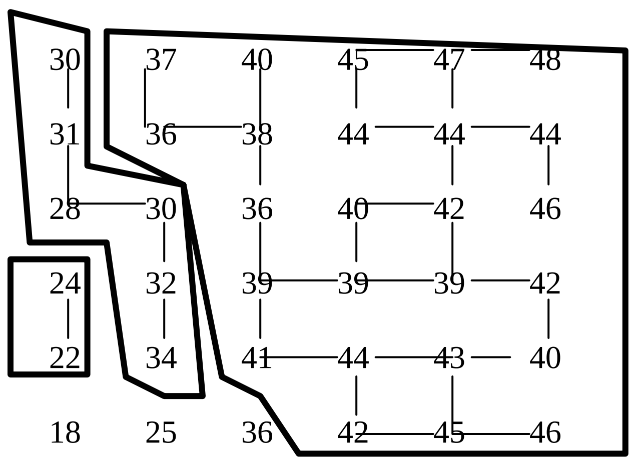
1.5 Graph Approaches Starting with Pixels.

In the following methods we consider the pixels to be nodes forming a graph. We then form the arcs of the graph to form a region connectivity graph. The pixels are in the same region if there is a path between them on the region connectivity graph. The simplest method forms arcs between the nodes according to properties of each pixel. Neighboring pixels whose properties are similar are joined by an arc [Haralick and Shapiro, 1985]. The regions are then formed as the maximal sets of pixels belonging to a connected component of the resulting region connectivity graph.

The most obvious measure of similarity would be the gray-level difference between the pixels. If p and p' are neighboring pixels then connect them with an arc if $\|g(p)-g(p')\| < a$ where a is a parameter. One could estimate the parameter 'a' by averaging the gray-level differences over the image or over a neighborhood of p. The following example demonstrates the process.

30	37	40	45	47	48
31	36	38	44	44	44
28	30	36	40	42	46
24	32	39	39	39	42
22	34	41	44	43	40
18	25	36	42	45	46

image data



regions
formed a=4

One can consider neighborhood properties instead of the properties of a single pixel in the region formation method. A property vector is assigned to each pixel where the property vector depends upon a neighborhood $N(x, y)$ of (x,y). There are many properties one could use such as average gray-level in the neighborhood. One property could be an edge property, E(p) where p is a pixel. E(p) is True if p is an edge point else it is False. Pixels p and p' would be connected by an arc if they are neighbors and both of them are not edge pixels, i.e. E(p) and E(p') are both False. The regions are the connected components of the graph. With this method, problems occur if the edge function, E, produces gaps in the boundary of regions. The regions would then merge together. Hence, the function is dependent upon the method for measuring edge properties.

Similarity Method

Assume that one has developed a measure of similarity between pixels p and p', S(p,p') arc [Haralich and Shapiro, 1985]. Then for all p' in N(p), compute the similarity between p and p', S(p,p'), and put the n most similar on a list L(p). Do this for every pixel p. An arc is drawn between p and p' on the region connectivity graph, if p is in L(p'), p' is in L(p) and the number of common neighbors is greater than parameter 'a'. The number of shared neighbors must be high. Observe that the neighborhood could be larger than the immediate neighbors of a pixel. For example, one could put the two most similar neighbors on the list L(p) for point p. Then draw an arc between p,p' if L(p) contains p' and L(p') contains p and they share one neighbor. One can devise different measure of similarity which determine the process. One is difference in gray-levels. Others could be based on neighborhood properties. For example,

$g(p) = i, \quad a = \dfrac{\sum_{\hat{p}} g(\hat{p})}{N}$ where $\hat{p}$ is in N_p a neighborhood of p and N is the number of points in the neighborhood,

$g(p') = j, \quad b = \dfrac{\sum_{\hat{p}} g(\hat{p})}{N}$ where $\hat{p}$ is in $N_{p'}$ a neighborhood of p' and N is the number of points in the neighborhood, and the similarity measure is

S(p,p')= $w_1(i-j)^2 + w_2(i-b)^2 + w_3(j-a)^2$. The w's are weights to be set.

The following example demonstrates the method. In the example, we use 4-connected neighbors and we put the two most similar on the list. The measure of similarity S(p,p') is $\|g(p) - g(p')\| \leq 5$. The number of shared neighbors for this example is one.

y \ x	0	1	2	3	4	5
5	30	37	40	45	47	48
4	31	36	38	44	44	44
3	28	30	36	40	42	46
2	24	32	39	39	39	42
1	22	34	41	44	43	40
0	18	25	36	42	45	46

image data

The following figure shows some of the data with the two most similar pixels circled. This determines which pixels are connected by an arc.

40 45 47

36 38 44 38 44 44 44 44 44

36 40 40

44

38 44

30 36 40 36 40 42 40 42 46

39 39 39

circles are
the most
similar

The next figure shows the resulting graph structure for region connectivity. The regions are the connected nodes of the graph.

30 37 40 45 47 48

31 36 38 44 44 44

28 30 36 40 42 46

24 32 39 39 39 42

22 34 41 44 43 40

18 25 36 42 45 46

graph structure

Figure 0-6. Region Connectivity Graph

1.6 Graph Method with Clustering

In this method we again assume we have nodes on a graph that correspond to regions of the image. The regions could be individual pixels. The minimum spanning tree (MST) is a graph structure that can be used to form regions [Nadler and Smith, 1998, pp.137]. The regions we are seeking are the nodes on the graph that form clusters. The distances between the nodes again reflect the "distance" between the regions. The MST can be used with any connected graph to form clusters [Duda and Hart, 1973, pp.238]. To form the MST, one starts with a node and connects it to the nearest node that has not already been connected. The process continues always selecting the next node that is closest to one of the nodes already in the MST. The MST Spans the point set with no loops (a tree). The following figure shows a MST for a set of nodes. The length of the arc gives the distance between nodes.

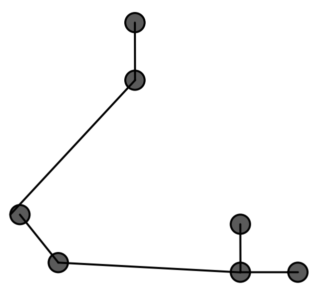


Figure 7. Minimum Spanning Tree

If we consider the pixels as nodes on a graph we can use the method to form regions. Consider the following small image as an example calculation. The image data are given in the following figure.

1	7	6	9
2	4	6	8
1	2	3	4
1	1	1	1

Figure 8. Image Data for Graph Structure

The next figure shows the graph structure using the absolute value of the difference of the gray-levels as the measure of uniformity between two pixels.

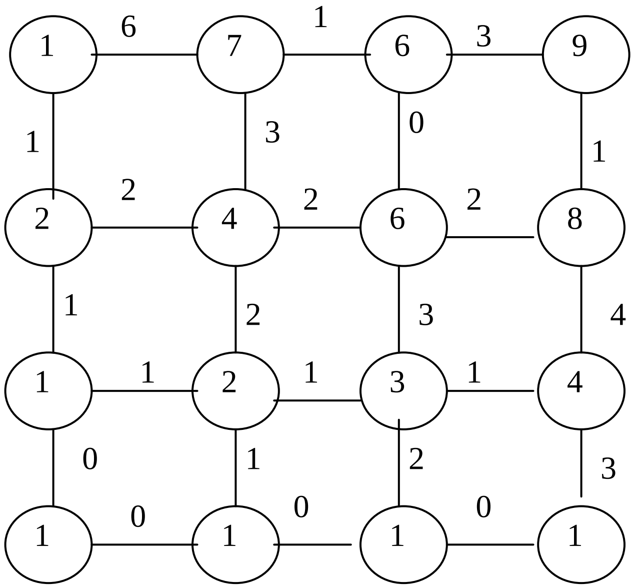


Figure 9. Graph Obtained from Image

The following figure shows the MST for the graph.

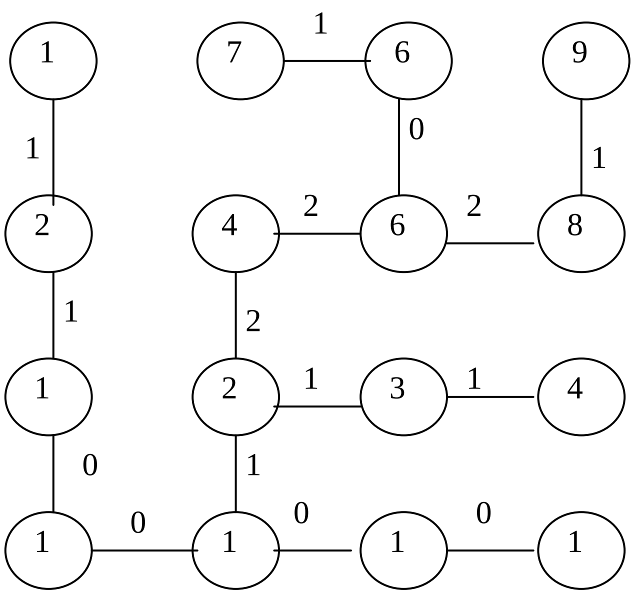


Figure 10. MST Obtained from Graph

One can partition the image into two regions by breaking the largest link (namely one of the 2 links) on the MST.

1.7 Multispectral Region Growing Method.

It is often the case that we have more than one band of image data. Color imagery and a wide variety of remotely sensed data fits into this pattern. Suppose in this case we have three bands with image functions $g_1(p)$, $g_2(p)$, and $g_3(p)$ where p is a pixel. Consider the criterion for putting a pixel p into a neighboring region R_k [Levine, 1985, pp.390]. For each region we compute the means and standard deviations for each band of data.

$\mu_k(i) = \frac{1}{\#(R_k)} \sum_{p \text{ in } R_k} g_i(p)$ and $\sigma_i^2(i) = \frac{1}{\#(R_k)} \sum_{p \in R_k} \left[g_i(p) - \mu_k(i) \right]^2$ where R_k denotes the kth region and g_i denotes the ith band.

First we compute the mean and variance of R_k assuming that p is added to R_k. The following equations can be used to compute the updated mean and variance. $\mu_k'(i) = \frac{1}{n+1}\left[g_i(p) + n\mu_k(i) \right]$ and $\sigma_k'^2(i) = \frac{1}{n+1}\left[n\sigma_k^2 + \left(\frac{n}{n+1} \right)\left[g_i(p) - \mu_k(i) \right]^2 \right]$ where $n=\#(R_k)$ the number of pixels in R_k. The i's refer to the different bands. The strategy is to add pixel p to the region if the uniformity of the region is preserved.

Consider the ratio of the variance to the mean $\frac{\sigma'_k(i)}{\mu_k'(i)}$. For a non uniform region, $\frac{\sigma'_k(i)}{\mu_k'(i)}$ is a large number. Therefore, the quantity $a_k(i) = \max\left\{ \left[1 - \frac{\sigma'_k(i)}{\mu_k'(i)} \right] b, 0 \right\}$ is small for non uniform regions. The term 'b' is a parameter to be set. The difference $\Delta\mu_k(i) = |g_i(p) - \mu_k(i)|$ for i=1,2,3 and region R_k is used for comparison to $a_k(i)$. If for each i , $\Delta\mu_k(i) \le a_k(i)$ then put p in R_k. For non uniform regions, the term $a_k(i) \approx 0$ and the region will not grow. If p cannot be added to any region, then start a new region. Consider the following example which demonstrates these calculations.

	5	30	37	40	45	47	48	R_1
	4	31	36	38	44	44	44	
	3	28	30	36	40	42	46	image data
y	2	24	32	39	39	39	42	
	1	22	34	41	44	43	40	
	0	18	25	36	42	45	46	
		0	1	2	3	4	5	
				x				

The mean of R_1 is μ_k = 45.43 and the standard deviation is σ_k = 1.5. Consider adding the pixel with value 42 to R_1. Then $\mu_k' = 45$, $\sigma_k' = 1.8$ and

$a_k(i) = \max\left\{ \left[1 - \frac{\sigma'_k(i)}{\mu_k'(i)} \right] b, 0 \right\} = \max\left\{ \left[1 - \frac{1.8}{45} \right] b, 0 \right\} = .96b$. And

$\Delta\mu_k(i) = |g_i(p) - \mu_k(i)| = |42 - 45.43| = 3.43$. The equation $\Delta\mu_k(i) \le a_k(i) = 3.34 \le .96b$. The setting of parameter b will determine if this test is met.

Consider adding the pixel with value 38 to R$_1$. Then $\mu_k^{'} = 44.5$, $\sigma_k^{'} = 2.8$ and

$$a_k(i) = \max\left\{\left[1 - \frac{\sigma'_k(i)}{\mu_k^{'}(i)}\right]b, 0\right\} = \max\left\{\left[1 - \frac{2.8}{44.5}\right]b, 0\right\} = .94b$$. Now

$\Delta\mu_k(i) = |g_i(p) - \mu_k(i)| = |38 - 44.5| = 6.5$. The equation becomes $\Delta\mu_k(i) \le a_k(i) = 6.5 \le .94b$. The setting of parameter b will again determine if the pixel goes in the region. One can observe that the pixel with the value 42 would go in R$_1$ before the pixel with the value 38 which is desirable.