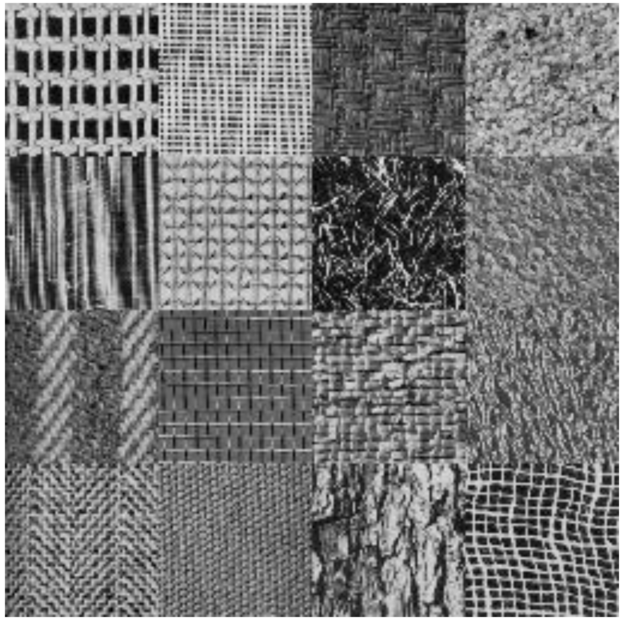
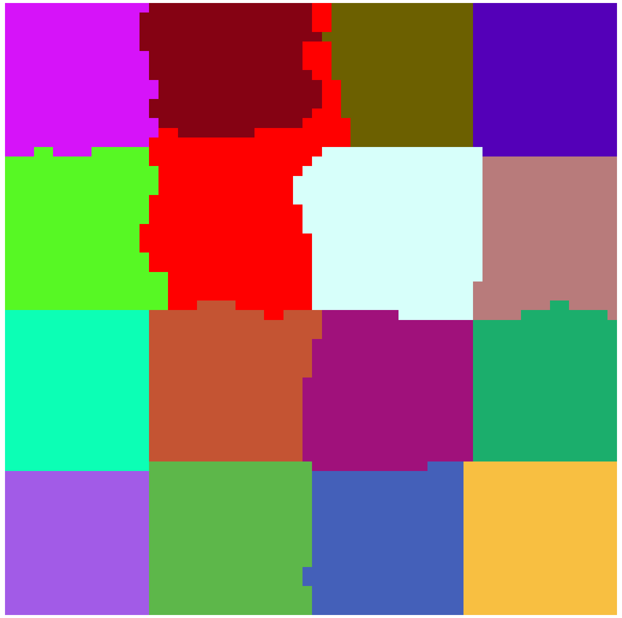
Efficient Graph based Image Segmentation

by P. Felzenszwalb & D. Huttenlocher- *In IJCV 2004*





Courtesy Uni Bonn

-Sahil Narang & Kishore Rathinavel

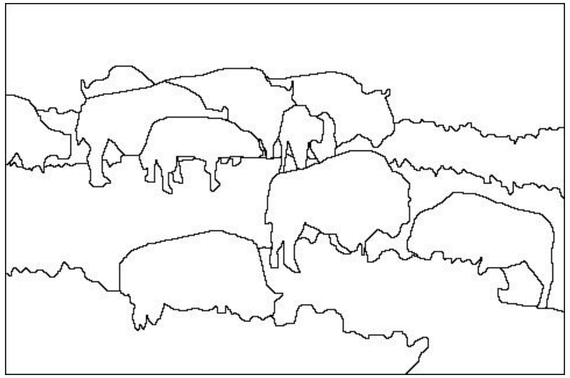
UNC Chapel Hill

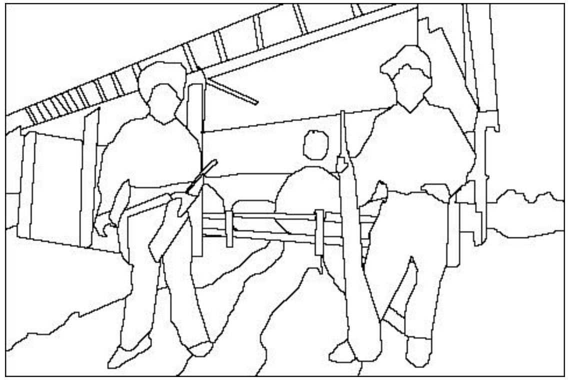
Goal

- Separate images into “coherent” objects.

image

human segmentation



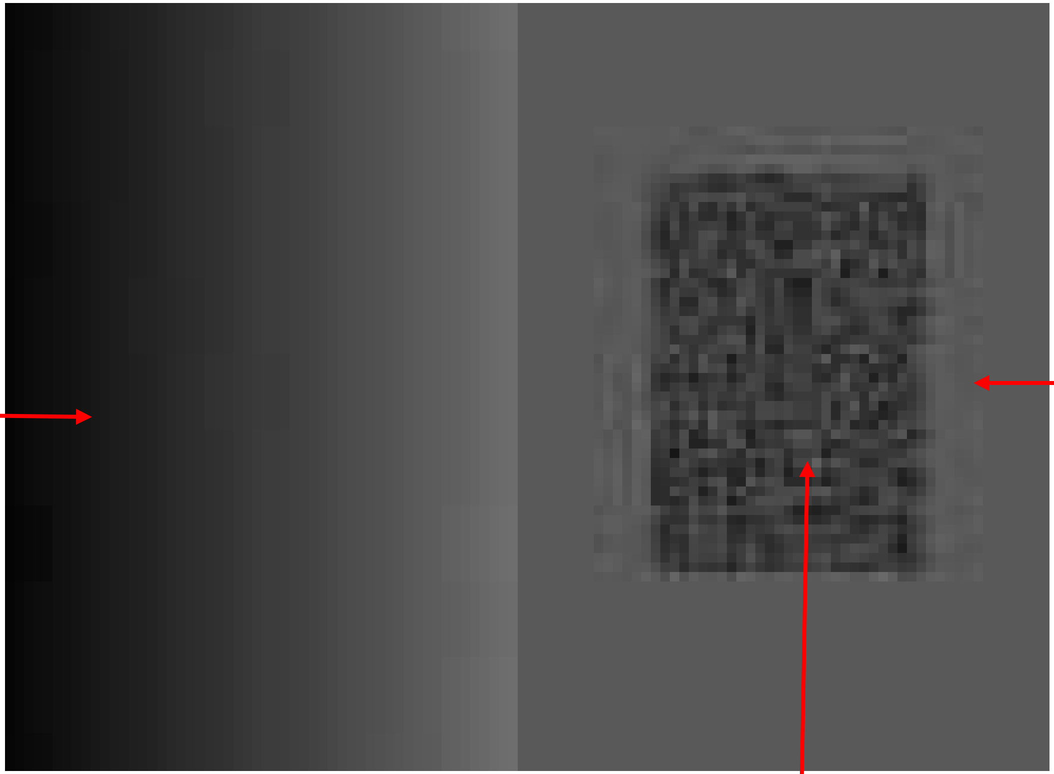


Berkeley Segmentation Database

Goal

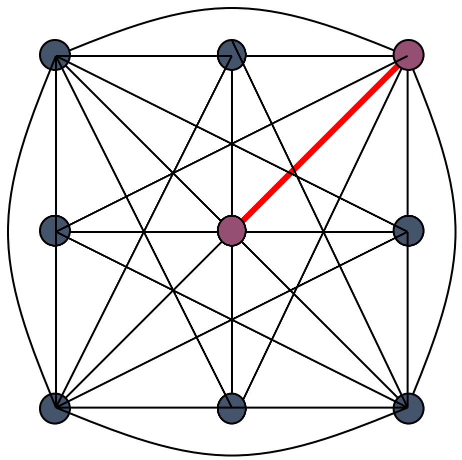
- Separate image into coherent “objects”
  - Top-down or bottom-up process?
  - Supervised or unsupervised?
- Group together similar-looking pixels for efficiency of further processing
  - Related to image compression
  - Measure of success is often application-dependent

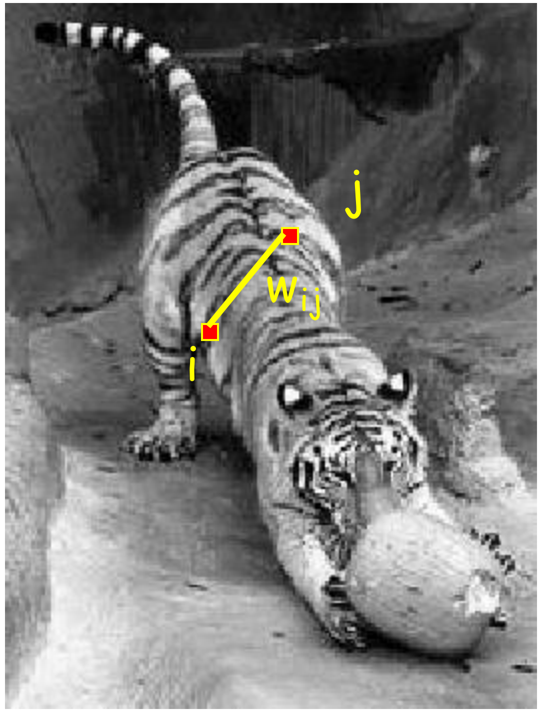
Algorithmic Requirements



1. Capture perceptually important groupings that reflect global aspects of the image
2. Be highly efficient, run time linear in the number of pixels

Images as graphs

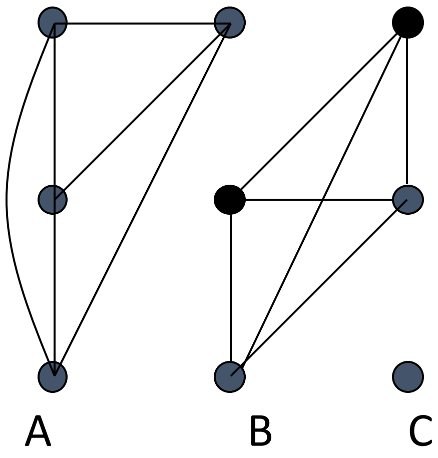




- Node for every pixel
- Edge between every pair of pixels (or every pair of "sufficiently close" pixels)
- Each edge is weighted by the *dis*similarity of the two nodes

Courtesy: Derek Hoeim

Segmentation via graph partitioning



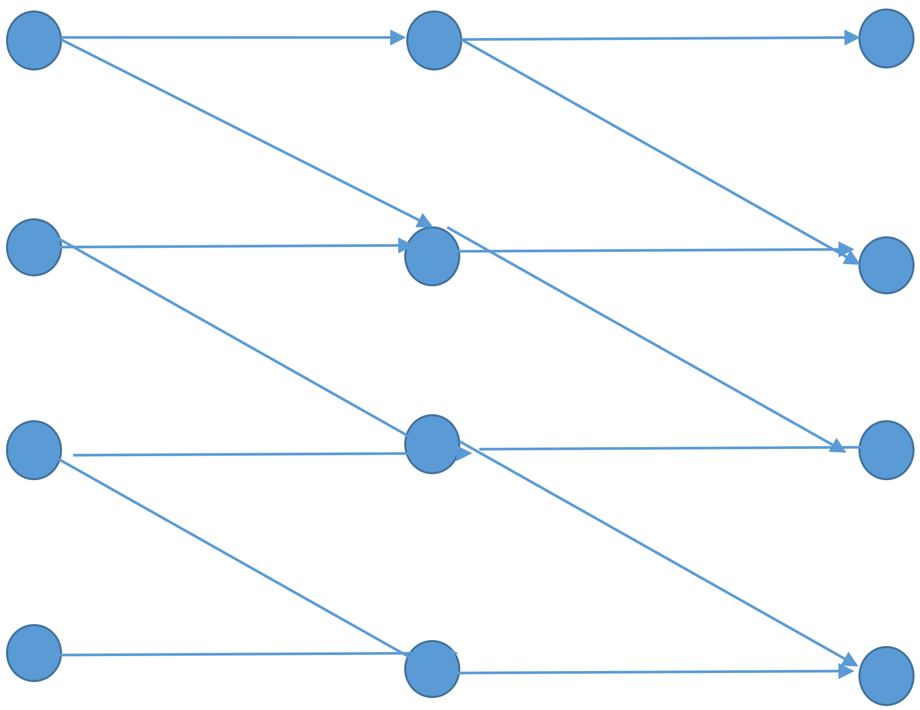
- Break Graph into Segments
  - Delete links that cross between segments
  - Easiest to break links that high weights
    - similar pixels should be in the same segments
    - dissimilar pixels should be in different segments

Courtesy: Derek Hoeim

Pairwise Region Comparison Predicate

Key Idea:

There exists a boundary between C1 and C2 iff the inter component differences is larger than the intra-component differences



Pairwise Region Comparison Predicate

Int(G)=6

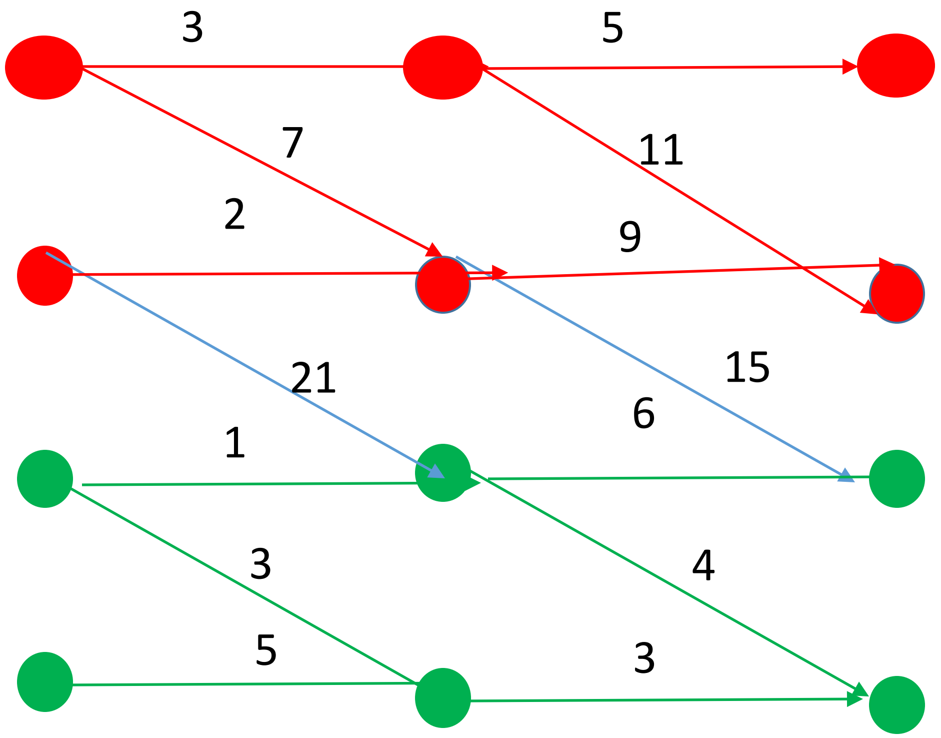
Int(R)=9

Mint=min(Int(G) + τ(G), Int(R) + τ(R))

= min(6 + 60/6, 9 +60/6)=16

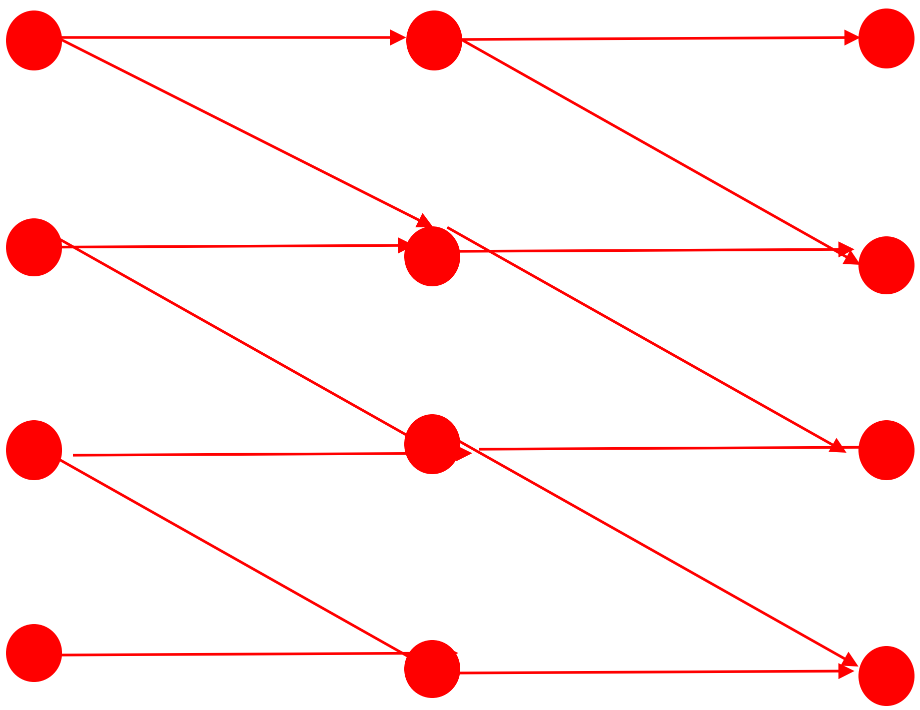
Diff(G,R)= min(21,15)=15

where τ(G) = k/|G|



Pairwise Region Comparison Predicate

- Diff(G,R) > Mint(G,R) ? False



Segmentation Algorithm

Input: $G = (V;E)$, with n vertices and m edges

Output: Segmentation of V into components $S = (C_1,\ldots,C_r)$

Algorithm

1. Sort E into $\pi = (o_1,\ldots.,o_m)$, by non-decreasing edge weight
2. Start with a segmentation S^0, where each vertex v_i is in its own component.
3. Repeat for $q = 1,\ldots.,m$.
   - Construct S^q from S^{q-1} as follows:
   - Let v_i and v_j denote the vertices connected by the q-th edge in the ordering, i.e., $o_q = (v_i; v_j)$.
   - If v_i and v_j are in disjoint components of S^{q-1} and $w(o_q)$ is small compared to the internal difference of both those components, then merge the two components otherwise do nothing.
4. Return $S=S^m$

Implementation

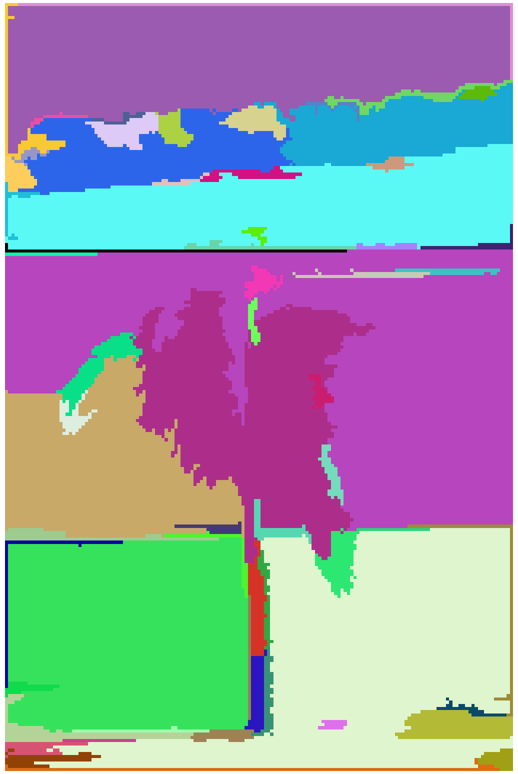
- Disjoint Set forests with union by rank and path compression
- Run Time
  - Sorting edges: O(mlogm)
  - Steps 2-4: O(mα(m)) where α is the inverse Ackerman's function
  - At most 3 disjoint set ops per edge
- (Not) Implemented
  - Channel based segmentation for color images
  - Nearest Neighbor graphs

Parameters

- σ
  - Gaussian Smoothing: Preprocessing to reduce noise
  - Can cause “bleeding” – the algorithm has difficulty separating background from the object if the boundaries are too smooth
  - Set to 0.8
- k
  - Sets scale of observation
  - Set to 300
- minSize
  - Post processing step to merge small components
  - Set to 20

Smoothing Effect

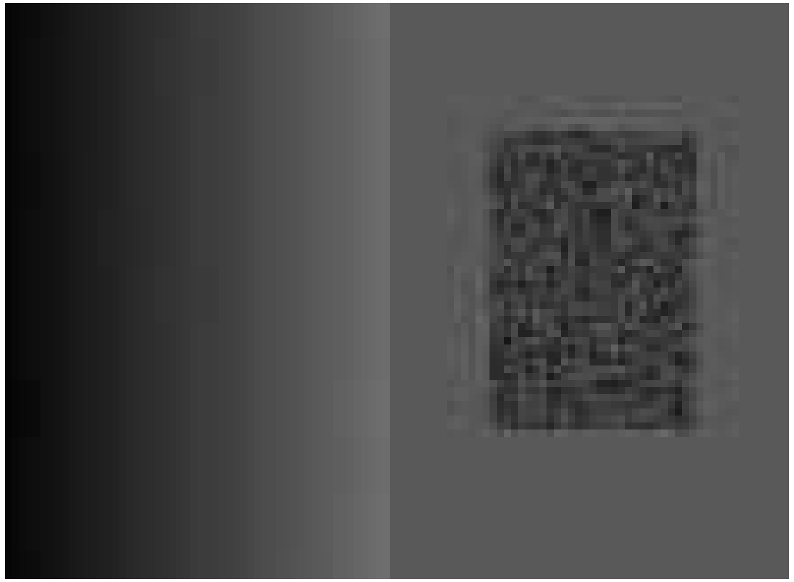
Original Image



σ=0.5

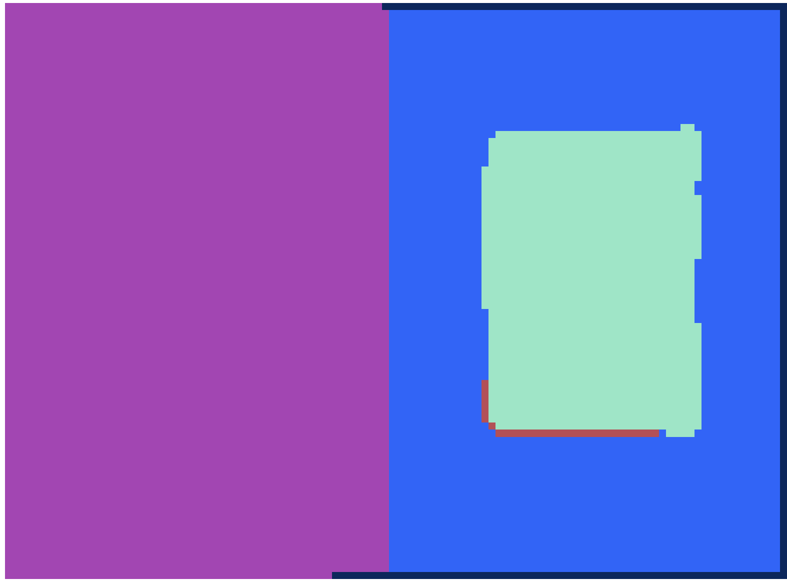
σ=1.5

Results



Original Image

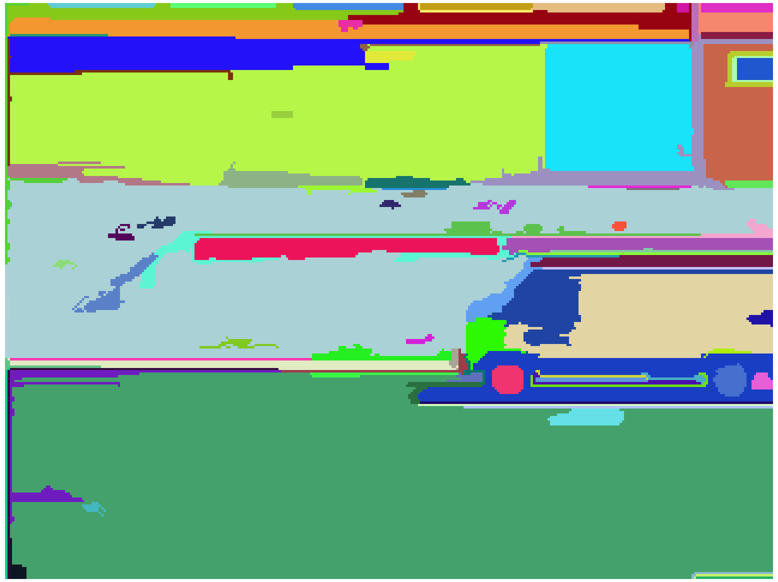
Author's implementation



Our implementation

Results

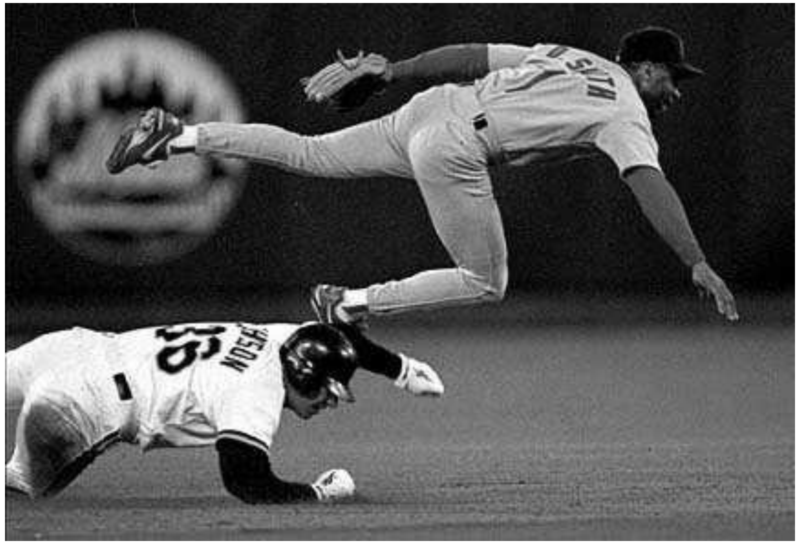
Original Image



Author’s implementation

Our implementation

Results



Original Image

Author’s implementation

Our implementation

Results

Original Image

Author’s implementation

Our implementation

Results

S.No	Scene	Dimensions	No. of Pixels	Time (sec)
1	Base case	81 x 110	8910	3.064557
2	mypeppers	192 x 125	24000	12.861267
3	Beach	240 , 159	38160	27.739771
4	Indoor	240 x 320	76800	125.466307
5	Street	240 x 320	76800	128.762103
6	Baseball	294 , 432	127008	481.096896

Conclusion

- Segmentation algorithm makes simple greedy decisions, yet obeys global properties
- Efficient: O(nlogn)
- Limited by use of minimum edge wt as evidence of boundary
- This assumption helps avoid making it NP-Hard
- Parameter dependent

Questions??

Supplemental Slides

Results

Original Image

Author's implementation

Our implementation

Pairwise Region Comparison Predicate

- Key Idea:
  - There exists a boundary between C1 and C2 iff the inter component differences is larger than the intra-component differences
- Internal Difference of a Component

$$Int(C) = \max_{e \in MST(C,E)} w(e)$$

- Difference between Components

$$Dif(C_1, C_2) = \min_{v_i \in C_1, v_j \in C_2, (v_i, v_j) \in E} w((v_i, v_j))$$

- Pairwise Comparison Predicate

$$D(C_1, C_2) = \begin{cases} \text{true} & \text{if } Dif(C_1, C_2) > MInt(C_1, C_2) \\ \text{false} & \text{otherwise} \end{cases}$$

Threshold parameter

- Minimum Internal Difference of two Components

$$MInt(C_1, C_2) = \min(Int(C_1) + \tau(C_1),\ Int(C_2) + \tau(C_2)).$$

  - Threshold dictates the degree to which the inter component difference must be greater that the intra component difference
  - Threshold based on size

$$\tau(C) = k/|C|$$

- K sets the scale of observation
- Can be used for shape based segmentation