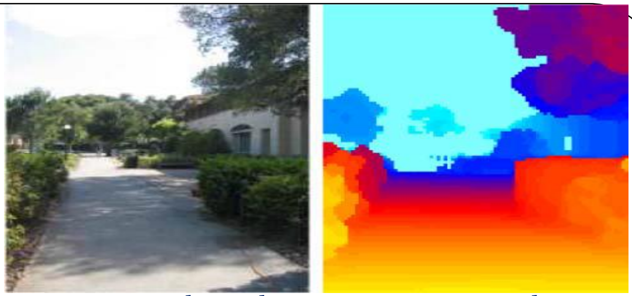
MRF and Stereo Image Segmentation (3D Reconstruction)

Bing Li

Introduction

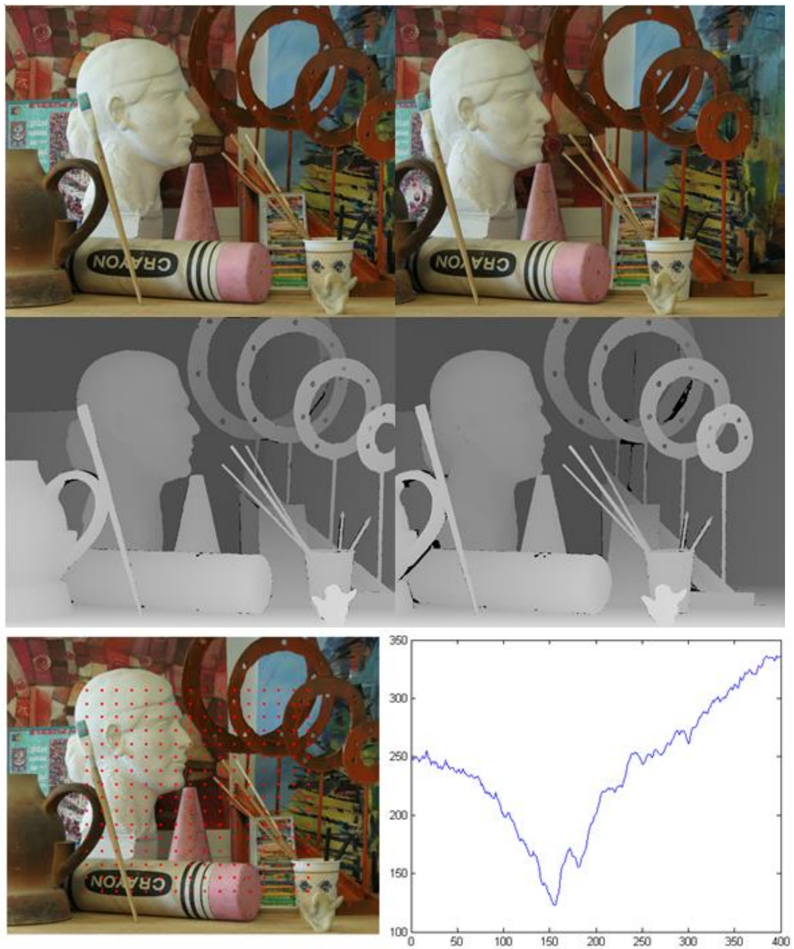


Ashutosh Saxena, Sung H. Chung and Andrew Y. Ng in 2007

- **Stereo Image Segmentation vs. 3D reconstruction**
  Learning depths helps in recognizing objects;
  Object coherent in space and color; Smoothness
- **Stereo Method vs. Single Image Method**
  Stereo cues: perspective relation, occlusion,
  Monocular cues: color, texture, focus, prior
- **Energy Functional over MRF**
  Optimization is NP-hard, needs Approximation
  State-of-the-arts techniques: Graph-cuts, Belief-propagation

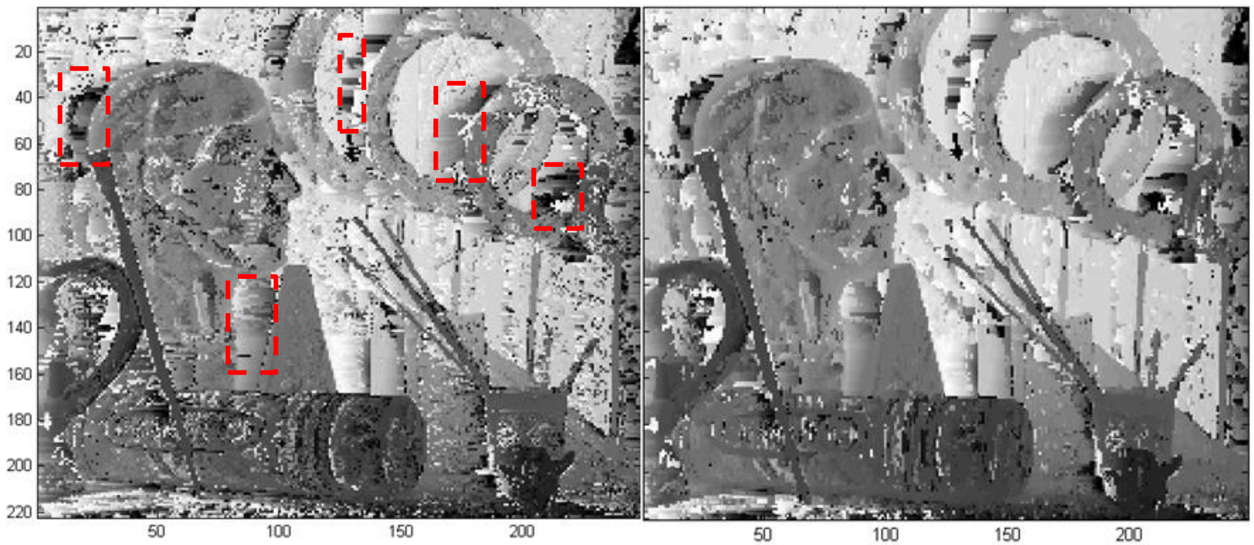
- **What I will do (Stereo method):**
  1. **Naive color matching without smoothness term**
  2. **Scanline using Dynamic Programming**
  3. **Dynamic Programming on a simple-tree structure** (Michael Bleyer and Margrit Gelautz)
  4. **Discuss the choice of Color Distance**
  5. **Occlusion Term in the Energy Functional**

Data Preprocessing



- Middlebury Stereo Datasets
  - “Art”
- Merge into Larger Pixels
  - 5 × 5 square
- Alternatively use super pixels to lower graph complexity
- Rough Alignment using a sampling array, minimize the total color distance vs. global shift/rotation
- Distance Definition:
  scalar, vector,
  sampling-insensitive

Stereo Matching without smoothness regularization



$$\sum match(p_i, q_{i+di})$$

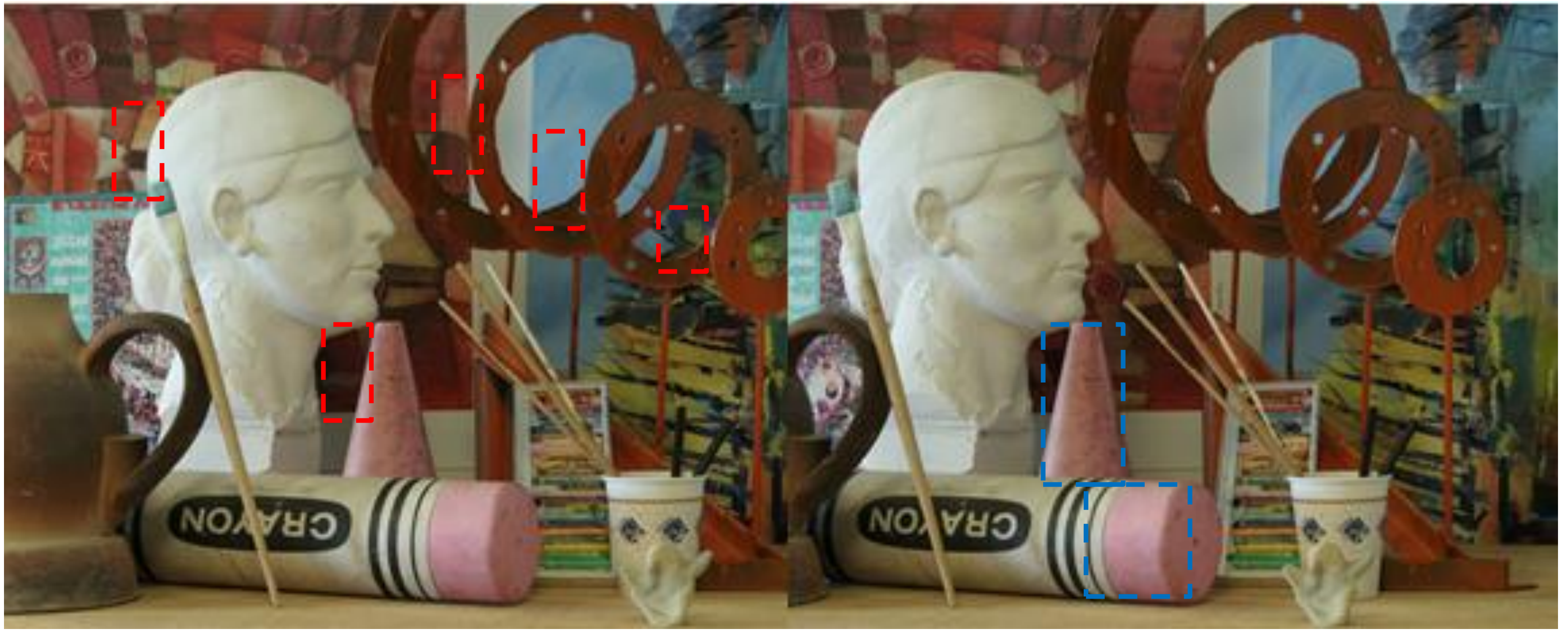
The left panel: search for the optimal match for each pixel (in the neighborhood)

The right panel: search for the optimal match for each 3-pixel vector

Drawback :

1. **noisy, didn't punish non-smoothness (consider a large region with the same color)**
2. **bad performance wherever occlusion happens, i.e. the pixel doesn't appear in pic2**
3. **texture is important, "the dirtier, the better"**

Stereo Matching without smoothness regularization

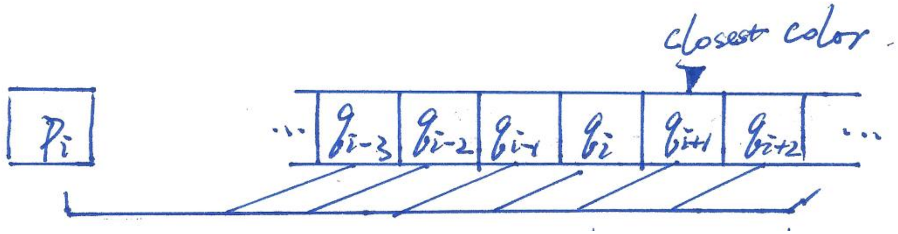


The left panel: search for the optimal match for each pixel (in the neighborhood)
The right panel: search for the optimal match for each 3-pixel vector

Drawback :

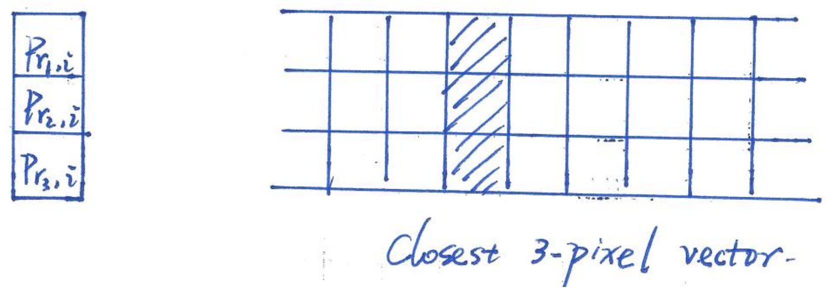
1. **noisy, didn't punish non-smoothness (consider a large region with the same color)**
2. **bad performance wherever occlusion happens, i.e. the pixel doesn't appear in pic2**
3. **texture is important, "the dirtier, the better"**

Color Distance

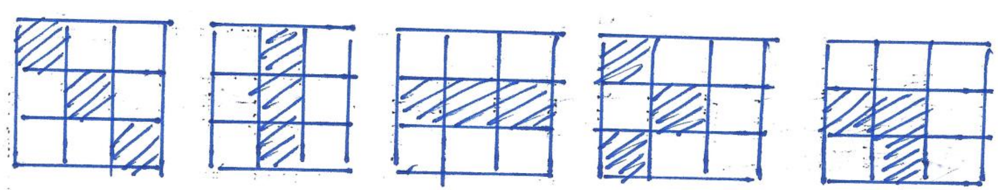


Compare with all qj's within some distance to qi

Closest color → optimal disparity

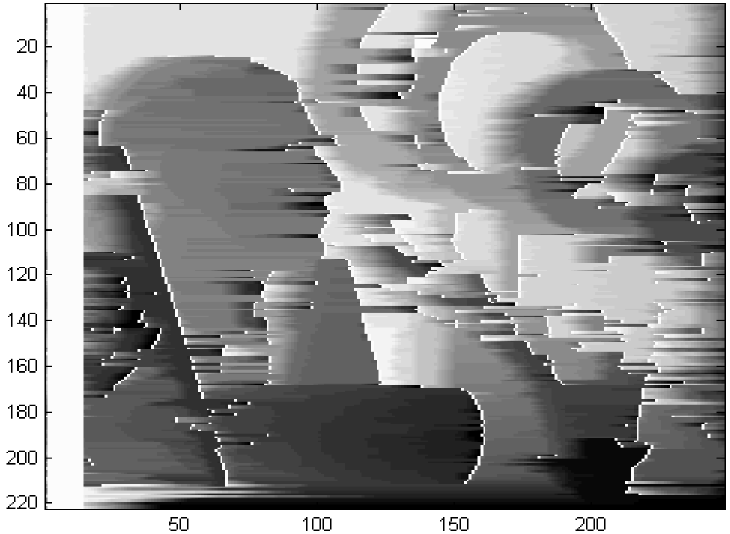


Closest 3-pixel vector → optimal disparity



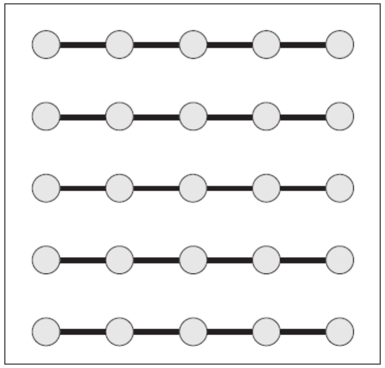
Comparison of miniature structure with other shapes

Scanline Using Dynamic Programming

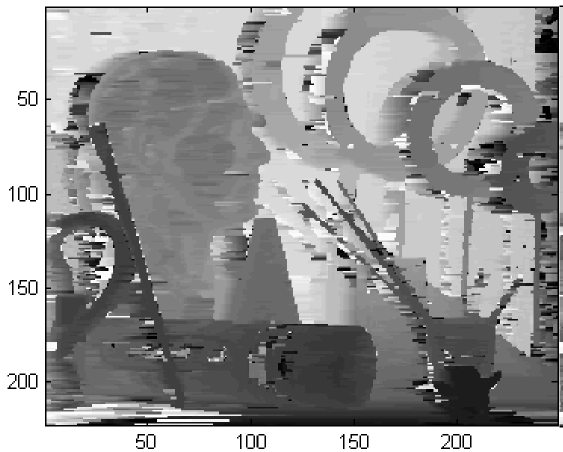


$$\sum match(p_i, q_{i+di}) + \sum discontinuity(d_i, d_{i+1})$$

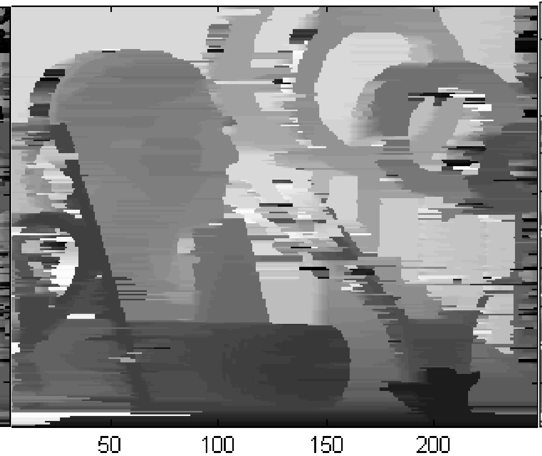
- Algorithm:
  - For each line:
  - … scan from left to right, pixel i = 1 to n
  - … define l(pi, dpi) as the minimum 'loss' up to pixel i if the disparity of pi is dpi (the possible values of dpi $\in [-m, m]$
  - … trace back i from n to 1 find the configuration which gives the minimal 'loss'(energy)
- Drawbacks:
  - Simplified too much, different lines don't talk to each other
  - Streaking issue (apparent in the image).



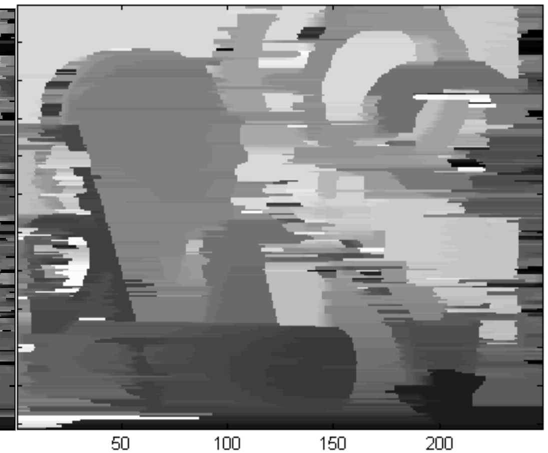
Vary the ratio between smoothing and matching term



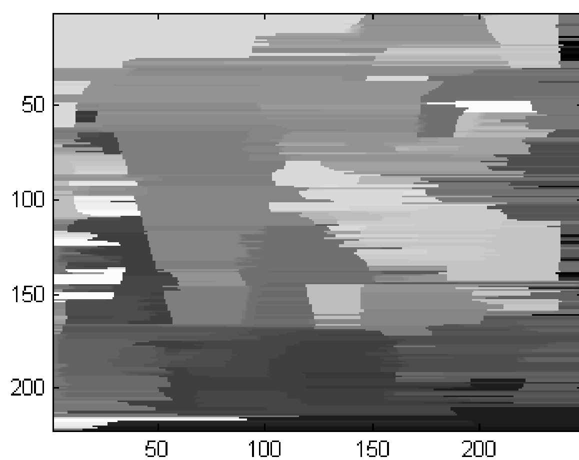
0.03



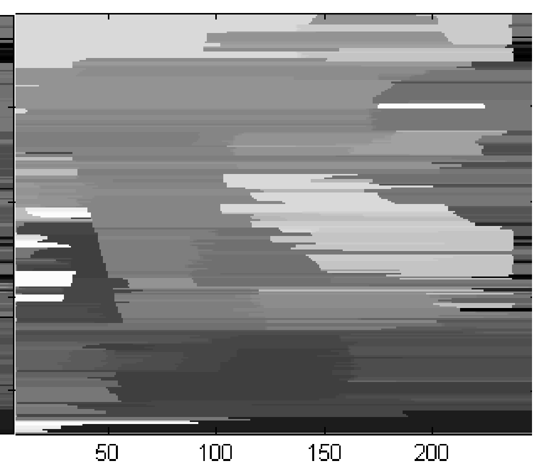
0.15



0.3

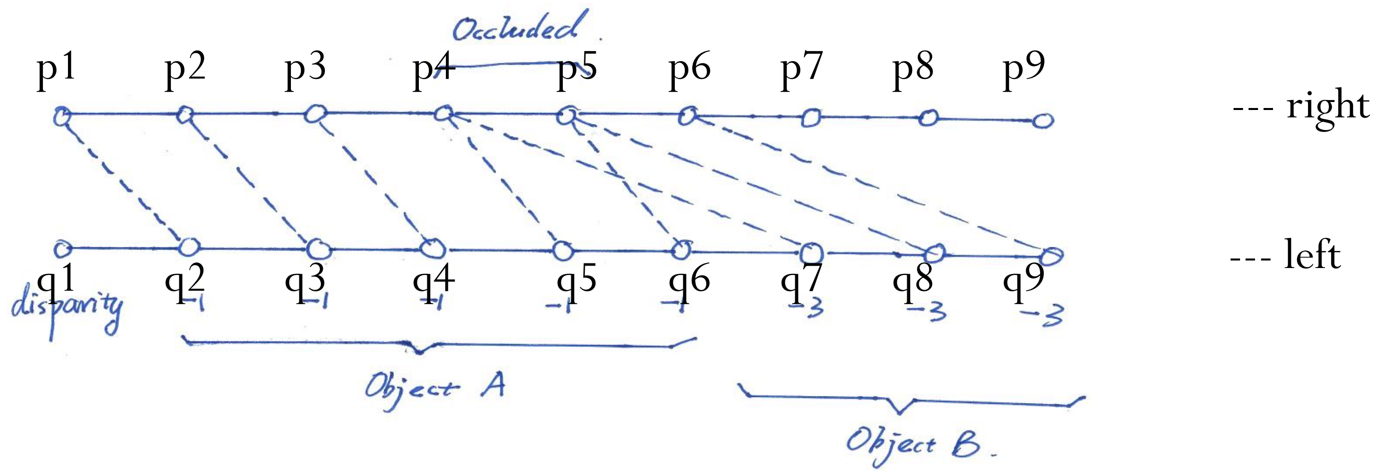


0.9



1.5

Occlusion term in the Energy func



The naïve Energy writes as :

$$\color{red}{m(q5, p4) + m(q6, p5)} + m(q7, p4) + m(q8, p5) + \cdots$$

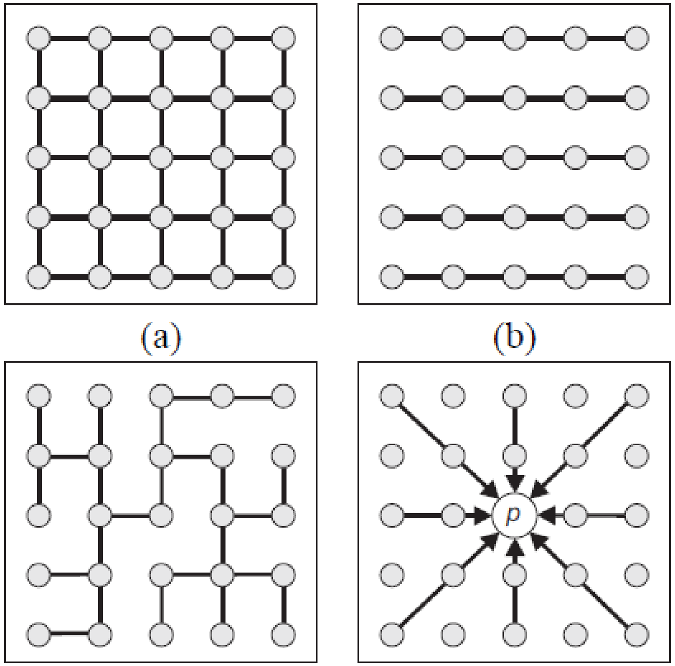
A better Energy function writes as :

$$m(q7, p4) + m(q8, p5) + penalization + \cdots$$

We don't know the correct occlusion relation, try using energy function :

$$\sum_{no\ overlap} match(q_i, p_{i+di}) + \sum discontinuity(d_i, d_{i+1}) + \sum_{q_i\ blocked} penalization\ on\ q_i$$

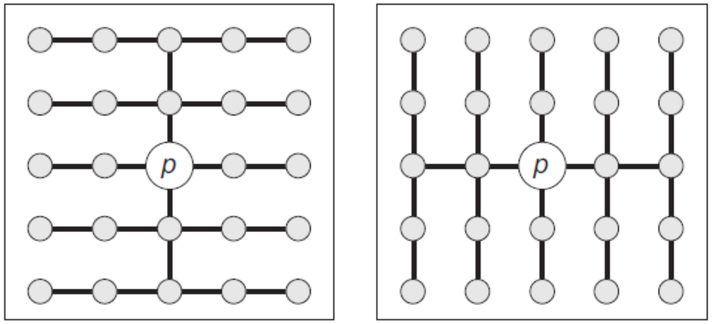
Line to Tree



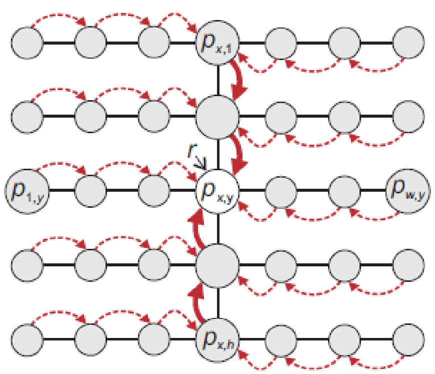
(a) (b) (c) (d)

- efficient DP-based optimization also works on tree structures and now we have both horizontal and vertical connections
- A possible way is to start with (a), the tree is constructed by discarding edges that show a high gradient in the intensity image, end up with graph (c).
- Scanline horizontally, then use results as bias, scanline vertically
- Disadvantage: still discard too many edges.

Simple-Tree Method



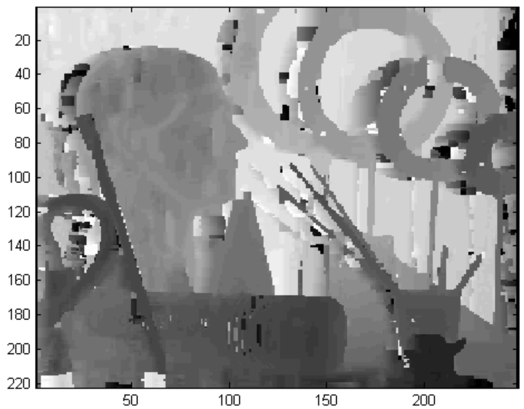
(a) (b)



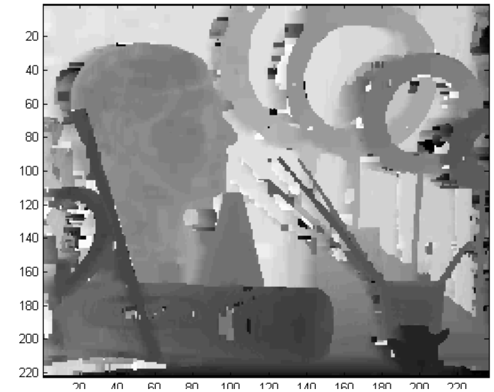
- For each pixel construct an individual tree, with that pixel as the root. Find disparity for p that minimizes the energy on this simple tree and assign it to this root pixel. Repeat for all pixels.
- Algorithm(for Horizontal tree)
  - Run DP scanline for each horizontal line, both forwards(F(p,d)) and backwards(B(p,d))
  - Calculate intermediate data structure C, V, H
  - C(p,d) is the cost of a whole line with optimal disparity solution conditioned that p is assigned d.
  - V(p,d) is the minimal cost of the whole tree, calculated by DP on C
- Calculate V on vertical tree, then use V as bias and calculate H on horizontal tree

$$m'(p,d) = m(p,d) + \lambda \cdot (V(p,d) - \min_{i \in \mathcal{D}} V(p,i))$$

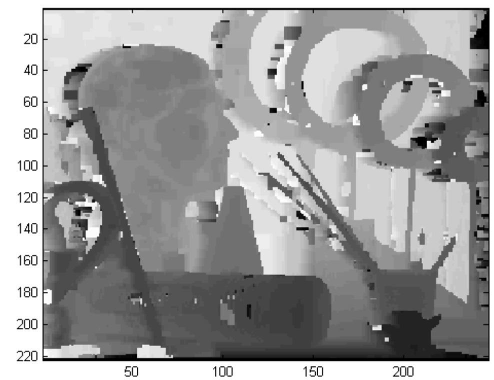
Varying Lambda



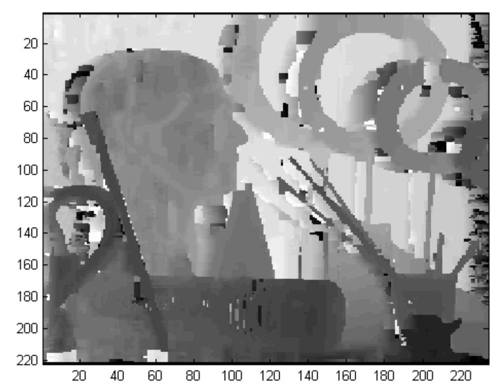
Vertical tree result:
using only V



Lambda = 0
(H limit)



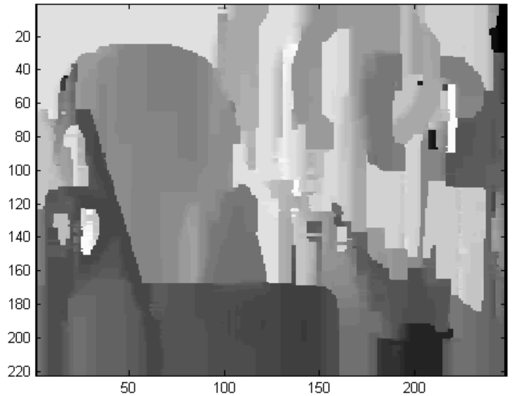
Lambda = 1



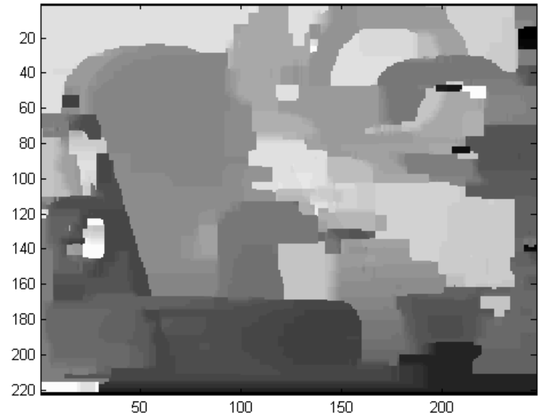
Lambda = inf
(V limit)

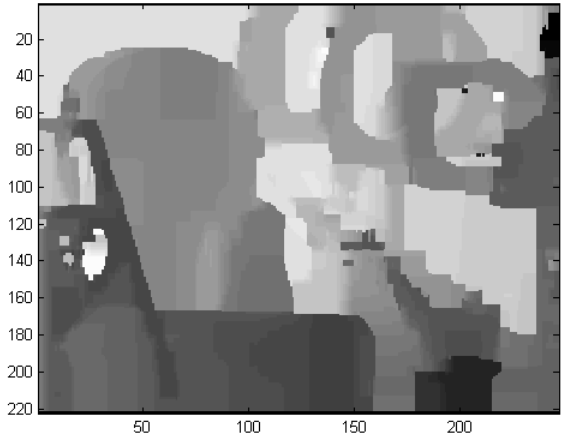
More examples:

Vertical



Horizontal





Combine

Other Successful Methods:

- Object Stereo - Joint Stereo Matching and Object Segmentation

  Michael Bleyer, Carsten Rother, Pushmeet Kohli, Daniel Scharstein, Sudipta Sinha

$$E(F,O) = E_{pc}(F,O) + E_{oc}(O) + E_{dc}(F,O) + E_{col}(O) + E_{par}(F,O) + E_{mdl}(O) + E_{con}(F,O)$$

- Before doing stereo matching, do image segmentation on single picture first, like superpixels.