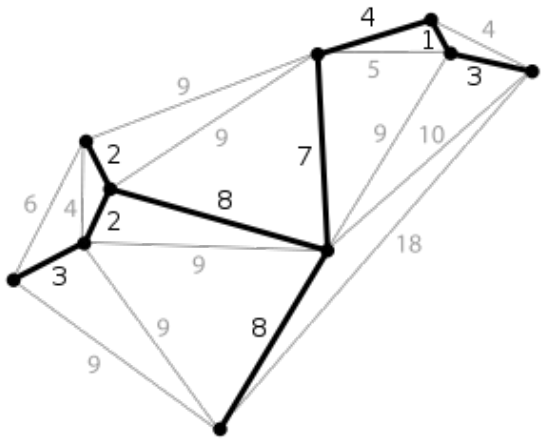
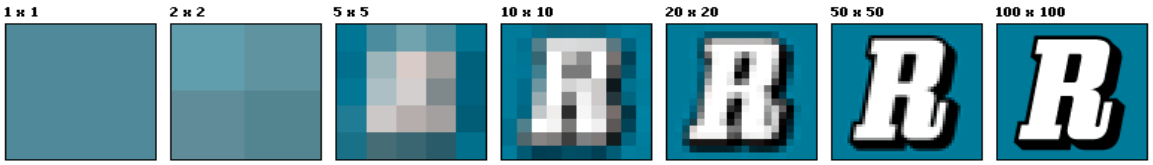
An Alternative to the Region Growing using Minimum Spanning Tree Clustering for Digital Image Segmentation

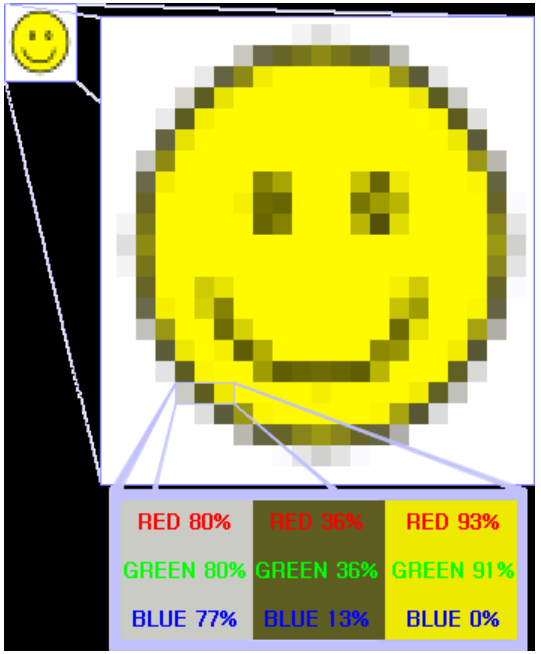


Outline

- Motivation
  - Segmentation
  - Region definition
- Growing Regions in Digital Images
  - Algorithm Description
  - Weaknesses
- Minimum Spanning Trees
- Pruning
  - Edge based pruning
  - Cluster based pruning

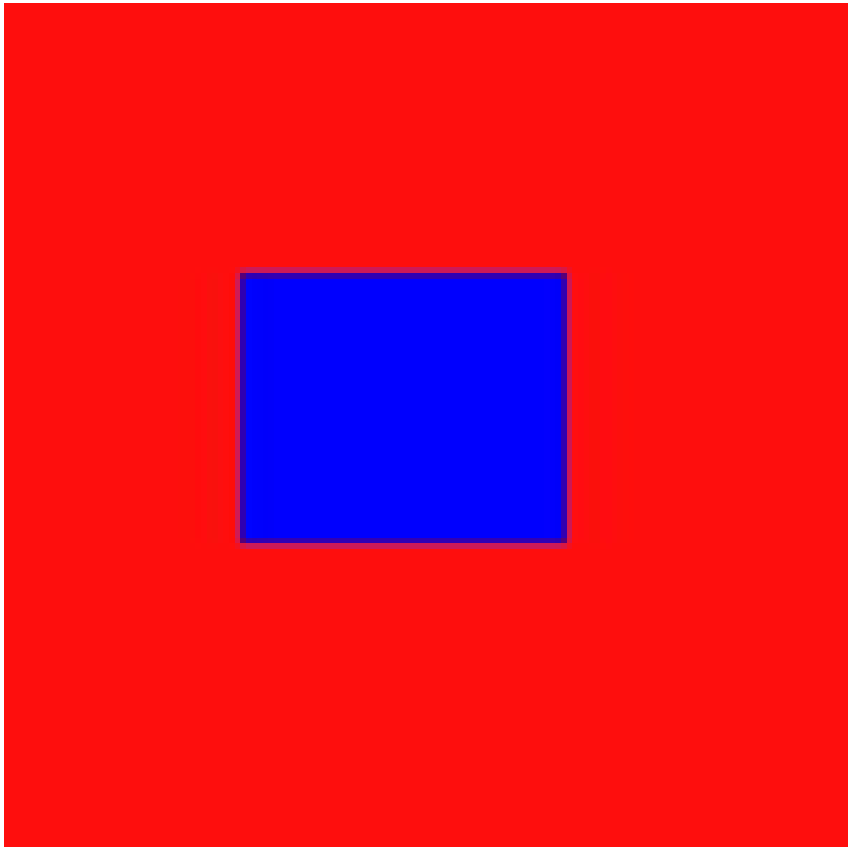
Digital Images





Motivation: Segmentation

- Segmentation is the first step in image analysis
  - Subdivides an image into constituent parts
  - Used to find objects of interest
- Goal: Partition a digital image into sets of connected pixels with similar color properties (regions)



The Region Growing Algorithm

```
RegionGrowth( x, y, region ) {
    region = c( region, pixel(x,y) )
    If color(pixel(x+1,y)) ~ color(pixel(x,y))
        region = RegionGrowth(x+1,y)
    If color(pixel(x-1,y)) ~ color(pixel(x,y))
        region = RegionGrowth(x-1,y)
    If color(pixel(x,y+1)) ~ color(pixel(x,y))
        region = RegionGrowth(x,y+1)
    If color(pixel(x,y-1)) ~ color(pixel(x,y-1))
        region = RegionGrowth(x+1,y)
    return(region)
}
```

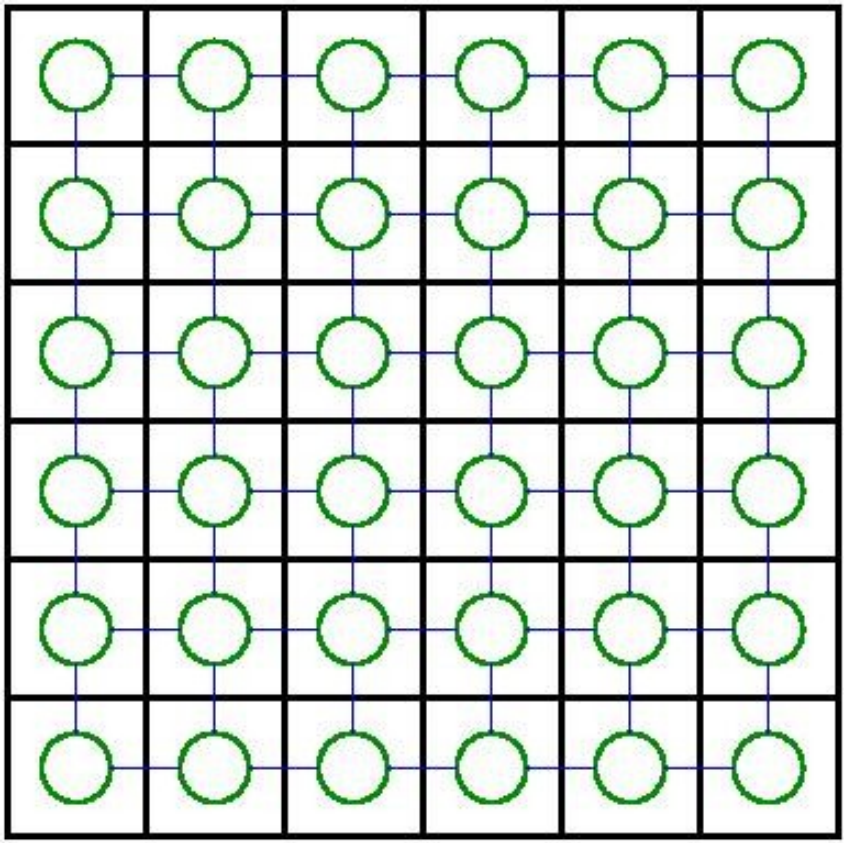
Example of Region Growth

Problems with Region Growth

- Need to start with an appropriate initial point
  - Assumes that you already know where an interesting region is
  - Keep performing RG until all points are contained by a region
- Sensitive to noise
  - Pick a tolerance?
  - Smooth the image before but how and by how much?
- Sensitive to shading
- Algorithm is slow and does not work well in R (recursion troubles)

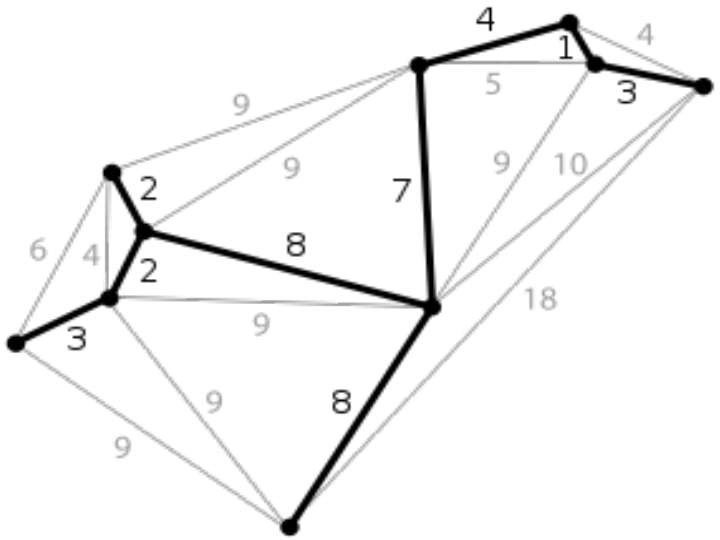
The Lattice Graph

- Boxes represent pixel locations in the image
- Circle is a vertex corresponding to a pixel
- Edges appear between vertices with adjacent pixels
- Edges are weighted by the difference between corresponding pixels



Minimum Spanning Tree

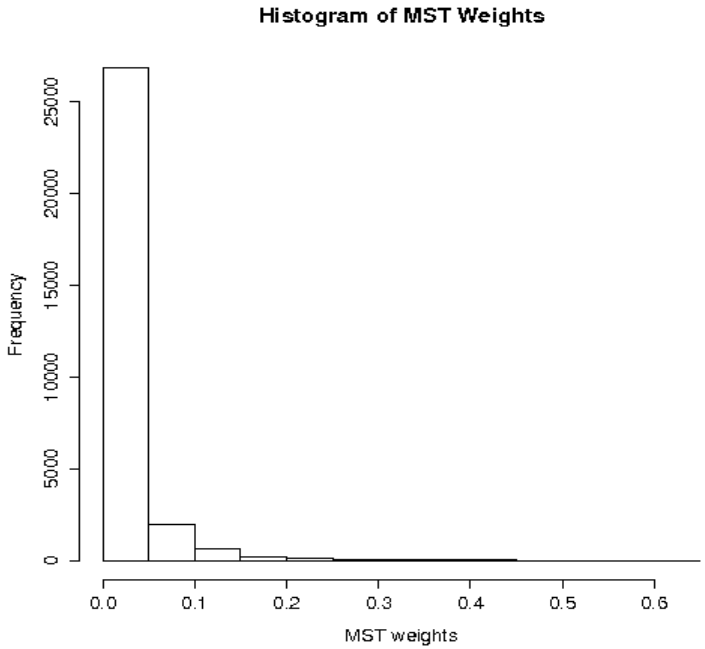
- Spanning Tree
  - Subgraph of original graph
  - Tree
  - Connects all vertices
- MST
  - Spanning tree where weights are minimized
- Algorithms
  - Kruskal O(E log E)
  - Prim O(E log V)



The Idea behind MST regions

- If two pixels are adjacent but lattice-graph edge weight is big, they are probably not in the same region
- MST algorithm will remove these edges with big weights
- Low-weight edges correspond to connections between pixels in the same region. Pixels in the same region are close in the MST.
- Big edge weights in the MST correspond the boundaries between regions. Removing these edges creates clusters of similar pixels (regions).
- How should we remove these edges?

Edge-Based Pruning

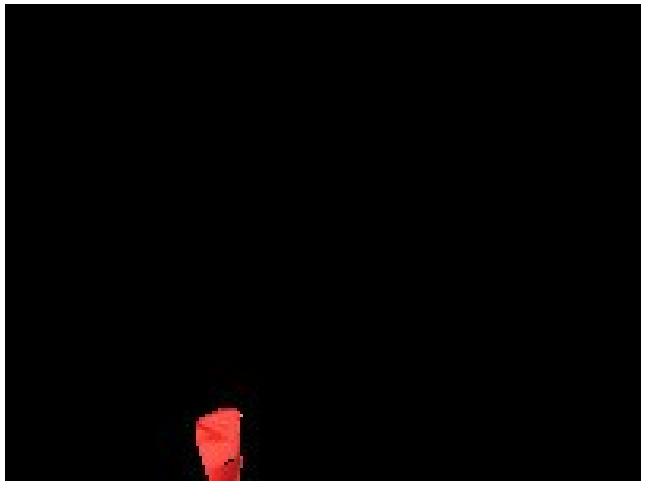


$breaks
[1] 0.00 0.05 0.10 0.15 0.20 0.25 0.30 0.35 0.40 0.45 0.50 0.55 0.60 0.65

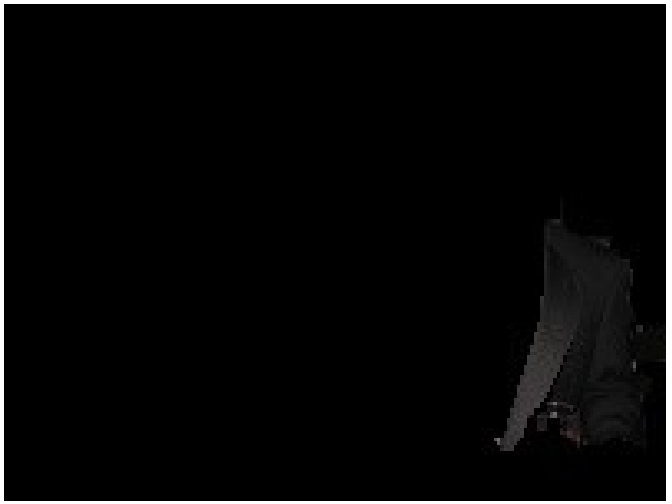
$counts
[1] 26853 1983 617 216 124 77 50 34 17 14 8 4 2

Pruning the Largest 5% of Edges

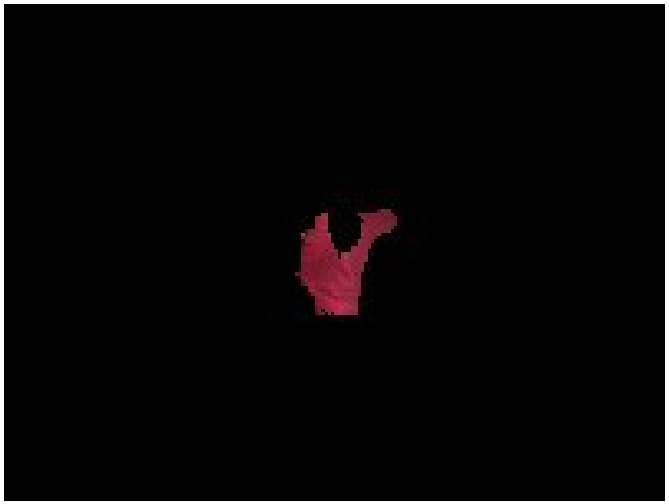
Pruning the Largest 5% of Edges



Pruning the Largest 15% of Edges



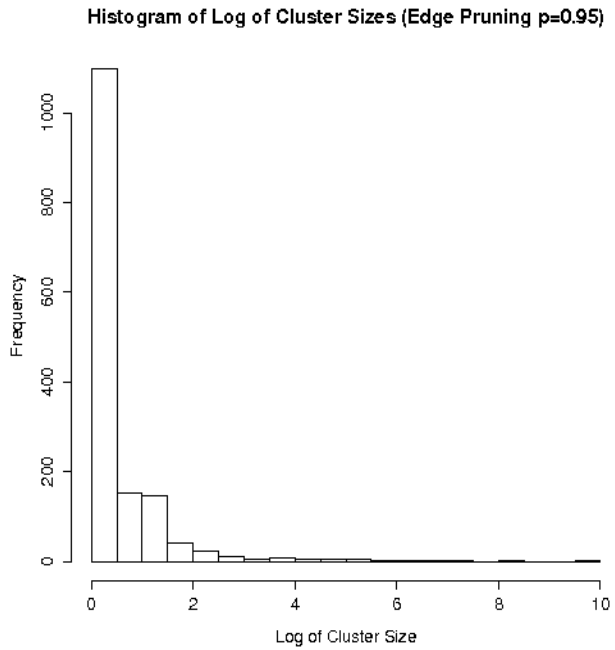
Pruning the Largest 15% of Edges

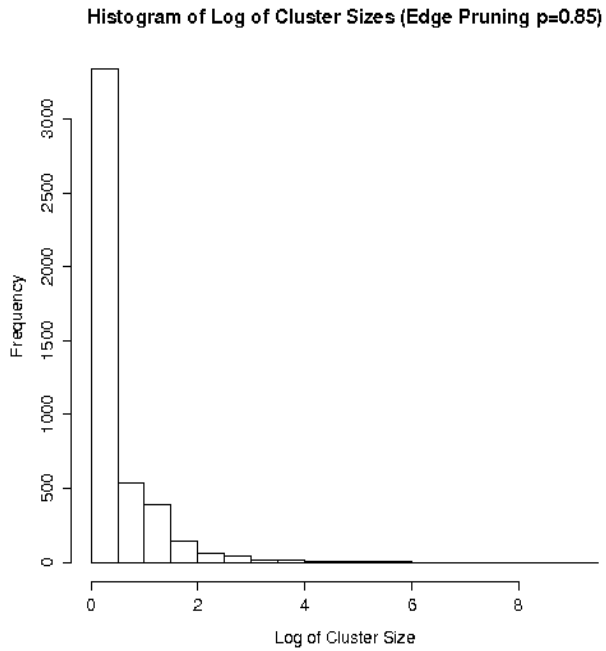


Edge Based Pruning

- Pros
  - Simple Interpretation in terms of MST
  - Easy to compute
- Cons
  - Relies on an arbitrary p-value to prune the tree
  - Difficult to interpret in terms of the image
  - Does not take into account the size of the clusters when pruning

Cluster Based Pruning





- Most of the clusters found are very small
- Increasing the p-value decreases the frequency of the smaller clusters and decreases the size of the largest clusters

Edge Based Pruning

- Start with the fully connected MST and iteratively remove the smallest edge, creating new clusters
- Very small clusters are “Runts” (Hartigan and Mohanty, 1992)
  - Runts are not part of a cluster
  - Can be ignored
- If the number of clusters is known, Runt Pruning (Stuetzle, 2003) can be used.
- When finding regions in digital images, the number of regions is not known beforehand.

Cluster Sizes

```
[1] 20
    1     2 29983
   15     1     1
[1] 100
    1     2     3     4     5     6     7     8    17 29610
  240    17     4     5     3     5     2     1     1     1


[1] 500
   1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
4791  781  367  151   95   58   35   29   23   15   17    7   10    5    4    4
  17   18   19   20   21   22   23   24   25   26   27   28   29   30   31   32
   9    3    5    1    2    4    4    1    1    2    2    5    2    1    2    5
  34   35   36   37   38   39   40   43   47   48   49   50   59   61   63   66
   3    2    1    2    1    2    1    1    1    2    2    2    1    1    1    1
  67   68   71   73   80   83   94  106  109  111  117  118  120  123  128  163
   1    1    1    1    1    1    1    1    1    2    2    1    1    1    1    1
 183  189  194  243  258  271  281  289 1928 2121 4135 5261
   1    1    1    1    2    1    1    1    1    1    1    1
```

Sketch for Edge Pruning Algorithm

- Start with MST
- Remove the edge with the largest weight
  - If this creates a cluster and a runt, remove the runt from consideration
  - Otherwise, a cluster has broken into two clusters
- When to stop?
  - Ad Hoc
    - When there are at least k clusters
    - When a cluster is smaller than a certain size
  - When a cluster is split into two runts
  - Method based on histogram of cluster sizes

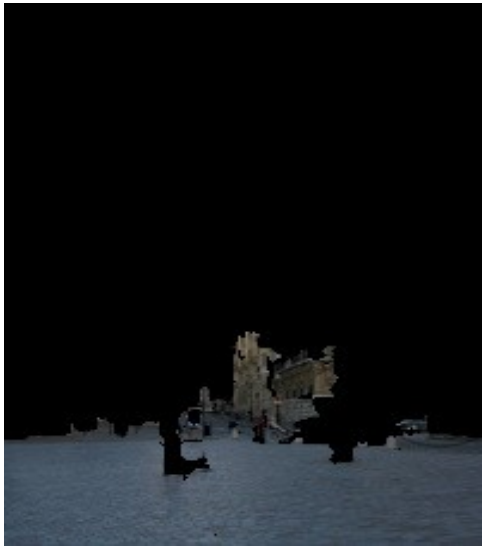
References

GONZALEZ and WOODS, Digital Image Processing, 1993, New York.

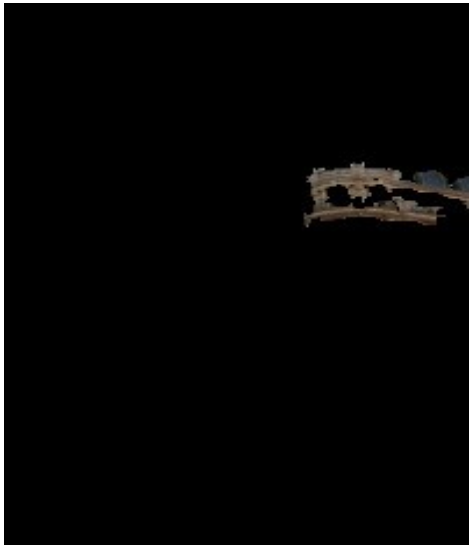
HARTIGAN, J.A., and MOHANTY, S. (1992), “The RUNT Test for Multi-modality”, Journal of Classification, 9, 63-70.

STUETZLE, W. “Estimating the cluster tree of a density by analyzing the minimal spanning tree of a sample.” Journal of Classification, Vol. 20, No. 5, 2003, pp. 25-47.

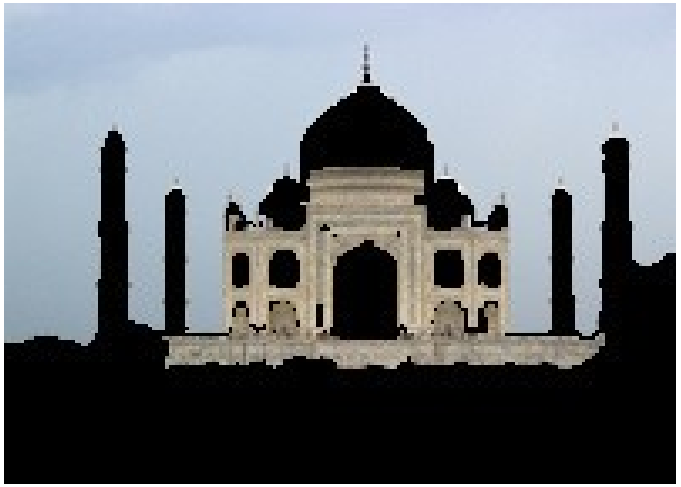
Pruning the Largest 5% of Edges

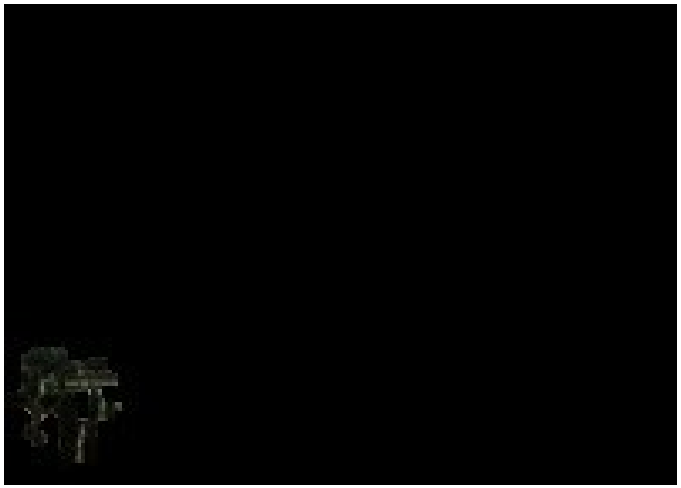


Pruning the Largest 5% of Edges



Pruning the Largest 5% of Edges





Pruning the Largest 5% of Edges

Pruning the Largest 5% of Edges

Pruning the Largest 5% of Edges