Normalized Graph Cuts

BASED ON ‘NORMALIZED CUTS AND IMAGE SEGMENTATION’

BY JIANBO SHI AND JITENDRA MALIK

Smit Patel

Image Segmentation

- Image segmentation is a grouping technique used for object grouping in a given image.
- It is a way of dividing an image into different regions, which posses similar properties such as intensity, texture, colors, features etc.

Perceptual Grouping

- Organizing image primitives into higher level primitives, thus extracting the global impression of the image, rather than focusing on local features in the given image data

Original image

Segmented image

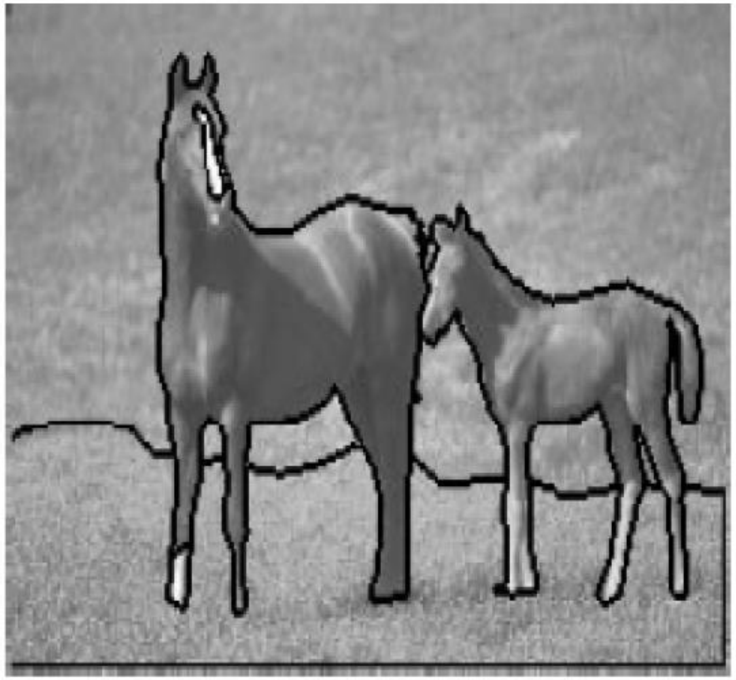
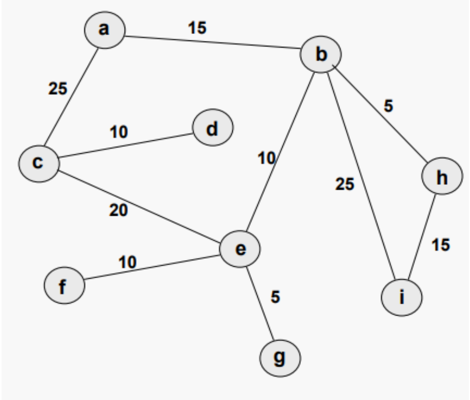
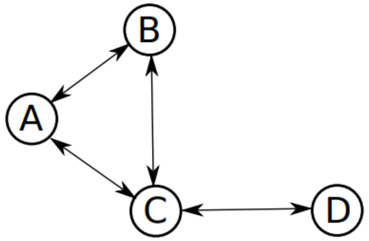


Image as a Graph

- Graph Terminology
  - A graph is a set of nodes V and edges E that connect various nodes
  - Represented as G = {V, E}
    - Where, V = set of nodes called as Vertices of G
    - E = edges connecting different nodes, called as Edges of G
  - Each edge may be assigned a numerical value, called weight, often represented as the ‘cost’ of the respective edge.
  - Weighted graph is the one where all the edges are assigned some weights.
  - Directed graph is where all the nodes are ordered with respect to their weights.
  - Connected graph is where all the nodes are connected to one another.

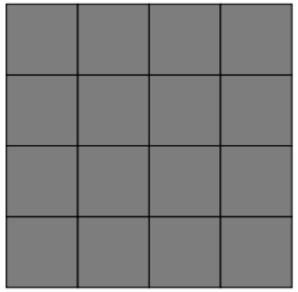
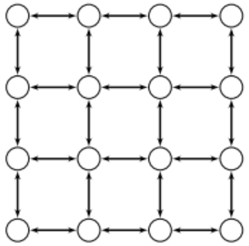
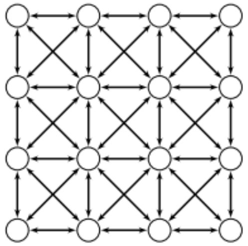
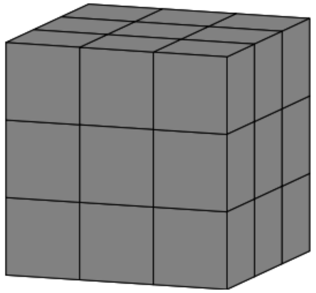
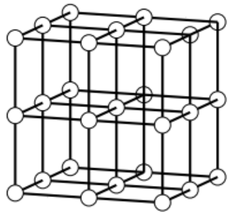
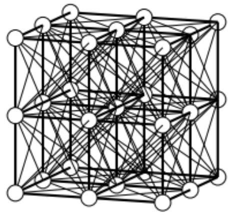




Graph with V = {A,B,C,D}
and edges E = {$e_{A,B}$, $e_{A,C}$, $e_{B,C}$, $e_{C,D}$}

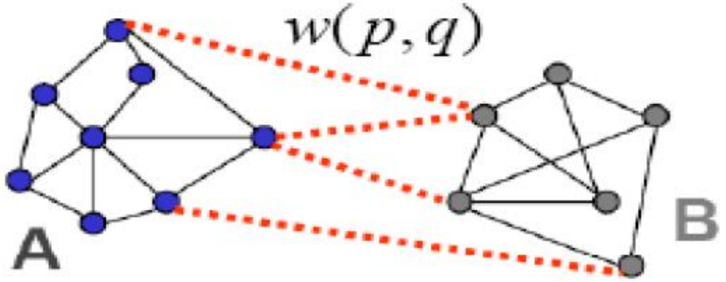
Image as a Graph (cont.)

- Each pixel represents a node
- Every pixel is connected to its neighbors using Edges
  - Edges can be 4-connected, 8-connected (for 2D)
  - Edges can be 6-connected, 26-connected (for 3D)

Graph Cuts

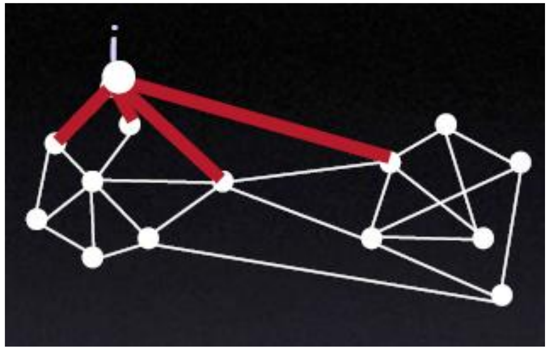
- In grouping, a weighted graph can be split into disjoint sets of nodes
  - The association between nodes within the same class is high
  - The disassociation between nodes from different class is low
- A graph cut is the technique to group different nodes
  - The degree of dissimilarity between these two groups is computed as total weight of the edges removed, in order to create multiple sets of nodes
  - Often referred as the ‘cost’ of the cut

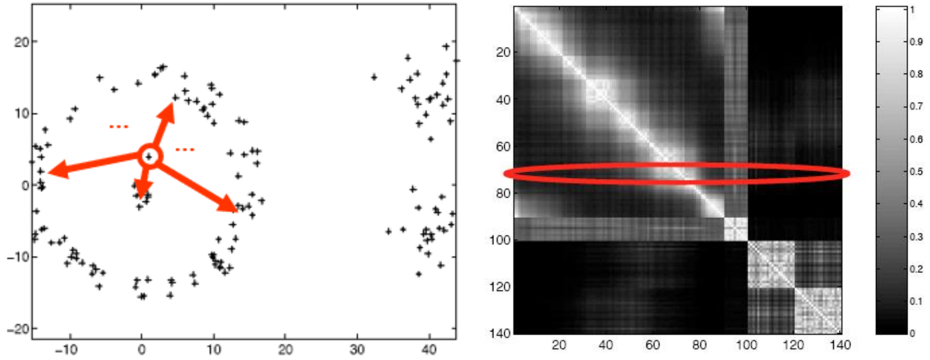


$$cut(A, B) = \sum_{p \in A, q \in B} w(p, q)$$

Graph Terminology

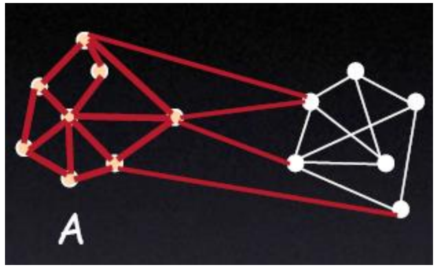
- Degree of Node : $d_i = \sum_j w_{i,j}$

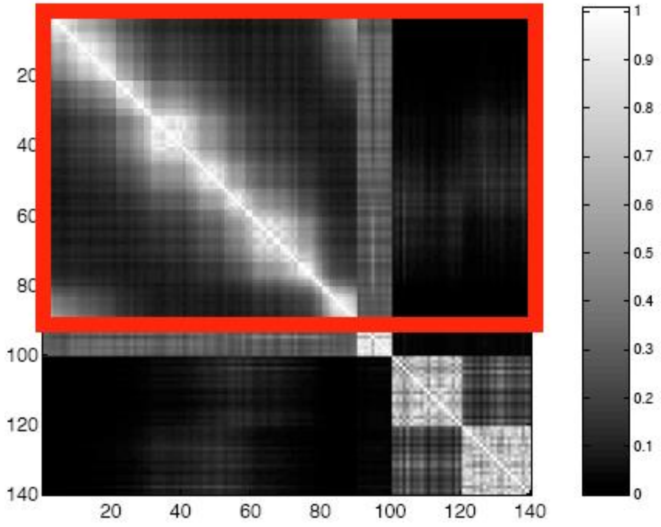




- Affinity Matrix : $W(i,j) = aff(i,j)$

- Volume of a set : $vol(A) = \sum_{i \in A} d_i, A \subseteq V$



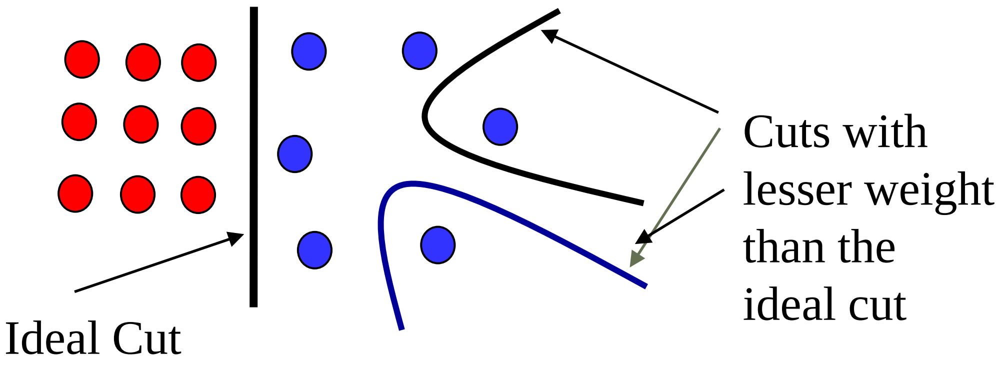


Minimum-Cuts Graph Partition

- To extract a 'good cut'
  - Simply compute the minimum cost cut in the graph

$$\min cut(A,B) = \min_{A,B} \sum_{u \in A, v \in B} w(u,v)$$

  - To partition into k-subgraphs, recursively find the minimum cuts that bisect the existing node segments
- Can lead to impractical segments when there are isolated nodes in the graph
  - Problem : Weights are directly proportional to the number of edges in the cut



Normalized Cut

- Computes the cut cost as a fraction of the total edge connections to all nodes in the graph.
- Given by:

$$Ncut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)}$$

  - Where ***assoc(A,V)*** defines the total weights of connection from nodes A to all nodes in the graph (V).

$$vol(A) = assoc(A,V) = \sum_{u \in A, t \in V} w(u,t)$$

  - Smallest ***Ncut(A,B)*** are selected to partition the image into two partitions.
- For the isolated nodes, value of *Ncut* will no longer be small, as the *Cut* value will almost always be a high percentage of the total connection from the isolated node to all other nodes.

Normalized Cut (cont.)

- **Normalized Association:**

$$Nassoc(A,B) = \frac{assoc(A,A)}{assoc(A,V)} + \frac{assoc(B,B)}{assoc(B,V)}$$

  - Defines how tightly on average within the cluster are connected to each others

- Problem of minimizing ***Ncut(A,B)*** is same as maximizing the ***Nassoc(A,B)***, since they are related to each other

$$Ncut(A,B) = 2 - Nassoc(A,B)$$

  - Which makes sense, as minimizing the disassociation between the groups and maximizing the association within the group is identiacal.

Normalized Cut (cont.)

- Computation of minimum ***Ncut :***
  - Convert ***Ncut*** equation into metrices using following method:

$$Ncut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(B,A)}{assoc(B,V)}$$
$$= \frac{\sum_{(\boldsymbol{x}_i>0,\boldsymbol{x}_j<0)} -w_{ij}\boldsymbol{x}_i\boldsymbol{x}_j}{\sum_{\boldsymbol{x}_i>0} \boldsymbol{d}_i} + \frac{\sum_{(\boldsymbol{x}_i<0,\boldsymbol{x}_j>0)} -w_{ij}\boldsymbol{x}_i\boldsymbol{x}_j}{\sum_{\boldsymbol{x}_i<0} \boldsymbol{d}_i}$$

  - Where, x is an N-dimensional indicator vector, such that x_i = 1 if i belongs to A, x_i = -1 of i belongs to B and $\boldsymbol{d}(\mathrm{i}) = \sum_j w(i,j)$
  - Let D be an NxN diagonal matrix, with *d(i)* on its diagonal. (Degree matrix)
  - Let W be an NxN symmetrical matrix with $W(i,j) = w_{i,j}$ (Affinity matrix)

Normalized Cut (cont.)

$$D = \begin{bmatrix} \sum_j w(1,j) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sum_j w(N,j) \end{bmatrix} \qquad W = \begin{bmatrix} w(1,1) & \cdots & w(1,N) \\ \vdots & \ddots & \vdots \\ w(N,1) & \cdots & w(N,N) \end{bmatrix}$$

- After simplifying, the ***Ncut*** can be represented as following:

$$\min_x Ncut(x) = \min_y \frac{y^T (D-W) y}{y^T D y}$$

Subject to: $y^T D 1 = 0$

  - This result can be solved by solving generalized eigenvalue equation: $(D-W)y = \lambda D y$
  - Note, the first eigenvector is $y_0 = 1$, with eigenvalue 0 (we discard it)
  - We pick the second smallest eigenvector, which is the solution to our problem.

Normalized Cut (cont.)

Algorithm:

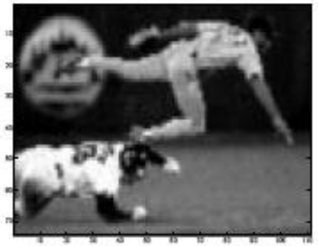
- Define a weight function (based on the similarity between pixels)
- Compute affinity matrix (W) and degree matrix (D)
- Solve eigenvector equation $(D - W)y = \lambda D y$
- Use the eigenvector with the second smallest value to bi-partition the graph
- Repeat using subsequent smaller eigenvalues to continue partitioning, until the stopping criteria is satisfied.

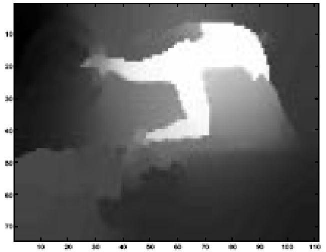
Note : The memory requirement for the computation is really high. ($O(n^3)$). But since the precision requirements are low, W is very sparse and only few eigenvectors are required, the eigenvectors can be extracted very fast using Lanczos algorithm (O(mn), where m is the number of steps required for Lanczos to converge).

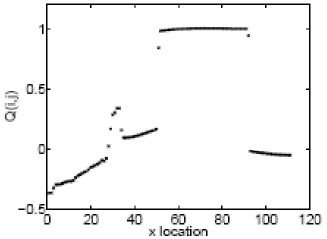
Normalized Cut (cont.)

Discretization:

- Since the eigenvector can take continuous values, we need to define a threshold to binarize







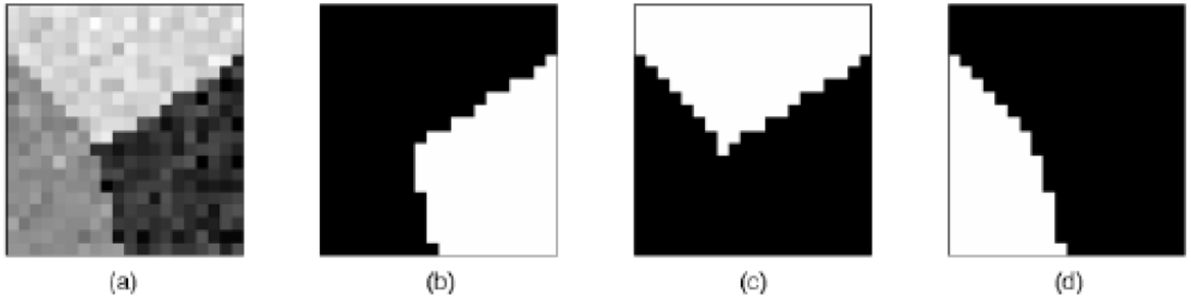
How to choose the splitting point:

- Pick a constant value (0.5)
- Pick the median value as splitting point
- Look for the splitting point that has minimum Ncut value:
  - Choose n possible splitting points
  - Compute Ncut
  - Pick n with the minimum corresponding Ncut

K-Normalized Cuts

- Recursive 2-way Ncut is slow
- We can use more eigenvectors to re-partition. However, not all eigenvectors are useful for partition

$$Ncut_k = \frac{cut(A_1, V - A_1)}{assoc(A_1, V)} + \frac{cut(A_2, V - A_2)}{assoc(A_2, V)} + \ldots + \frac{cut(A_k, V - A_k)}{assoc(A_k, V)}$$



(a) A synthetic image showing three image patches forming a junction and Gaussian noise with σ = 0.1 is added. (b-d) top three components of the partition

Pixel Similarity Functions

- The edge weight can be selected based on the type of the image:
  - Intensity Image:

$$W(i, j) = e^{\frac{-\left\| I_{(i)} - I_{(j)} \right\|_2^2}{\sigma_I^2}}$$

  - Scatter plots :

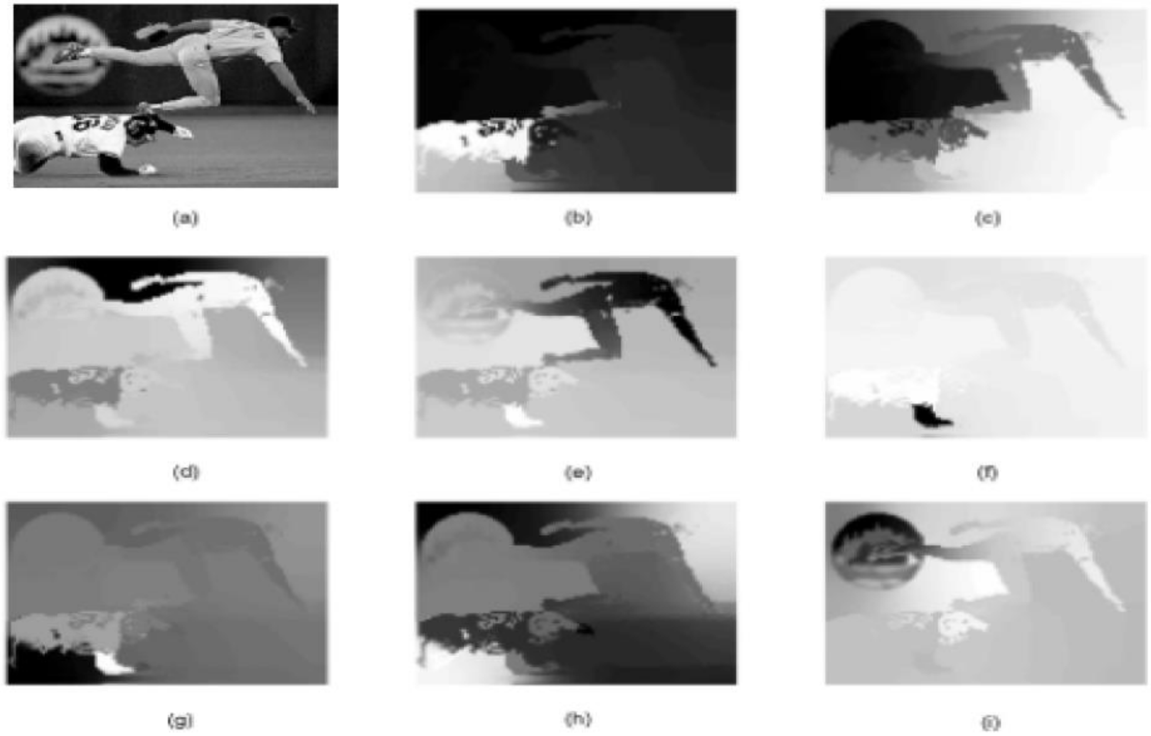
$$W(i, j) = e^{\frac{-\left\| X_{(i)} - X_{(j)} \right\|_2^2}{\sigma_X^2}}$$

  - Colors or textures:

$$W(i, j) = e^{\frac{-\left\| c_{(i)} - c_{(j)} \right\|_2^2}{\sigma_c^2}}$$

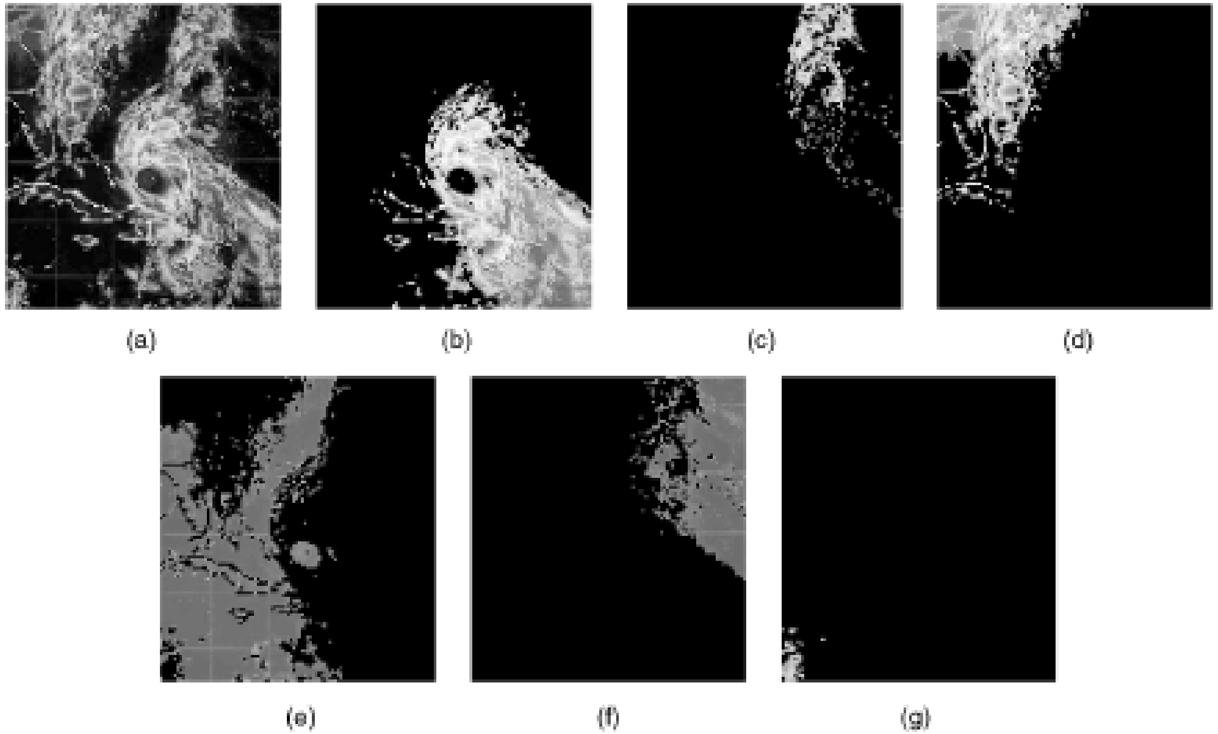
Here, c(i) is a vector of filter output

Results



a) A 80x100 baseball scene, with feature as image intensity, (b-h) components of partition with Ncut value less than 0.04, Parameter setting: σ_i=0.01, σ_X=4 and r=5

Results



(a) Shows a 126 x 106 weather radar image
(b) to (g) shows the components of the partition with *Ncut* value less than 0.08 corresponding to the eigenvectors from 2nd smallest to 7th smallest values
Parameters : $\sigma_I = 0.005,\ \sigma_x = 15.0,\ r = 10$

Questions

Thank You