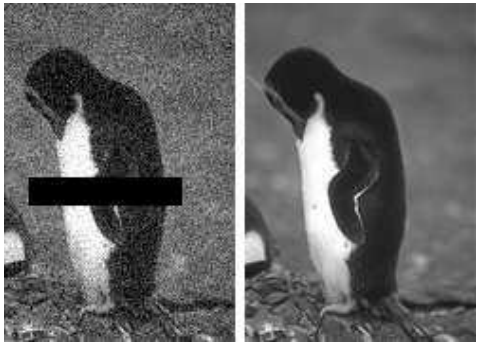
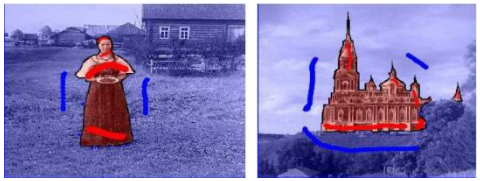
Markov Random Fields and its Applications

Huiwen Chang

Introduction

- Markov Random Fields(MRF)
  - A kind of undirected graphical model
- To model vision problems:
  - Low level: image restoration, segmentation, texture analysis…
  - High level: object recognition and matching(structure from motion, stereo matching)…

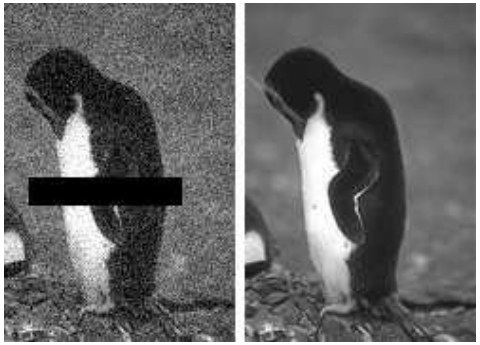


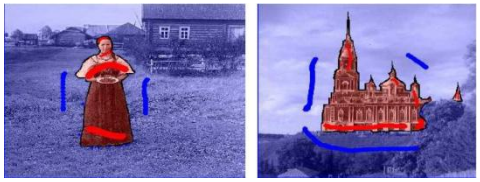


Introduction

- Markov Random Fields(MRF)
  - A kind of undirected graphical model
- To model vision problems:
  - Low level: image restoration, segmentation, texture analysis…
  - High level: object recognition and matching(structure from motion, stereo matching)…

Labeling Problem





Graphical Models

- Probabilistic graphical models

  Nodes: random variables

  Edges: statistical dependencies among random variables

  – Advantage

    Compact and efficient way to visualize conditional independence assumptions to represent probability distribution

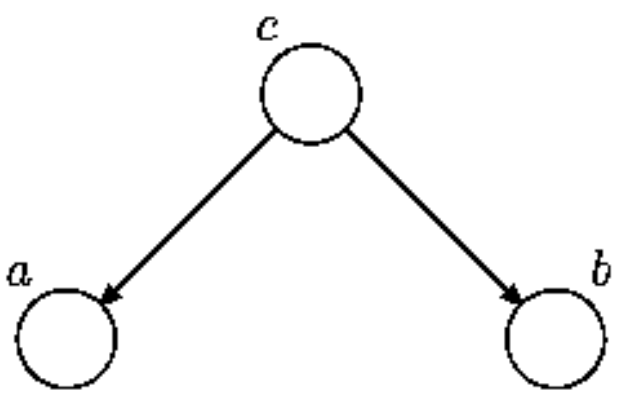
Conditional Independence

Conditional Independence

Random variables $\mathbf{y}$ and $\mathbf{x}$ are *conditionally independent* given $\mathbf{z}$ if $\mathrm{P}(\mathbf{y}, \mathbf{x} \mid \mathbf{z}) = \mathrm{P}(\mathbf{y} \mid \mathbf{z})\,\mathrm{P}(\mathbf{x} \mid \mathbf{z})$.

Graphical Model

- Bayesian Network
  - Directed acyclic graph:

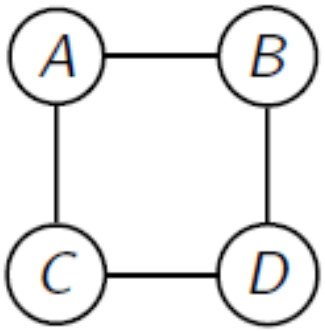


- Factorization: $p(a, b, c) = p(a|c)p(b|c)p(c)$

- Conditional Independence: $a \perp\!\!\!\perp b \mid c$

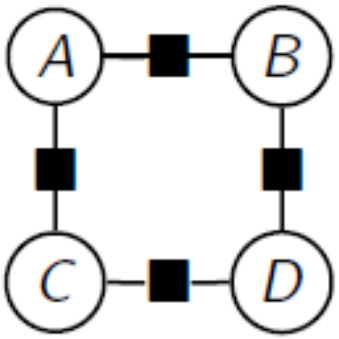
Graphical Model

- Markov Network (MRF)



$a \perp\!\!\!\perp d \mid b, c$

$b \perp\!\!\!\perp c \mid a, d$



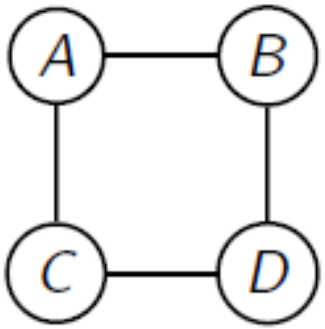
$$P(a,b,c,d) = \frac{1}{Z}\Psi(a,b)\Psi(b,d)\Psi(d,c)\Psi(c,a)$$

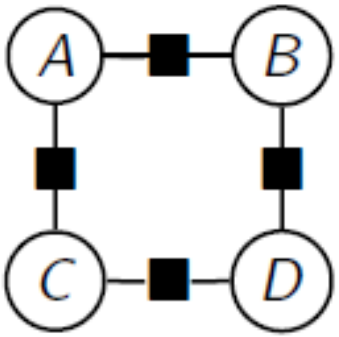
Potential functions over the **maximal clique** of the graph

Potential function $\psi(.) > 0$

Graphical Model

- Markov Network (MRF)





$a \perp\!\!\!\perp d \mid b, c$

$b \perp\!\!\!\perp c \mid a, d$

$$P(a,b,c,d)=\frac{1}{Z}\Psi(a,b)\Psi(b,d)\Psi(d,c)\Psi(c,a)$$

Hammersley-Clifford theorem says
such distributions that are consistent with the set of conditional independence statements & the set of such distributions that can be expressed as a factorization by the maximal cliques of the graph are **identical**

MAP inference

- Posterior probability of the labelling y given observation x is :

Bayes Rule

$$\underbrace{\mathrm{P}(\mathbf{y} \mid \mathbf{x})}_{\text{posterior}} = \frac{\overbrace{\mathrm{P}(\mathbf{x} \mid \mathbf{y})}^{\text{likelihood}} \cdot \overbrace{\mathrm{P}(\mathbf{y})}^{\text{prior}}}{\mathrm{P}(\mathbf{x})}$$

Maximum a Posteriori (MAP) inference: $\mathbf{y}^\star = \operatorname{argmax}_{\mathbf{y}} \mathrm{P}(\mathbf{y} \mid \mathbf{x})$.

Markov Random Field

- Posterior probability of the labelling y given observation x is :

$$\mathrm{P}(y|x) = \frac{1}{Z(x)} \prod_{c} \varphi_c(\boldsymbol{y}_C; x)$$

cliques

Potential functions

where $Z(x) = \sum \prod_c \varphi_c(\boldsymbol{y}_C; x)$ is called the *partition function*.

- Since we define potential function is strictly positive, we can express them as exponentials:

$$\mathrm{P}(\mathbf{y} \mid \mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\{-E(\mathbf{y}; \mathbf{x})\}$$

The *energy function* $E(\mathbf{y}; \mathbf{x})$ usually has some structured form:

$$E(\mathbf{y}; \mathbf{x}) = \sum_{c} \psi_c(\mathbf{y}_c; \mathbf{x})$$

MAP inference

- Posterior probability of the labelling y given observation x is :

Bayes Rule

$$\underbrace{\mathrm{P}(\mathbf{y} \mid \mathbf{x})}_{\text{posterior}} = \frac{\overbrace{\mathrm{P}(\mathbf{x} \mid \mathbf{y})}^{\text{likelihood}} \cdot \overbrace{\mathrm{P}(\mathbf{y})}^{\text{prior}}}{\mathrm{P}(\mathbf{x})}$$

Maximum a Posteriori (MAP) inference: $\mathbf{y}^\star = \operatorname{argmax}_{\mathbf{y}} \mathrm{P}(\mathbf{y} \mid \mathbf{x})$.

- The most possible labeling is to minimize the Energy

$$\mathbf{y}^\star = \operatorname*{argmax}_{\mathbf{y}} \mathrm{P}(\mathbf{y} \mid \mathbf{x}) = \operatorname*{argmin}_{\mathbf{y}} E(\mathbf{y}; \mathbf{x}).$$

Pairwise MRF

- Most common energy function for image labeling

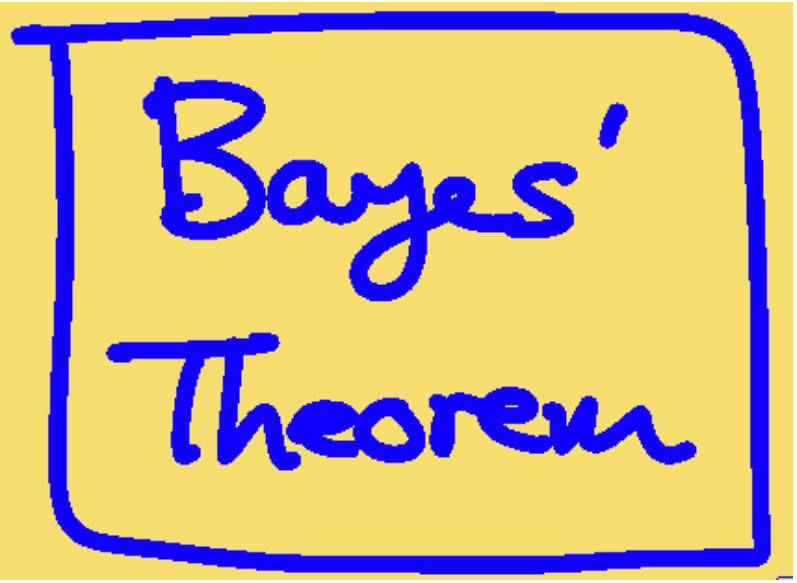
$$E(\mathbf{y};\mathbf{x}) = \sum_{c} \psi_c(\mathbf{y}_c;\mathbf{x})$$

$$= \sum_{i\in\mathcal{V}} \underbrace{\boxed{\psi_i^U(y_i;\mathbf{x})}}_{\text{Unary}} + \sum_{ij\in\mathcal{E}} \underbrace{\boxed{\psi_{ij}^P(y_i,y_j)}}_{\text{Pairwise}}$$

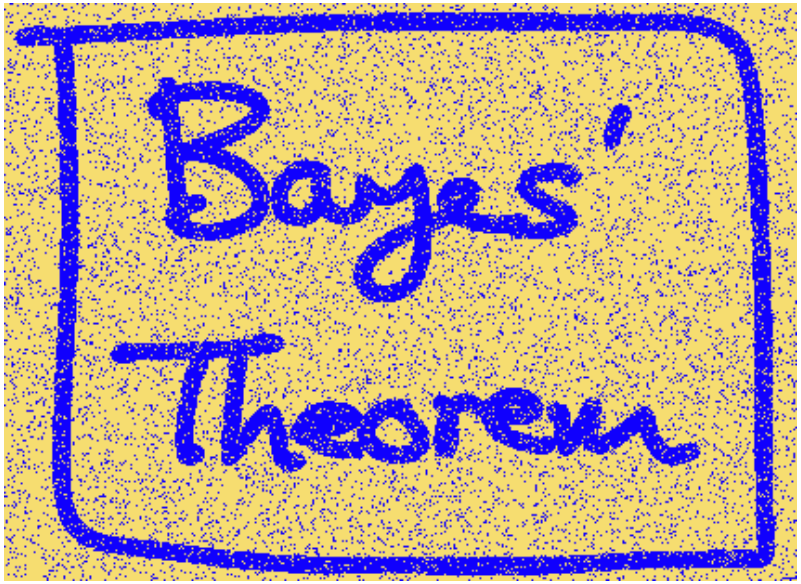
- Which of the energy acted as the prior?

Example MRF model: Image Denoising

- How can we retrieve the original image given the noisy one?



Original image Y

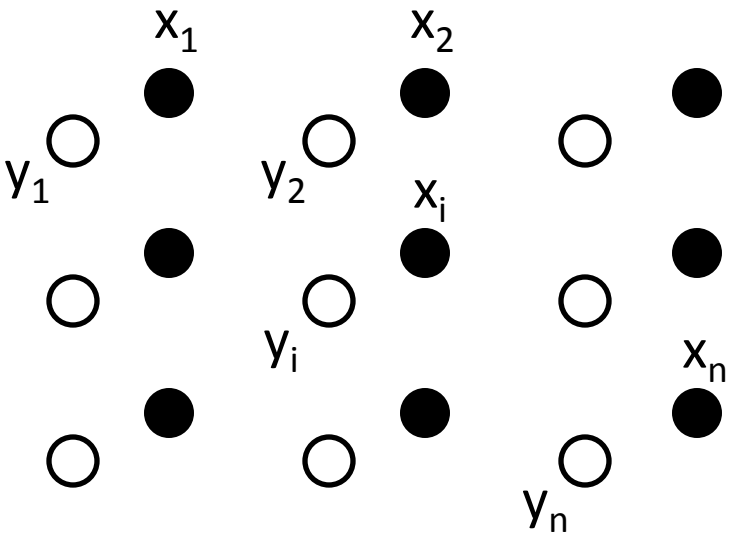


Noisy image X(**Input**)

MRF formulation

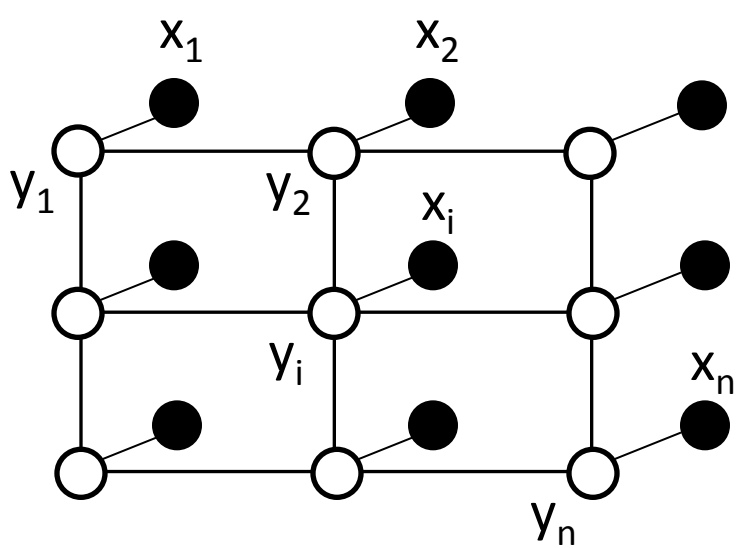
- <u>Nodes</u>
  - For each pixel i,
    - y_i : latent variable (value in original image)
    - x_i : observed variable (value in noisy image)

    Simple setting : $x_i, y_i \in \{-1,1\}$



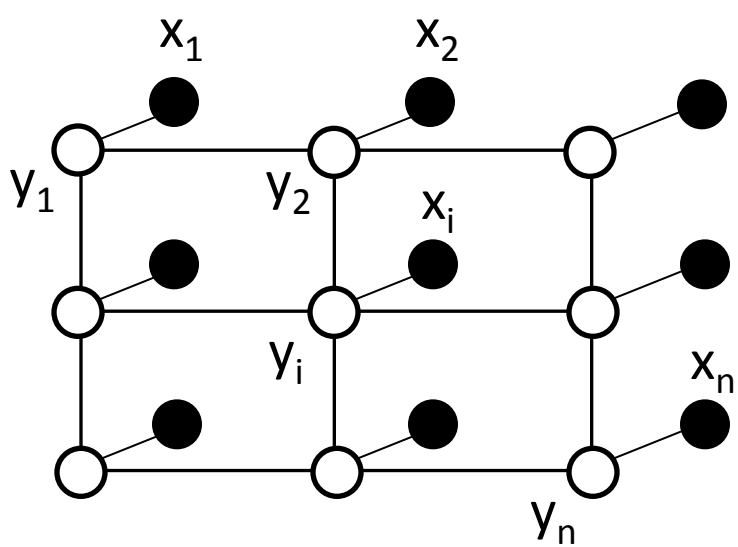
MRF formulation

- <u>Edges</u>
  - x_i,y_i of each pixel i correlated
  - neighboring pixels, similar value(smoothness)



MRF formulation

- <u>Edges</u>
  - x_i,y_i of each pixel i correlated $\phi(x_i, y_i) = -\beta x_i y_i$
  - neighboring pixels, similar value(smoothness) $\psi(y_i, y_j) = -\alpha y_i y_j$

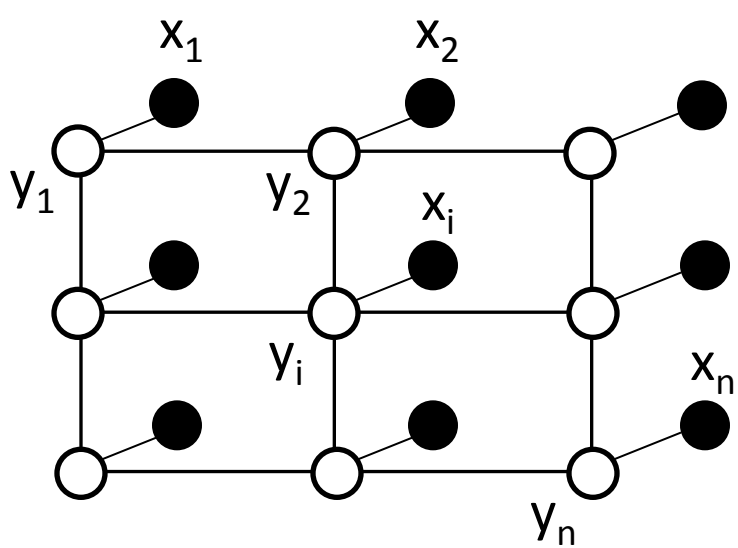


MRF formulation

Energy function

$$\mathrm{E}(\mathrm{y}; x) = \sum_{ij} \psi(y_i, y_j) + \sum_i \phi(x_i, y_i)$$

$$= -\alpha \sum_{ij} y_i y_j - \beta \sum_i x_i y_i$$



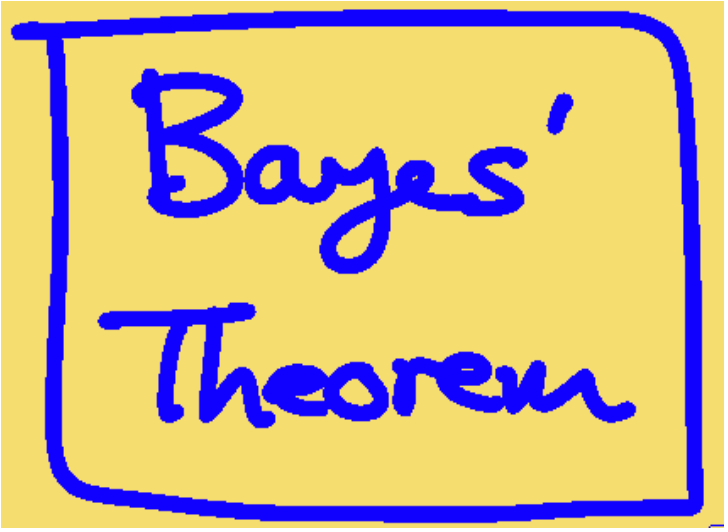
Optimization

Energy function

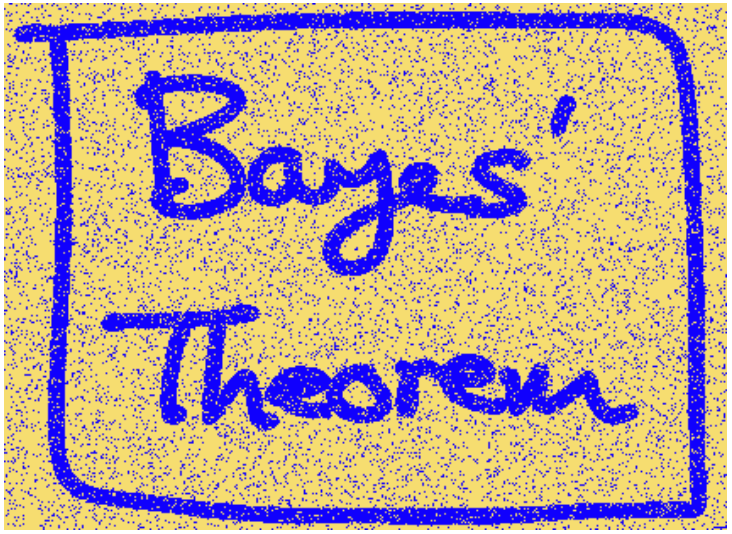
$$\mathrm{E}(\mathrm{y}; x) = \sum_{ij} \psi(y_i, y_j) + \sum_{i} \phi(x_i, y_i)$$
$$= -\alpha \sum_{ij} y_i y_j - \beta \sum_{i} x_i y_i$$

Iterated Conditional Modes (ICM)

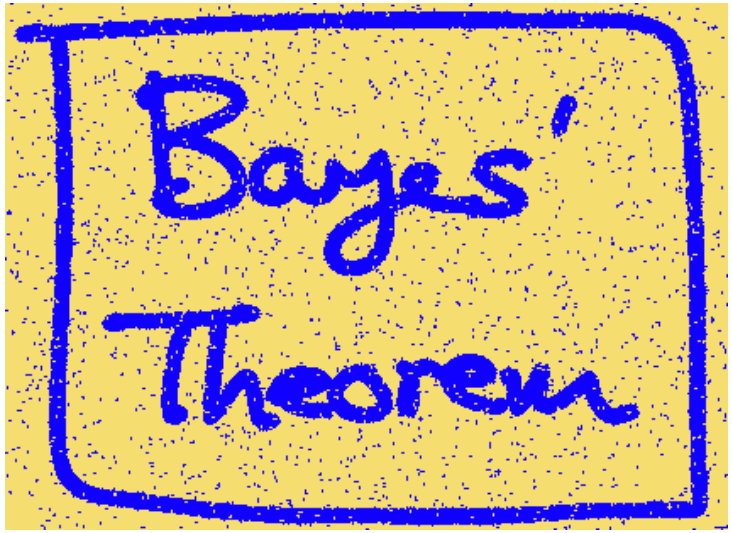
- Initialize $y_i = x_i$ for all i
- Take a y_i, fix others, flip y_i if $-y_i$ make energy lower
- Repeat until converge



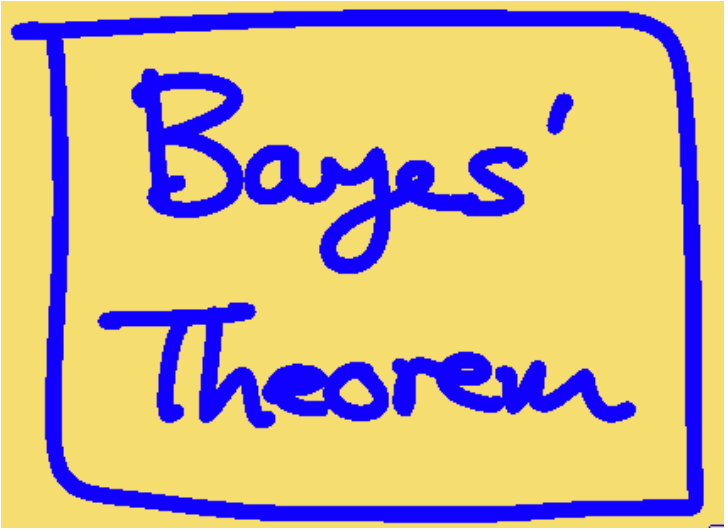
original image



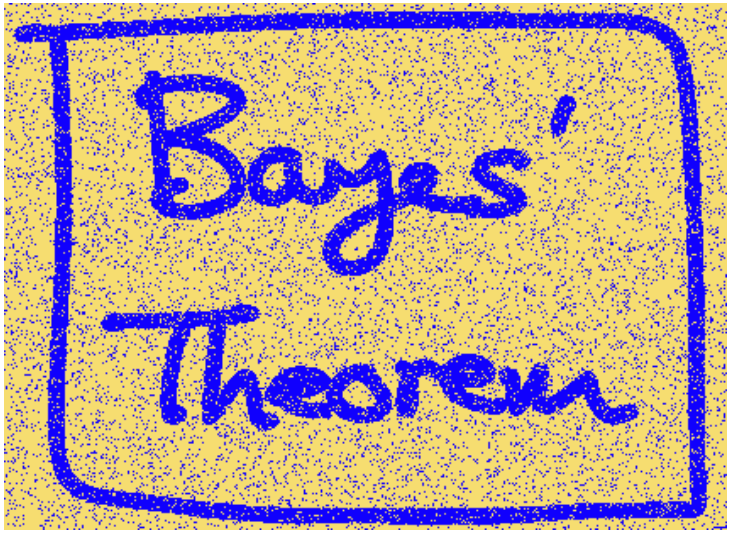
noise image(10%)



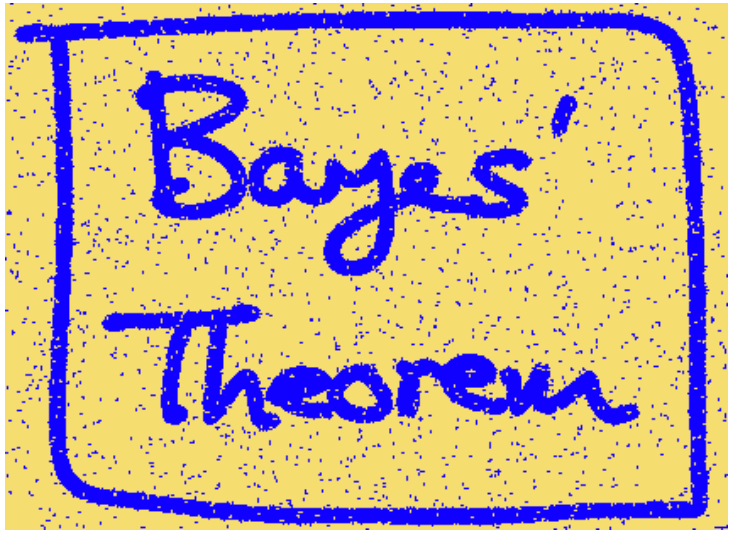
Restored by ICM(4%)



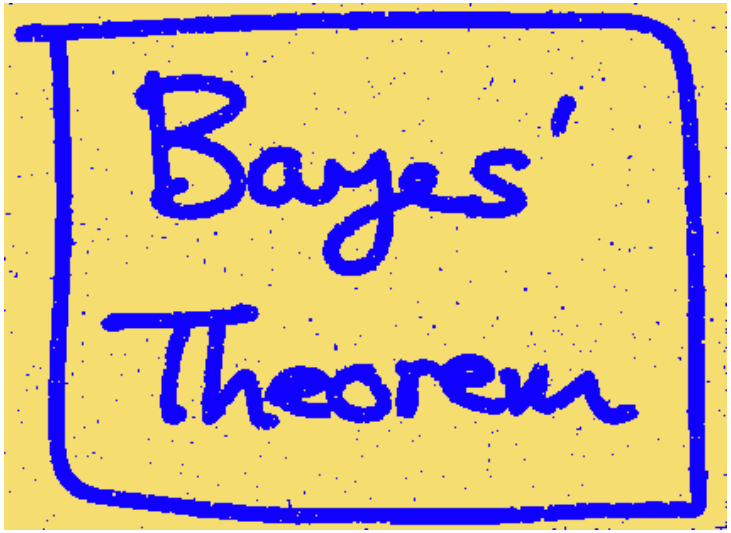
original image



noise image(10%)



Restored by ICM(4%)



Restored by Graph Cut(<1%)

Optimization

- Iterated Conditional Modes(ICM)
- Graph Cuts (GC)
- Message Passing
  - Belief Propagation (BP)
  - Tree Reweighted (not concluded here)
- LP-relaxation
  - Cutting-plane (not concluded here)

……

Graph Cuts

- To find labeling f that minimizes the energy

$$E(f) = \sum_{\{p,q\}\in\mathcal{N}} \boxed{V_{p,q}(f_p, f_q)} + \sum_{p\in\mathcal{P}} \boxed{D_p(f_p)},$$

$\mathrm{E}(f; X)$

neighbors

E_{smooth}

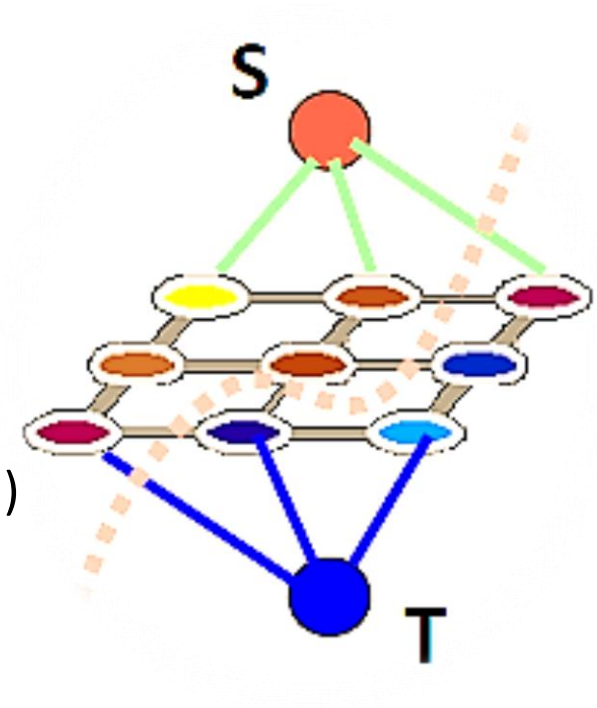
pixels

E_{data}

$D_p(f_p; X_p)$

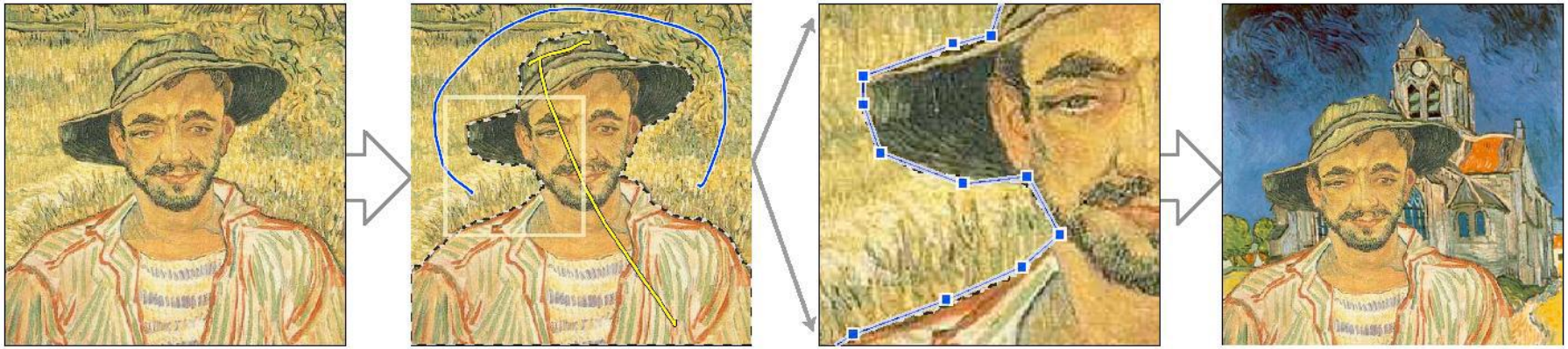
Graph Cuts

- For labeling problem
  - 2 Labels: Find global minimum
    - Max-flow/min-cut algorithm
    - Boykov and Kolmogrov 2001
      - the worst case complexity O(mn^2 |C|)
      - Fast in practice
  - For multi-labels: Computing the global minimum is NP-hard
    - Will discuss later



Two-Label Example: Lazy snapping

- Goal: Separate foreground from background



- 1st step: Use stroke to select
- 2nd step: Use polygon with vertices to refine

Model the problem

$$E(X) = \sum_{i \in \mathcal{V}} E_1(x_i) + \lambda \sum_{(i,j) \in \mathcal{E}} E_2(x_i, x_j)$$

$E_1(x_i)$: Likelihood Energy
$E_2(x_i, x_j)$: Prior Energy

X_i: Label of node i.
in {0: *background*, 1: *foreground*}

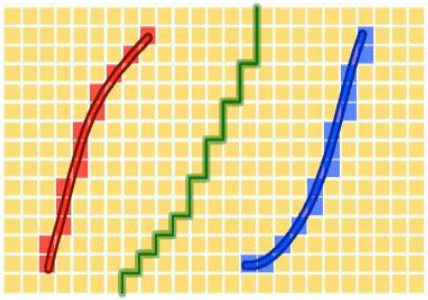
Likelihood Energy

- Data Term

$$E_1(x_i = 1) = 0 \qquad E_1(x_i = 0) = \infty \qquad \forall i \in \mathcal{F}$$

$$E_1(x_i = 1) = \infty \qquad E_1(x_i = 0) = 0 \qquad \forall i \in \mathcal{B}$$

$$E_1(x_i = 1) = \frac{d_i^{\mathcal{F}}}{d_i^{\mathcal{F}} + d_i^{\mathcal{B}}} \qquad E_1(x_i = 0) = \frac{d_i^{\mathcal{B}}}{d_i^{\mathcal{F}} + d_i^{\mathcal{B}}} \qquad \forall i \in \mathcal{U}$$



- Use color similarity with known pixel to give an energy for uncertain pixel

– d_i^F : Minimum distance to front color

– d_i^B : Minimum distance to background color

Prior Energy

- Smoothness Term

$$E_2(x_i, x_j) = |x_i - x_j| \cdot g(C_{ij})$$

$$C_{ij} = ||C(i) - C(j)||^2 \quad g(\varepsilon) = \frac{1}{\varepsilon + 1}$$

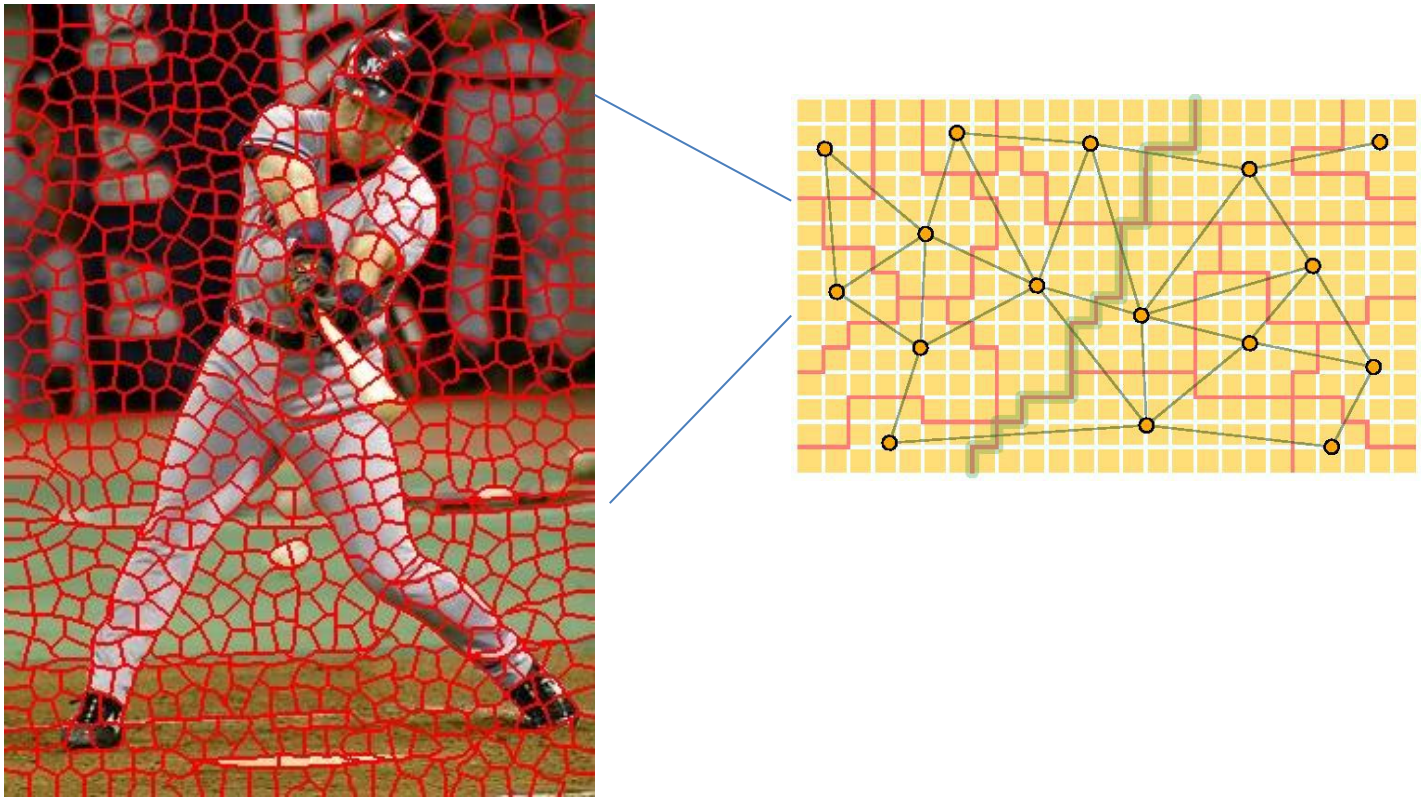
- Penalty Term for boundaries
  - Only nonzero if across segmentation boundary
  - Larger if adjacent pixels have similar colors

Problem

- Too slow for “real-time” requirement!

Problem

- Too slow for “real-time” requirement!

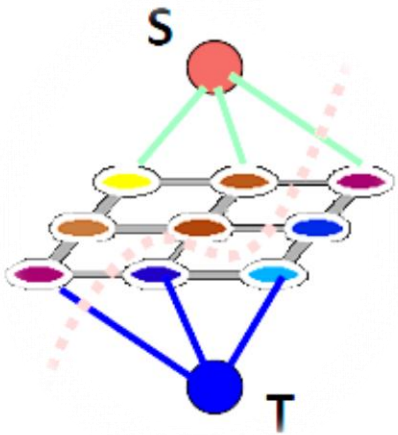


Pre-segmentation

- Graph Cut on segment instead of pixel level
- C_{ij}: the mean color difference between the two segments, weighted by the shared boundary length
- Speed Comparison

Image	Dimension	Nodes Ratio	Edges Ratio	Lag with Pre-segmentation	Lag without Pre-segmentation
Boy	$(408, 600)$	10.7	16.8	$0.12s$	$0.57s$
Ballet	$(440, 800)$	11.4	18.3	$0.21s$	$1.39s$
Twins	$(1024, 768)$	20.7	32.5	$0.25s$	$1.82s$
Girl	$(768, 1147)$	23.8	37.6	$0.22s$	$2.49s$
Grandpa	$(1147, 768)$	19.3	30.5	$0.22s$	$3.56s$

Graph Cuts



- 2 Labels: Find global minimum
  - Max-flow/min-cut algorithm
  - Fast
- Multi-Labels: computing the global minimum is NP-hard
  - Approximation algorithm(for some forms of energy function: smoothness energy term V which is metric or semi-metric)

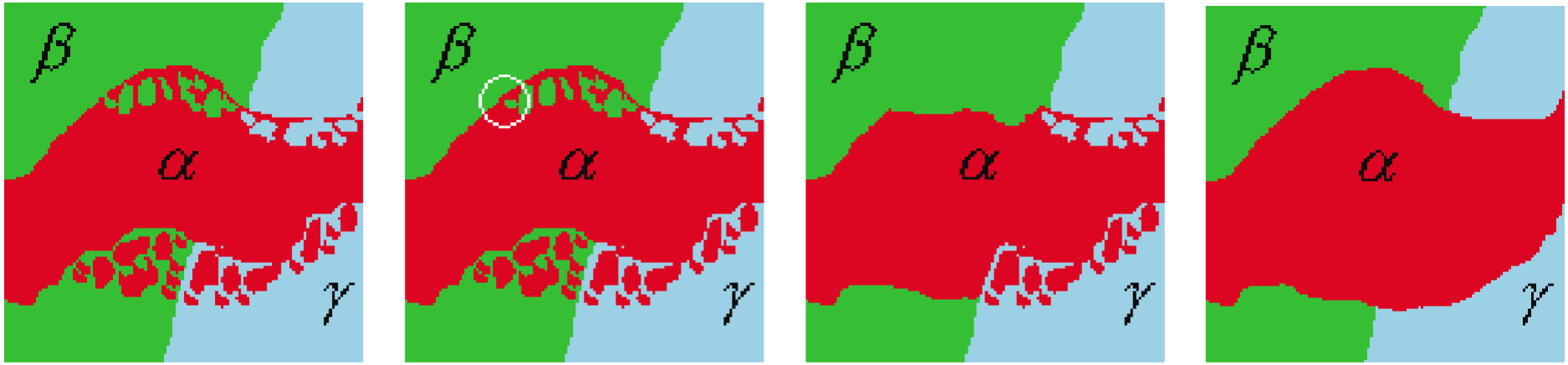
$$V(\alpha,\beta) = 0 \quad \Leftrightarrow \quad \alpha = \beta,$$ Identity

$$V(\alpha,\beta) \quad = \quad V(\beta,\alpha) \geq 0,$$ Symmetry &non-negativity

$$V(\alpha,\beta) \quad \leq \quad V(\alpha,\gamma) + V(\gamma,\beta),$$ Triangle inequality

Graph Cuts for multi labels

- α-expansion(d)
  - V is metric, within a known factor of global minimum
- αβ-swap(c)
  - V is semi-metric, local minimum



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

Graph Cut for multi labels

- αβ-swap algorithm

```
1.  Start with an arbitrary labeling f
2.  Set success := 0
3.  For each pair of labels {α, β} ⊂ ℒ
    3.1.  Find f̂ = arg min E(f′) among f′ within one α-β swap of f
    3.2.  If E(f̂) < E(f), set f := f̂ and success := 1
4.  If success = 1 goto 2
5.  Return f
```

- α-expansion algorithm

```
1.  Start with an arbitrary labeling f
2.  Set success := 0
3.  For each label α ∈ ℒ
    3.1.  Find f̂ = arg min E(f′) among f′ within one α-expansion of f
    3.2.  If E(f̂) < E(f), set f := f̂ and success := 1
4.  If success = 1 goto 2
5.  Return f
```

Graph Cut for multi labels

- αβ-swap algorithm

1. Start with an arbitrary labeling f
2. Set success := 0
3. For each pair of labels $\{\alpha, \beta\} \subset \mathcal{L}$
   3.1. Find $\hat{f} = \arg\min E(f')$ among f' within one α-β swap of f
   3.2. If $E(\hat{f}) < E(f)$, set $f := \hat{f}$ and success := 1
4. If success = 1 goto 2
5. Return f

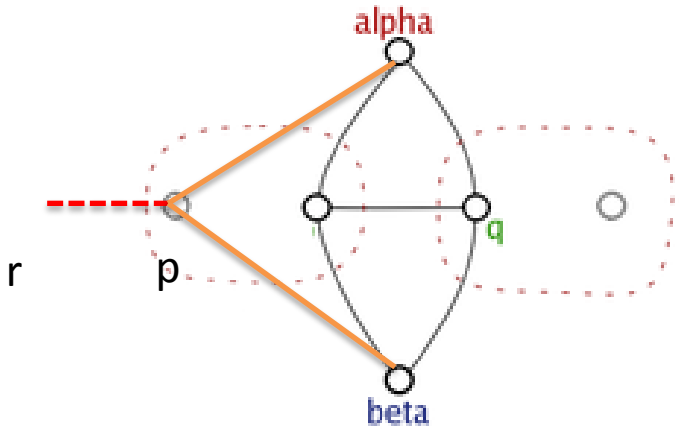
Min cut/
Max Flow

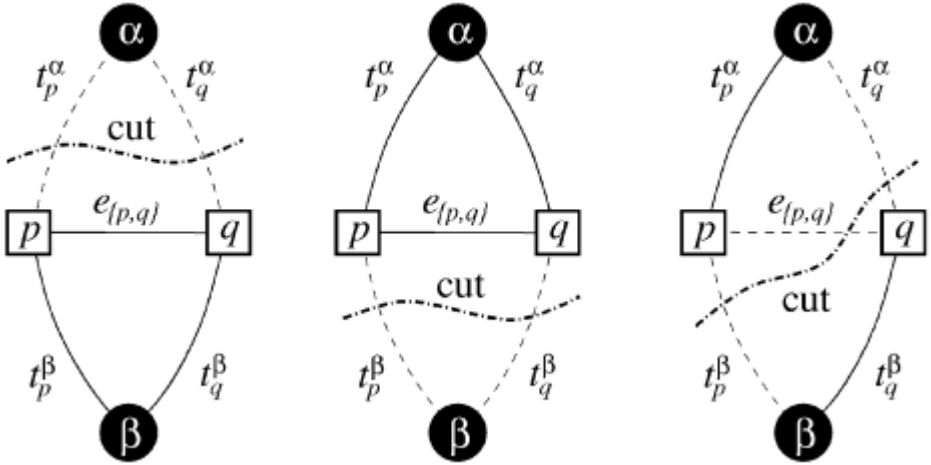
- α-expansion algorithm

1. Start with an arbitrary labeling f
2. Set success := 0
3. For each label $\alpha \in \mathcal{L}$
   3.1. Find $\hat{f} = \arg\min E(f')$ among f' within one α-expansion of f
   3.2. If $E(\hat{f}) < E(f)$, set $f := \hat{f}$ and success := 1
4. If success = 1 goto 2
5. Return f

αβ-swap algorithm

3.1. Find $\hat{f} = \arg\min E(f')$ among f' within one α-β swap of f

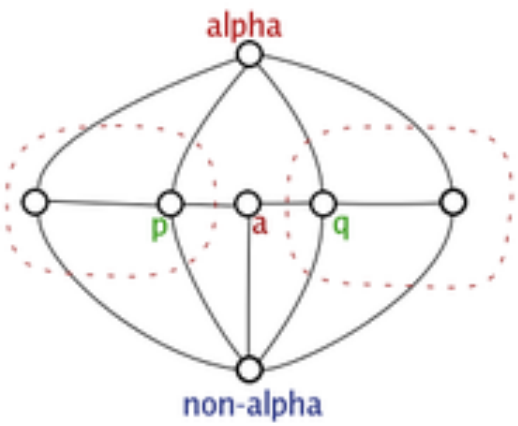


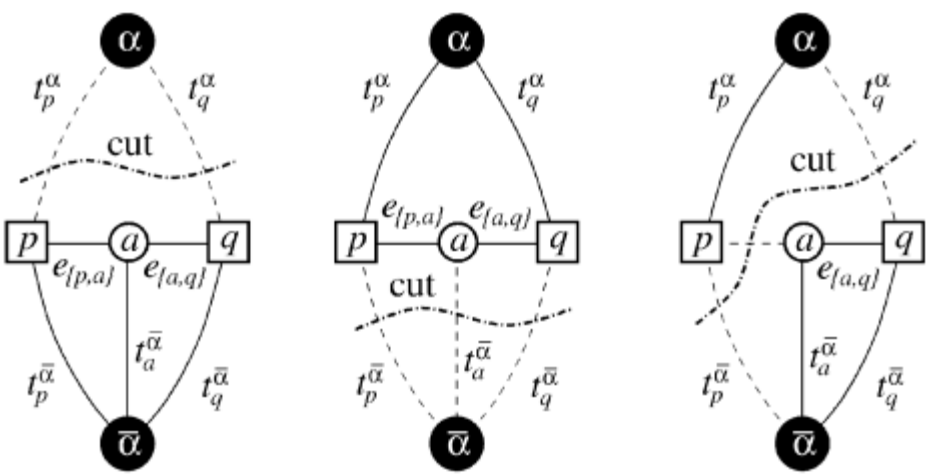


edge	weight	for
t_p^α	$D_p(\alpha) + \sum_{\substack{q \in \mathcal{N}_p \\ q \notin \mathcal{P}_{\alpha\beta}}} V(\alpha, f_q)$	$p \in \mathcal{P}_{\alpha\beta}$
t_p^β	$D_p(\beta) + \sum_{\substack{q \in \mathcal{N}_p \\ q \notin \mathcal{P}_{\alpha\beta}}} V(\beta, f_q)$	$p \in \mathcal{P}_{\alpha\beta}$
$e_{\{p,q\}}$	$V(\alpha, \beta)$	$\{p,q\} \in \mathcal{N}$ $p,q \in \mathcal{P}_{\alpha\beta}$

α-expansion algorithm

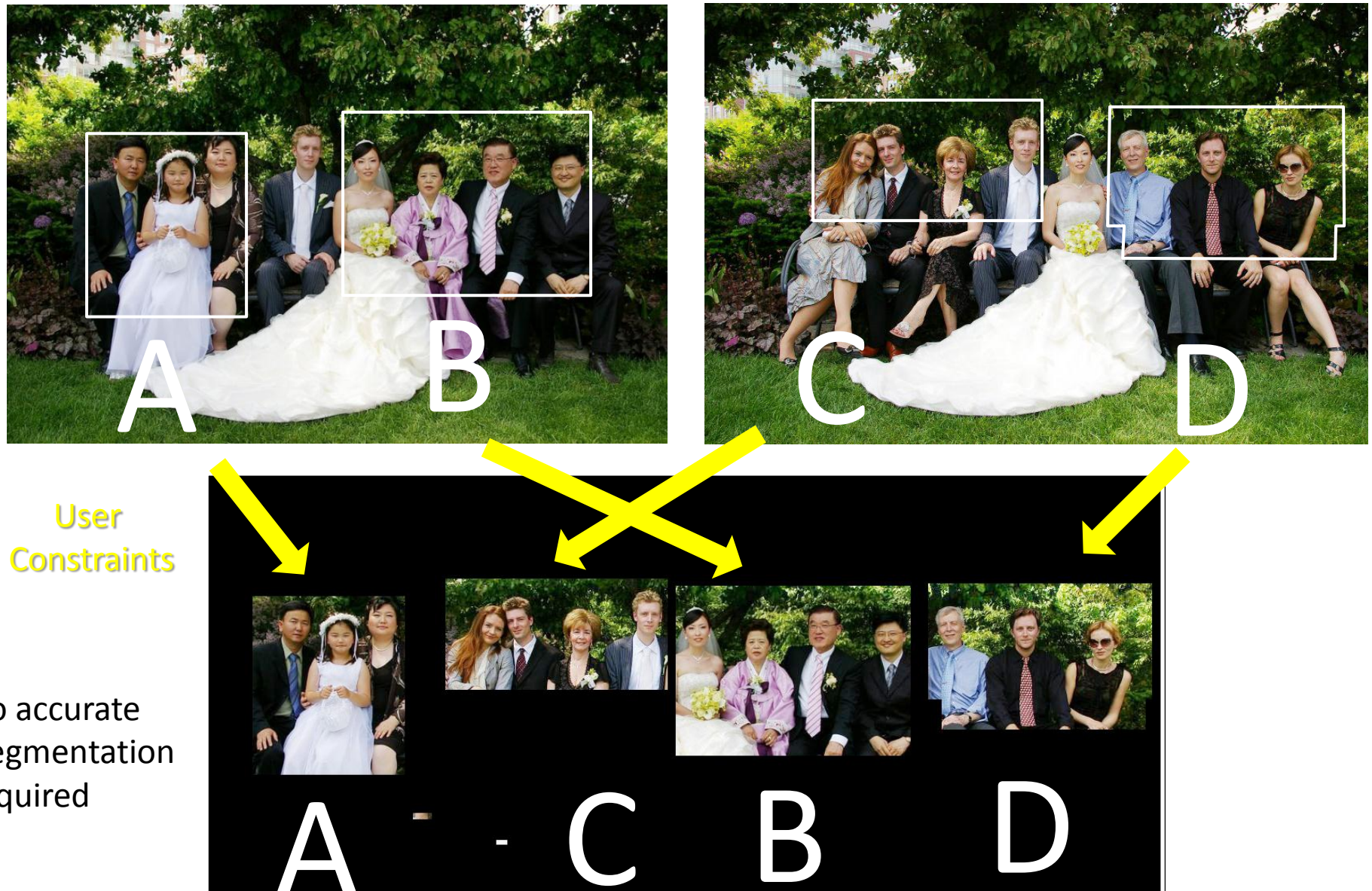
3.1. Find $\hat{f} = \arg\min E(f')$ among f' within one α-expansion of f





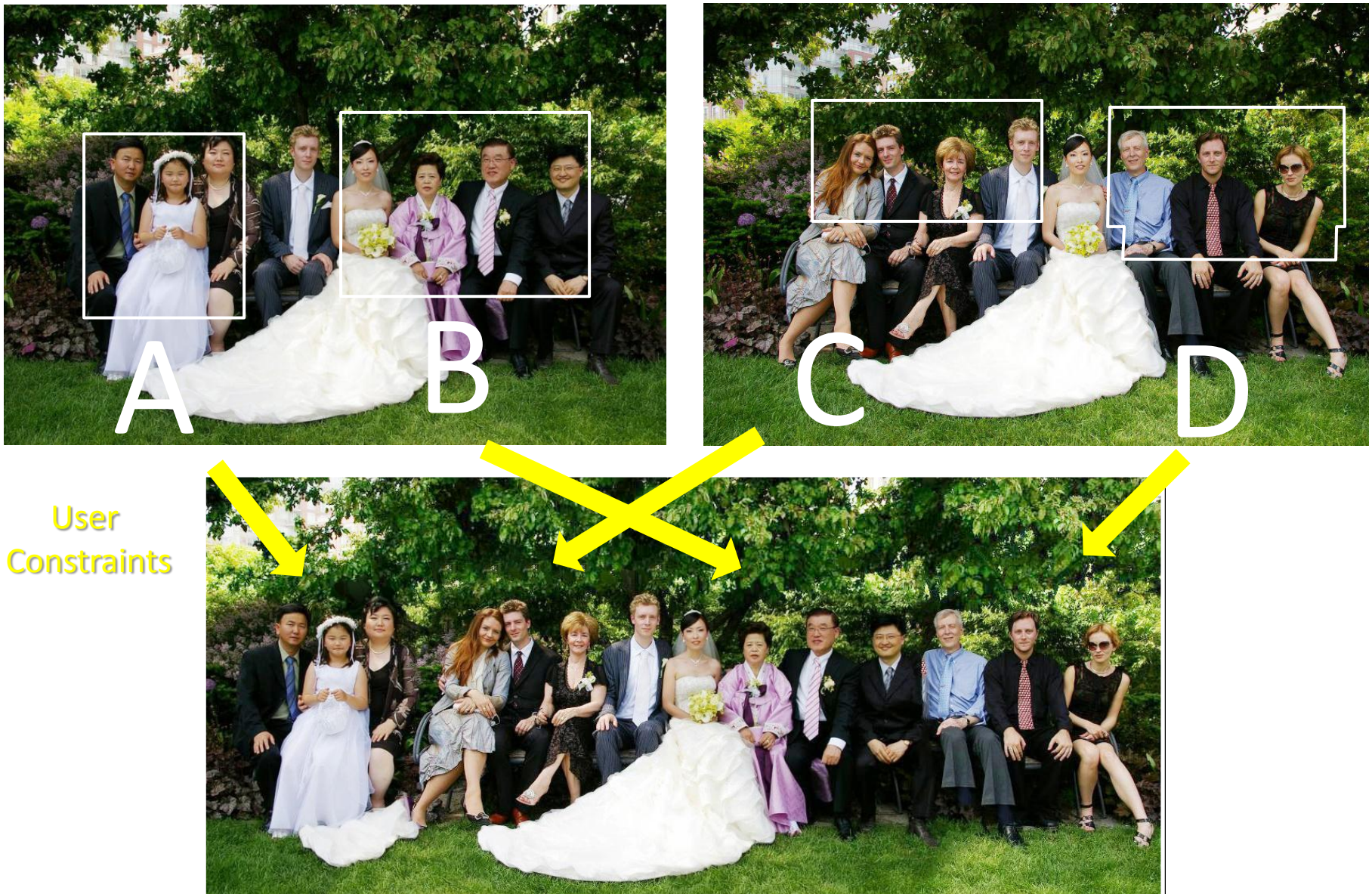
edge	weight	for
$t_p^{\bar\alpha}$	∞	$p \in \mathcal{P}_\alpha$
$t_p^{\bar\alpha}$	$D_p(f_p)$	$p \notin \mathcal{P}_\alpha$
t_p^{α}	$D_p(\alpha)$	$p \in \mathcal{P}$
$e_{\{p,a\}}$	$V(f_p, \alpha)$	$\{p,q\} \in \mathcal{N},\ f_p \neq f_q$
$e_{\{a,q\}}$	$V(\alpha, f_q)$	
$t_a^{\bar\alpha}$	$V(f_p, f_q)$	
$e_{\{p,q\}}$	$V(f_p, \alpha)$	$\{p,q\} \in \mathcal{N},\ f_p = f_q$

Multi-label Examples: Shift Map



No accurate segmentation required

Multi-label Examples: Shift Map



Multi-label Examples: Shift Map

Image completion/In-painting(object removing)

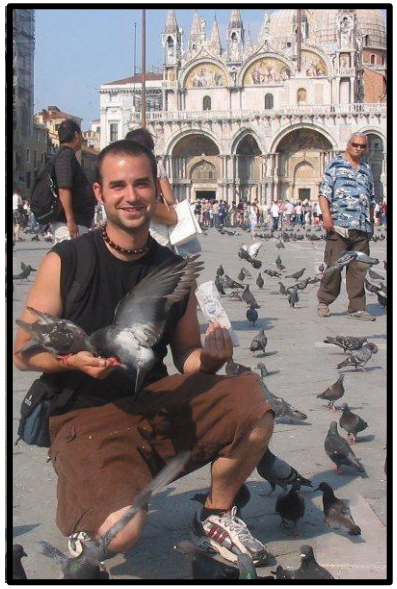
Input

User's mask

Output

Multi-label Examples: Shift Map

Retargeting(content-aware image resizing)



Output

Input

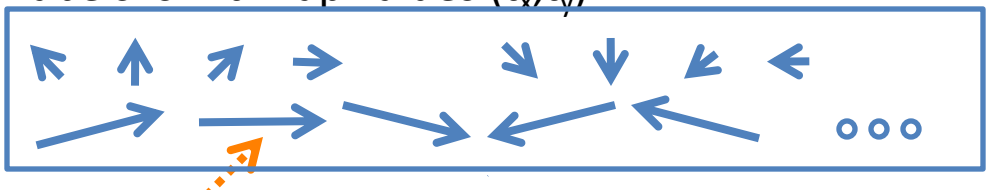
Multi-label Examples: Shift Map

- Label Set: relative mapping coordinate M

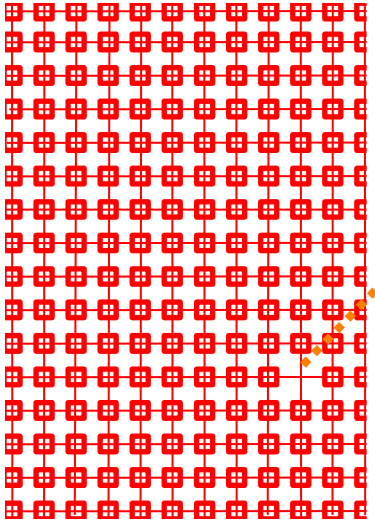
$$M(u,v) = (t_x, t_y)$$

$$R(u,v) = I(u+t_x, v+t_y) = I(x,y)$$

Labels: shift-map values (t_x,t_y)



Nodes:
pixels



assign
a label to
pixel

Output : R(u,v)

Input : I(x,y)

Multi-label Examples: Shift Map

- Energy function:

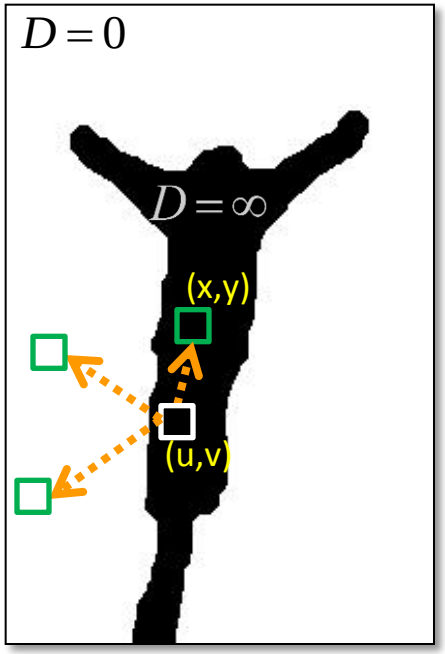
$$E(M) = \alpha \sum_{p \in R} E_d(M(p)) + \sum_{p,q \in N} E_s(M(p), M(q))$$

Data term :
External Editing Requirement

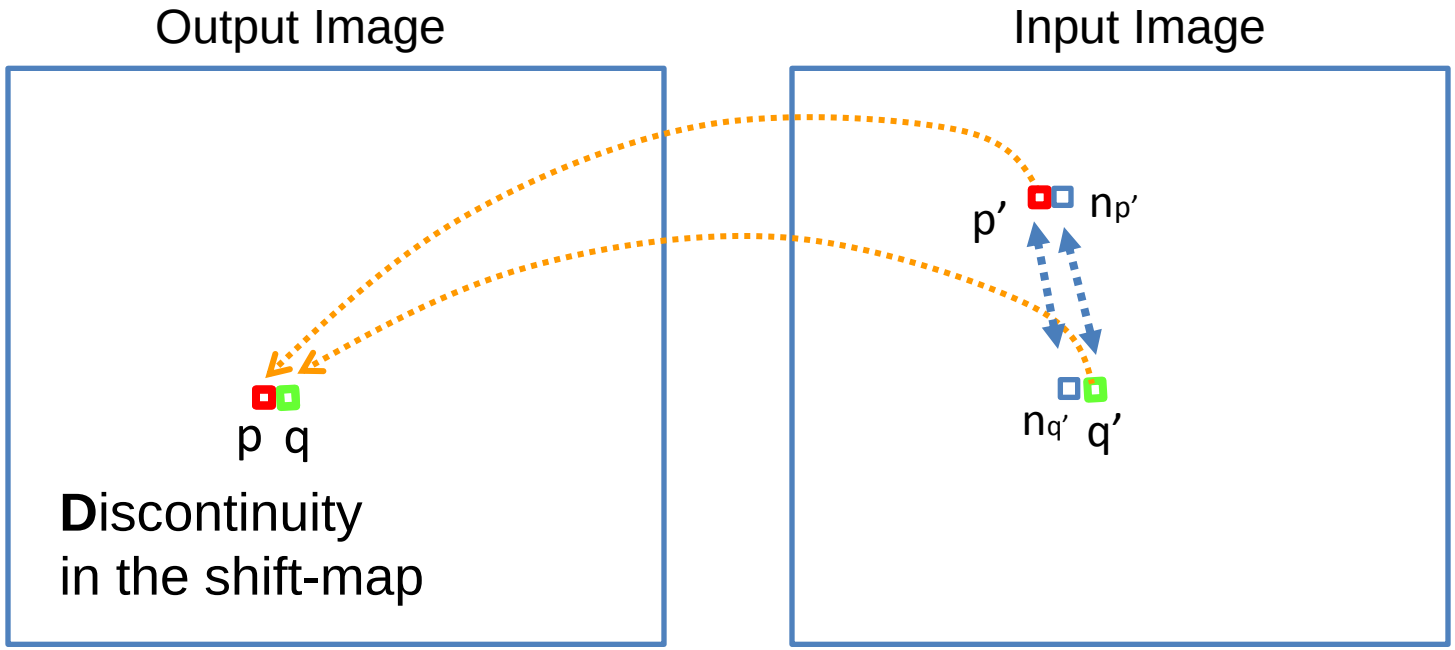
Smoothness term :
Avoid Stitching Artifacts

- Data term
  - varies between different application
  - Inpainting :

    Specific input pixels can be forced not to be included in the output image by setting *D(x,y)=∞*



Smoothness term



For p For q

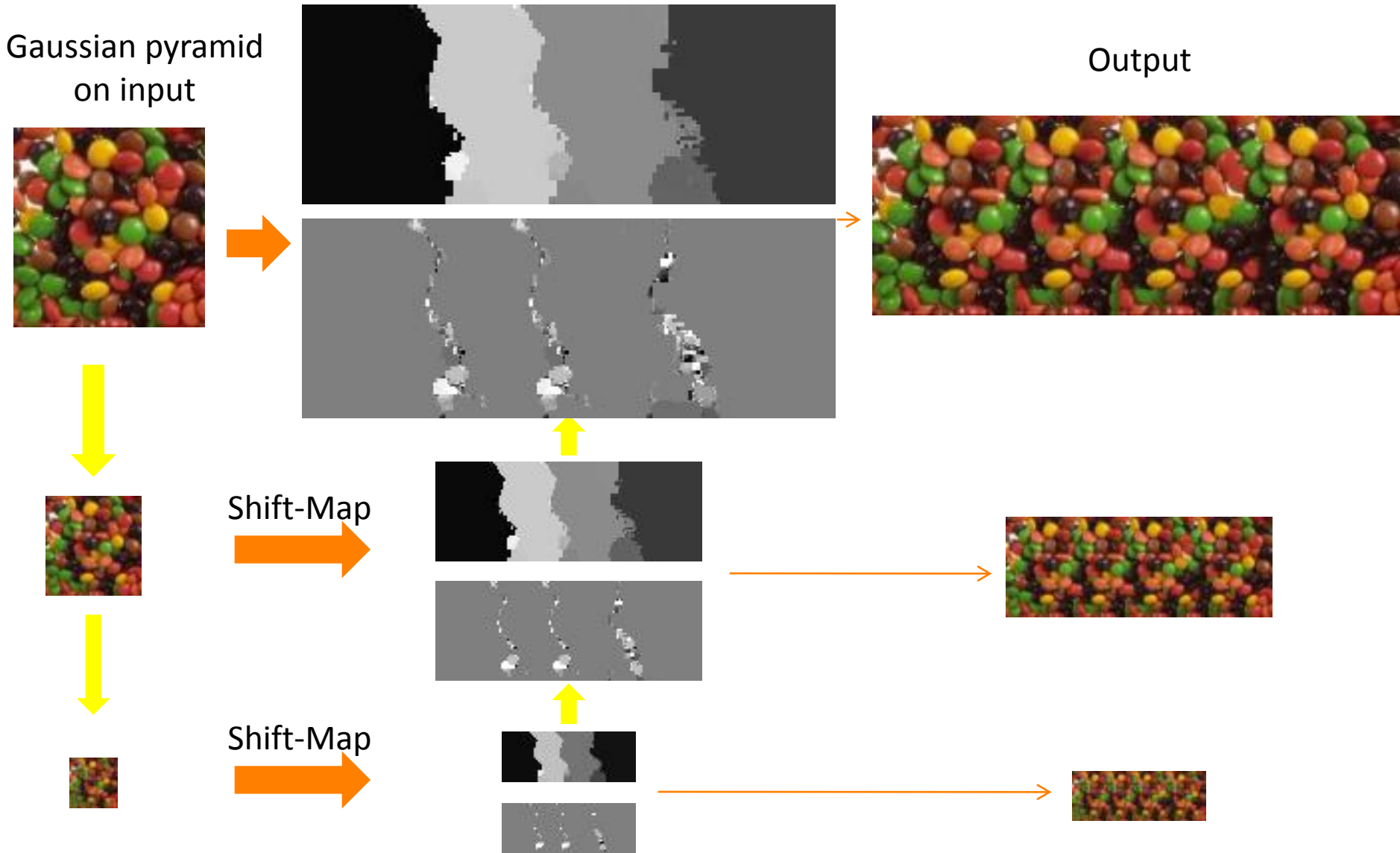
$$E_s(M(p),M(q)) = (I(n_{p'}) - I(q'))^2 + (I(n_{q'}) - I(p'))^2 + \quad \textit{color}$$

$$(\nabla I(n_{p'}) - \nabla I(q'))^2 + (\nabla I(n_{q'}) - \nabla I(p'))^2 \quad \textit{gradient}$$

Multi-label Examples: Shift map

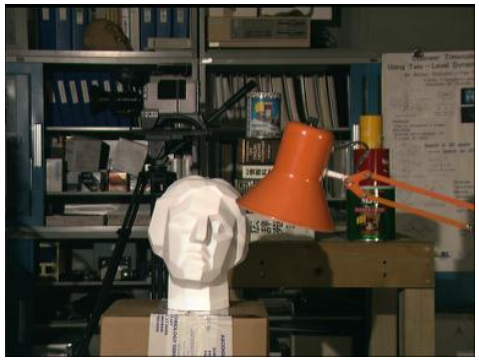
- Graph Cuts : α-expansion
- Why design label as disparity instead of absolute coordinate of input image?

Hierarchical Solution



Multi-label Examples: Dense Stereo

- Label set for each pixel:
  - disparity $d \in \{0, 1, .. D\}$



Left Camera Image

Right Camera Image

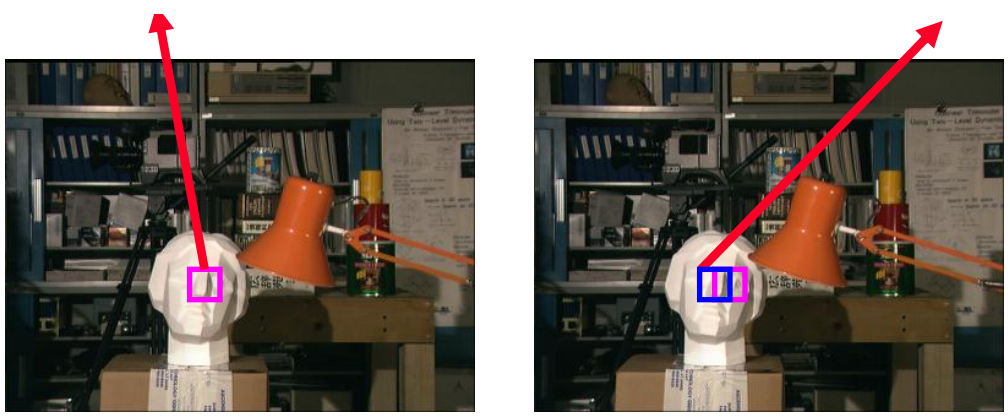
Dense Stereo Result

Multi-label Examples: Dense Stereo

- Data term: for pixel p, label d

$$C(p,d) = \left(\min\{C_{fwd}(p,d), C_{rev}(p,d), const\}\right)^2$$

$$C_{rev}(p,d) = \min_{p-\frac{1}{2}\le x\le p+\frac{1}{2}} |I_x - I'_{p+d}| \qquad C_{fwd}(p,d) = \min_{d-\frac{1}{2}\le x\le d+\frac{1}{2}} |I_p - I'_{p+x}|.$$



Left Camera Image Right Camera Image

Multi-label Examples: Dense Stereo

- Smoothness term for neighboring pixels p and q
  - $V_{p,q} = \min(K|f_p - f_q|, T)$
  - or,

$$V_{p,q} = u_{\{p,q\}} \cdot T(f_p \neq f_q)$$

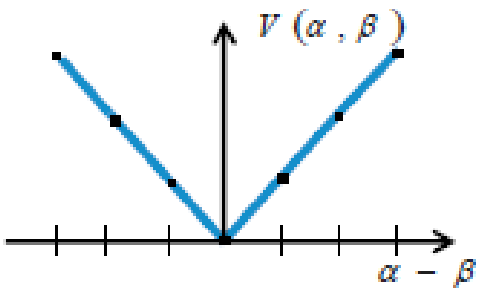
$$U(|I_p - I_q|) = \begin{cases} 2K & \text{if } |I_p - I_q| \leq 5 \\ K & \text{if } |I_p - I_q| > 5. \end{cases}$$

Design smoothness term V

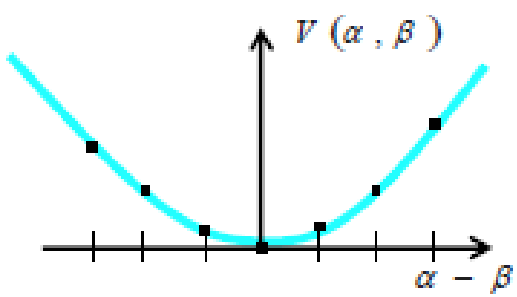
- Choices of V: not Robust

Better!

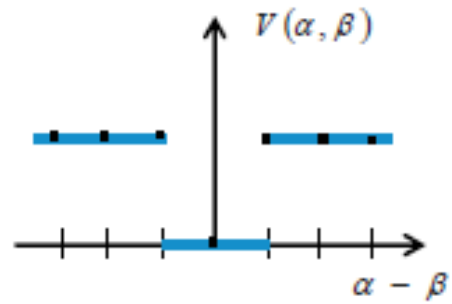
Linear model



Quadratic model

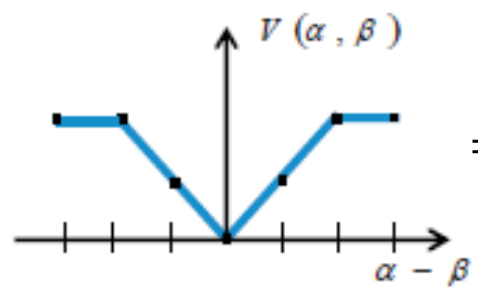


Potts model



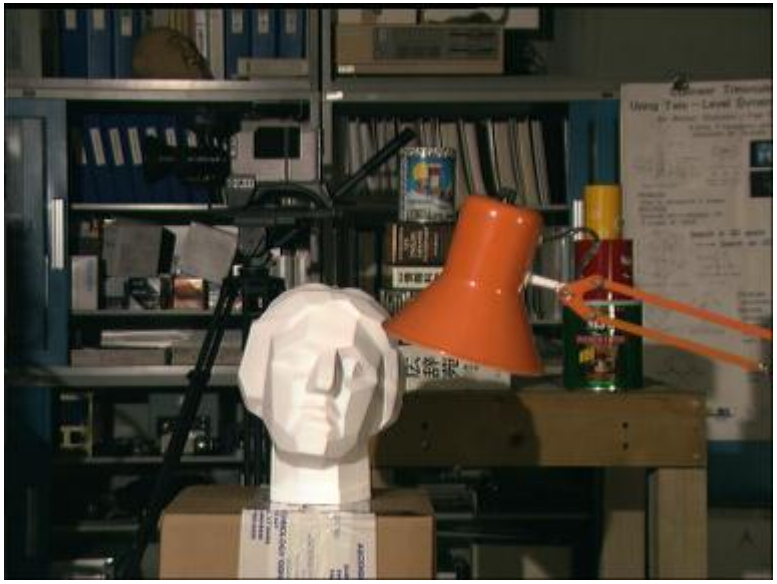
$= 1(|\alpha - \beta| > \mathrm{k})$

Truncated linear model



$= \min(|\alpha - \beta|, \mathrm{k})$

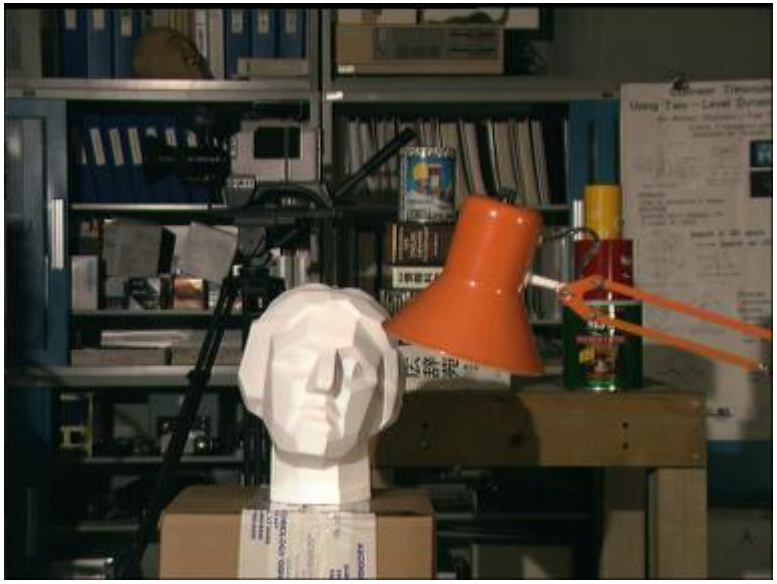
Results



Original Image

Initial Solution

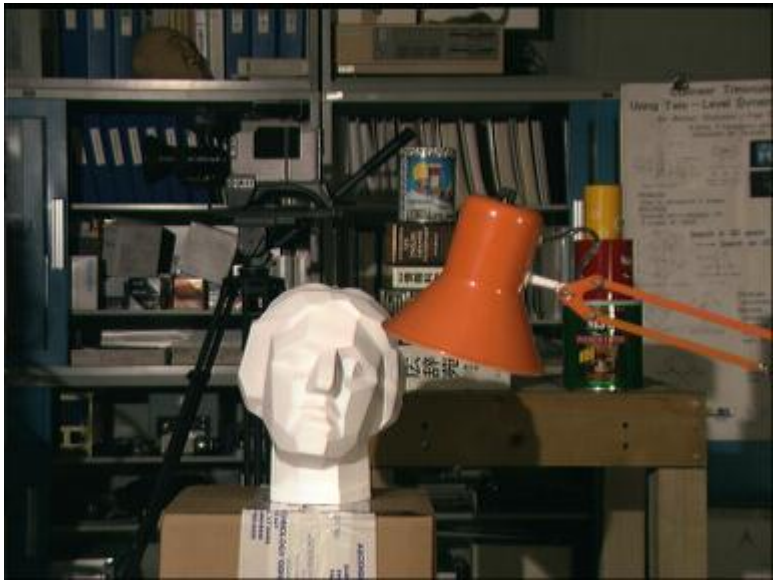
Results



Original Image

1st Expansion

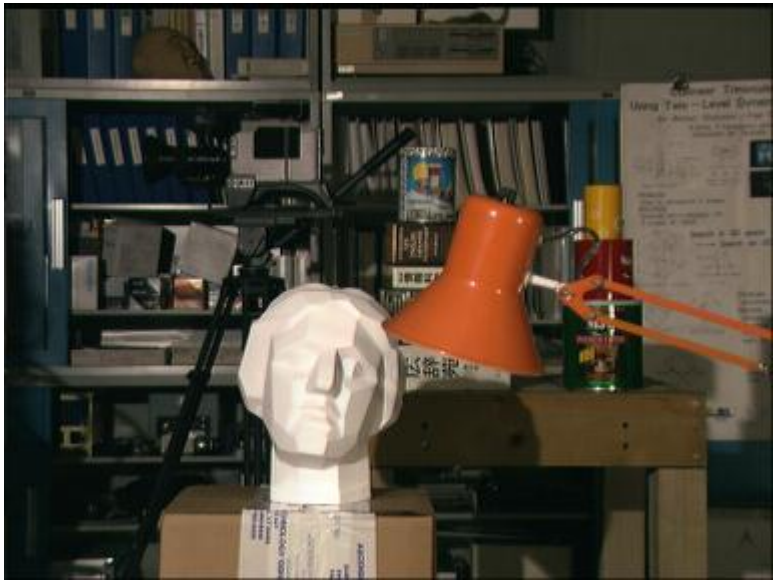
Results



Original Image

2nd Expansion

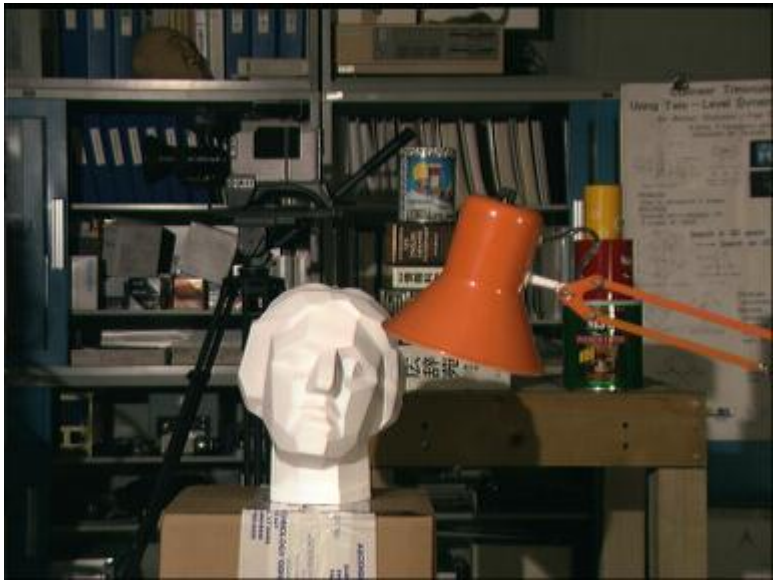
Results



Original Image

3rd Expansion

Results

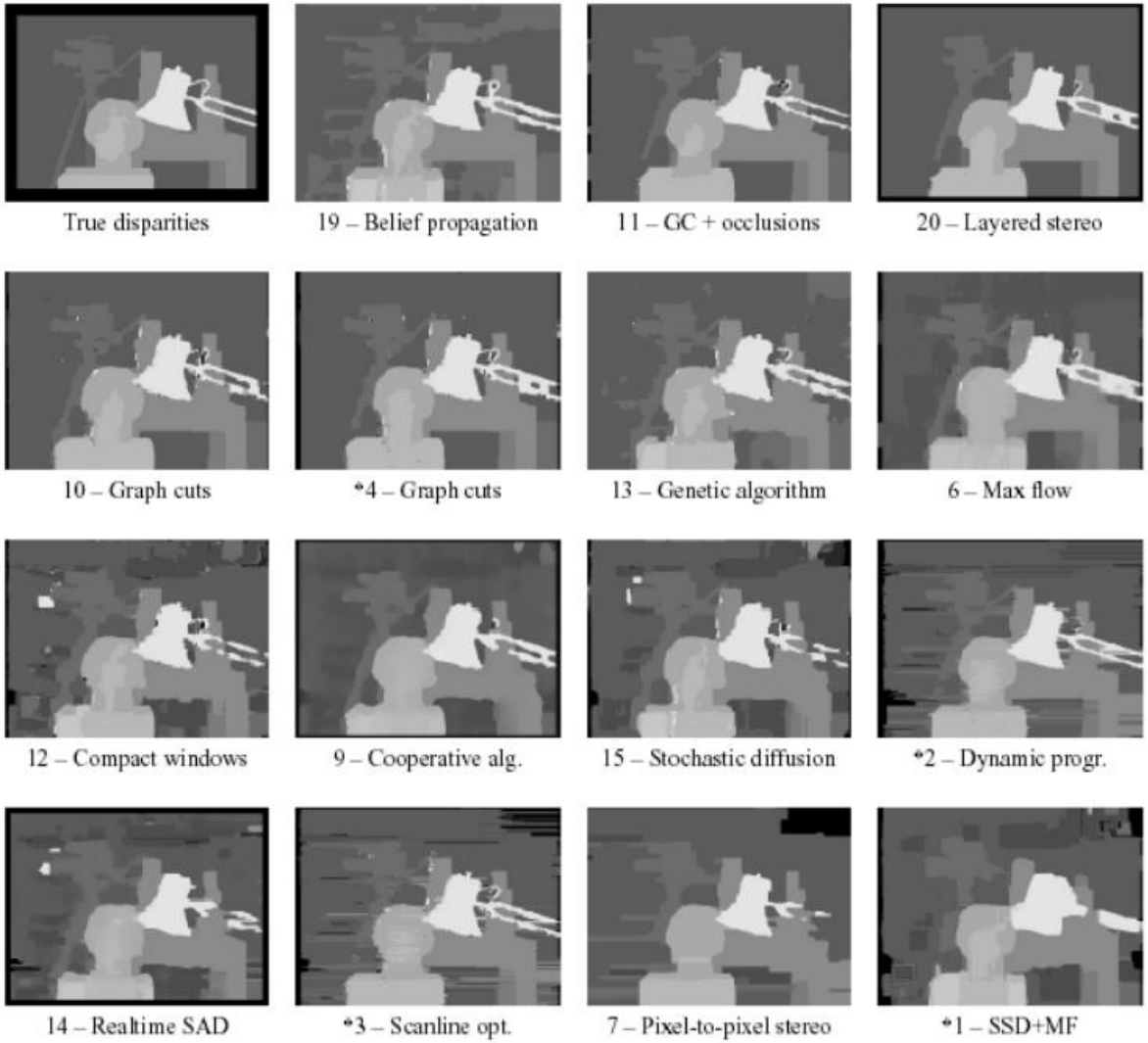


Original Image

Final expansion

Results

- http://vision.middlebury.edu/stereo/eval/



Comments on Graph Cuts

- In practice, GraphCut α-expansion algorithm usually outperforms $\alpha\beta$-swap method

- Limitations of GC algorithm:
  - Constraint on energy term
  - Speed

Belief Propagation(BP)

- **Belief Propagation** allows the marginals and maximizer to be computed efficiently on graphical models.
- **Sum-product** BP is a message passing algorithm that calculate the marginal distribution on a graphical model

$$P(x_1), P(x_2), \ldots, P(x_N)$$

- **Max-product** BP(or **max-sum** in log domain), is used to estimate the state configuration with maximum probability.

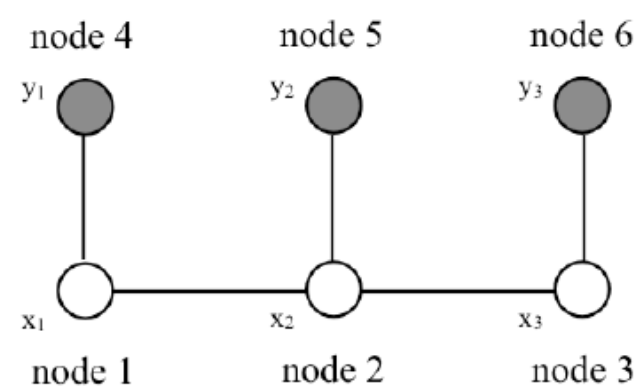
$$\arg \max_{x_1, x_2, \ldots x_N} P(x_1, x_2, \ldots, x_N)$$

- Exhaustive search O(|state|^N)

Sum-product BP

$$P(x_1|\vec{y}) = \sum_{x_2}\sum_{x_3} P(x_1, x_2, x_3|\vec{y})$$

$$\begin{aligned}
P(x_1|\vec{y}) &= \frac{1}{P(\vec{y})}\sum_{x_2}\sum_{x_3}\phi_{12}(x_1,x_2)\phi_{23}(x_2,x_3)\psi_1(y_1,x_1)\psi_2(y_2,x_2)\psi_3(y_3,x_3) \\
&= \frac{1}{P(\vec{y})}\psi_1(y_1,x_1)\sum_{x_2}\phi_{12}(x_1,x_2)\psi_2(y_2,x_2)\sum_{x_3}\phi_{23}(x_2,x_3)\psi_3(y_3,x_3) \\
&= \frac{1}{P(\vec{y})}\psi_1(y_1,x_1)\sum_{x_2}\phi_{12}(x_1,x_2)\psi_2(y_2,x_2)\sum_{x_3}\phi_{23}(x_2,x_3)m_{63}(x_3) \\
&= \frac{1}{P(\vec{y})}\psi_1(y_1,x_1)\sum_{x_2}\phi_{12}(x_1,x_2)\psi_2(y_2,x_2)m_{32}(x_2) \\
&= \frac{1}{P(\vec{y})}\psi_1(y_1,x_1)m_{21}(x_1)
\end{aligned}$$

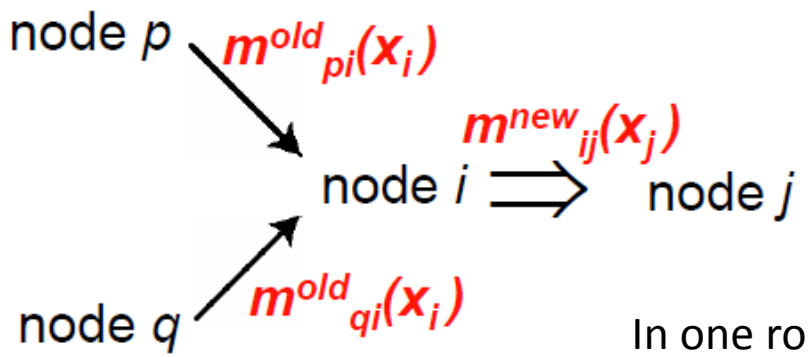


Messages

$$P(x_1, x_2, \ldots, x_N) = \frac{1}{Z} \prod_{i=1}^{N} g_i(x_i) \prod_{<ij>} f_{ij}(x_i, x_j)$$

- Initialize all messages to uniform (any non-negative)
- Update: message from *i* to *j*, consider all messages flowing into *i* (except for message from *j*):

$$m_{ij}^{new}(x_j) = \sum_{x_i} f_{ij}(x_i, x_j) g_i(x_i) \prod_{k \in Nbd(i) \setminus j} m_{ki}^{old}(x_i)$$



In one round: O(*#edges #states)*

Noisy neighbor rule

- BP follows the “noisy neighbor rule” (from Brendan Frey, U. Toronto).
- Every node is a house in some neighborhood.
- A message: The noisy neighbor says “given everything I’ve heard, here’s what I think is going on inside your house.”

Belief Propagation

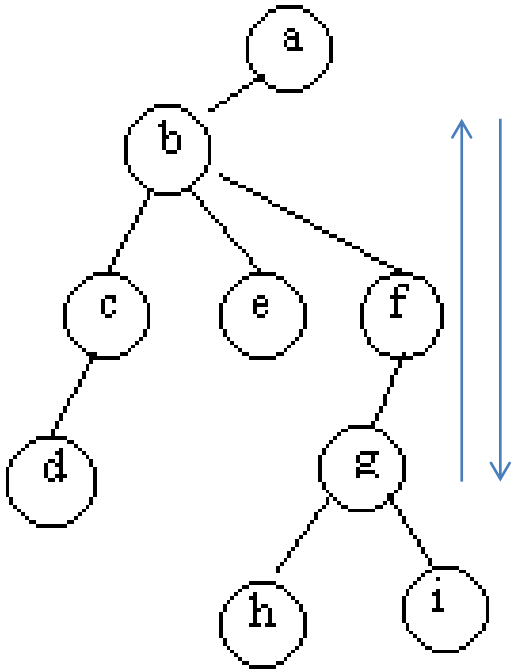
- Once messages have converged, output the normalized belief b as the marginal

$$b_i(x_i) \propto g(x_i) \prod_{k \neq i, k \in N(i)} m_{ki}^t(x_i)$$

- Max-product: x* that maximizes $b_i(x_i)$ is individually selected for each node is the maximal configuration(compute maximum during update)

Comments on Belief Propagation

- BP gives exact solutions when the graph is a tree (ie. has no loops), but only approximates the truth in loopy graphs(no convergence guarantee)
- Scheduling
  - For a tree or chain, with proper scheduling of the message updates, it will terminate after 2 steps.
  - For a grid (e.g. stereo on pixel lattice), people often sweep in an “up-down-left-right fashion” [Tappen & Freeman]

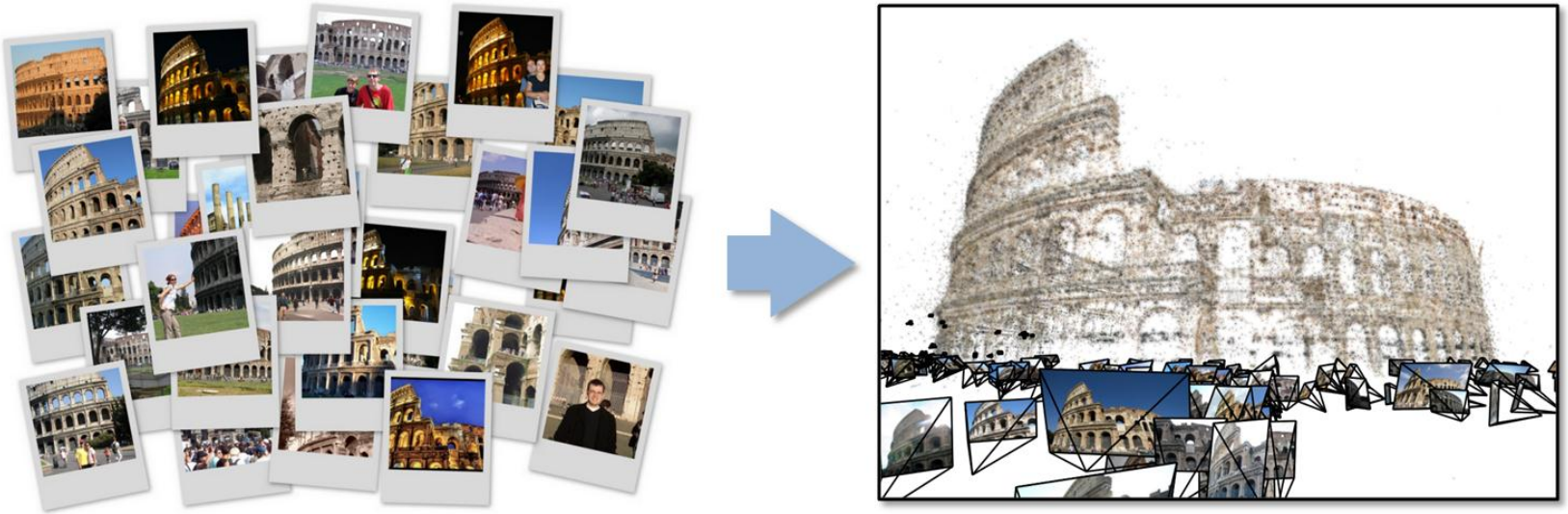


Speed-ups

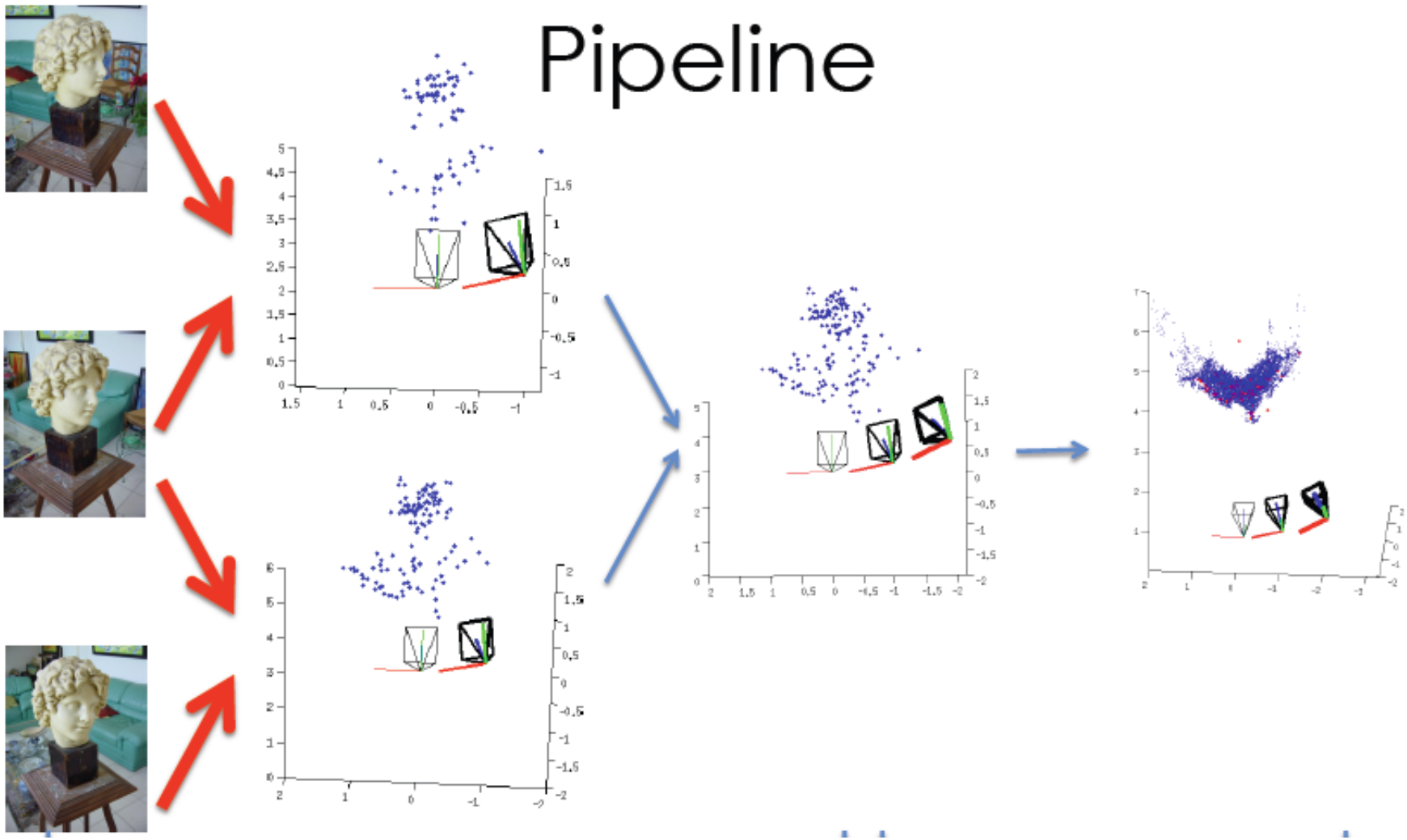
- Binary variables: Use log ratios [Mackay]
- Distance transform and multi-scale [Felzenszwalb & Huttenlocher]
- Sparse forward-backward [Pal et al]
- Dynamic quantization of state space [Coughlan & Shen]
- Higher-order factors with linear interactions [Potetz & Lee]
- GPU [Brunton et al]

Large-scale Structure from Motion

- Recent work has built 3D models from large, unstructured online image collections
  - [Snavely06], [Li08], [Agarwal09], [Frahm10], Microsoft's PhotoSynth, …



Pipeline



Two-view reconstruction Incremental Bundle adjustment Surface Reconstruction

From Fisher Yu's Lecture

Incremental Bundle Adjustment

Incremental BA

- ✅ Works very well for many scenes

- ❌ Poor scalability, much use of bundle adjustment
- ❌ Poor results if a bad seed image set is chosen
- ❌ Drift and bad local minima for some scenes

Reconstruction pipeline

Feature detection

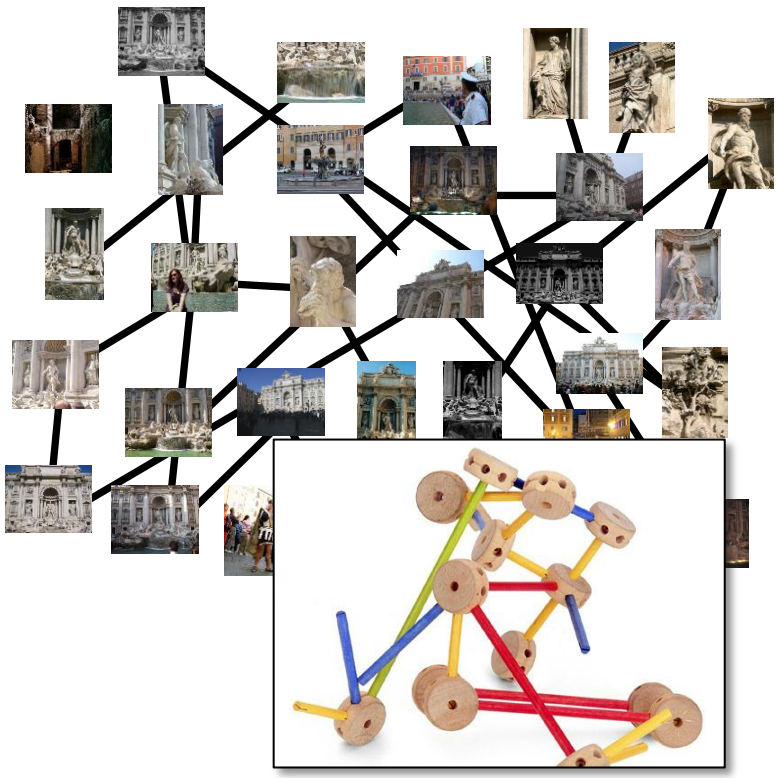
Feature matching
Find scene points seen by multiple cameras

Initialization
Robustly estimate camera poses and/or scene points

Bundle adjustment
Refine camera poses **R, T** and scene structure **P**

-Structure from motion-
Examples: Optimizing MRF using BP

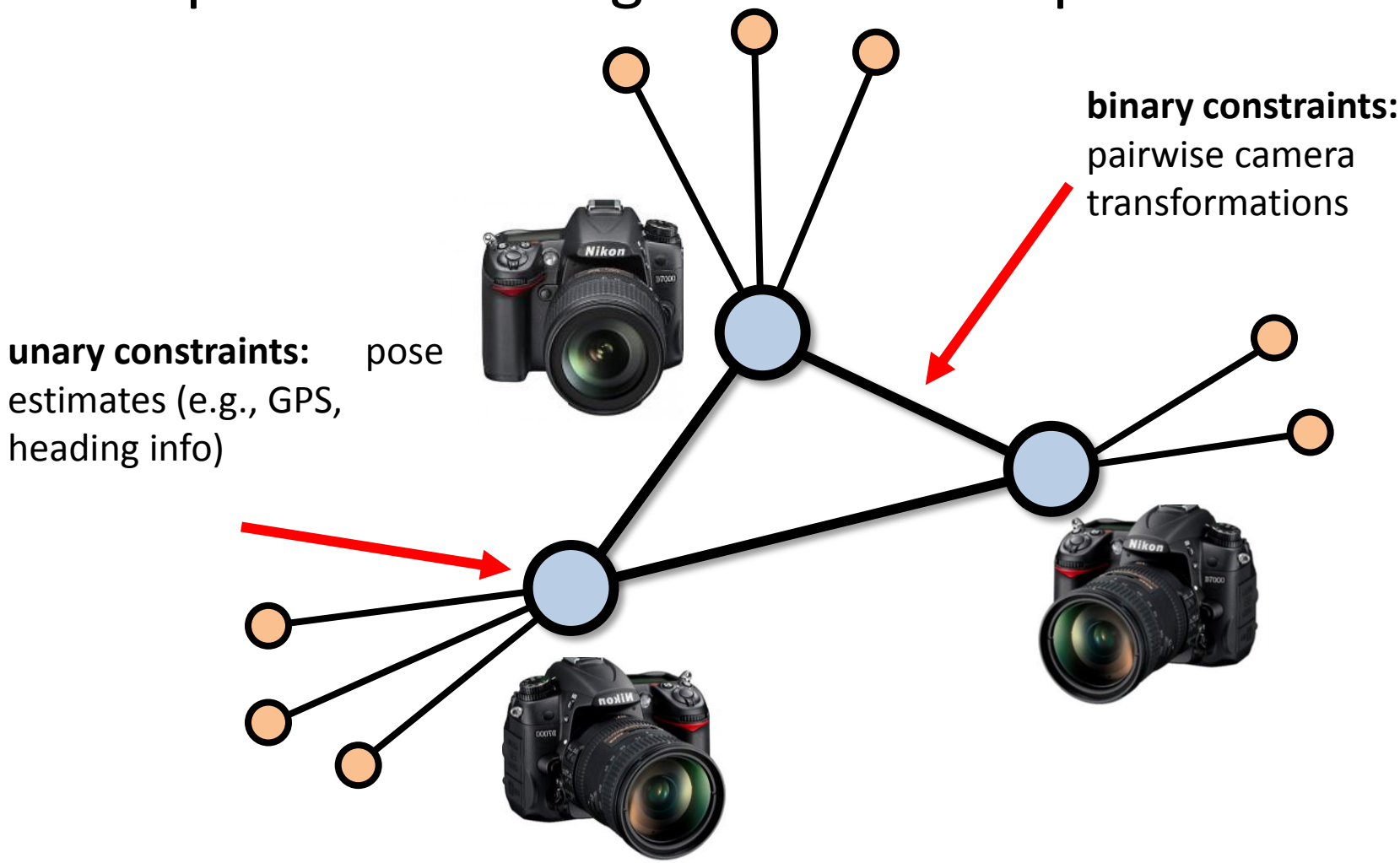
- View SfM as inference over a Markov Random Field, solving for all camera poses at once



  - Vertices are images (or points)
  - Inference problem: label each image with a camera pose, such that constraints are satisfied
  - Initializes all cameras at once
  - Can avoid local minima
  - Easily parallelized
  - Up to **6x speedups** over IBA for large problems

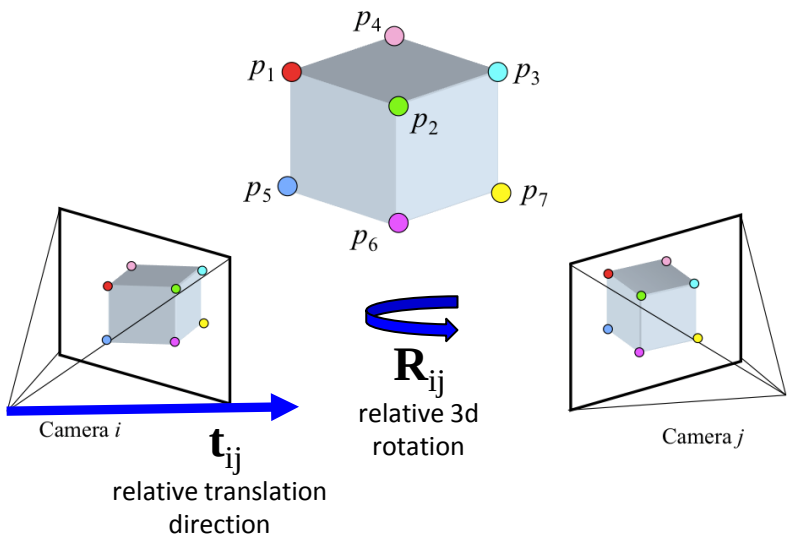
The MRF model

- Input: set of images with correspondence



Constraints on camera pairs

- Recall how to compute relative pose between camera pairs using 2-frame SfM



- Find absolute camera poses ($\mathbf{R}_i$, $\mathbf{t}_i$) and ($\mathbf{R}_j$, $\mathbf{t}_j$) that agree with these pairwise estimates:

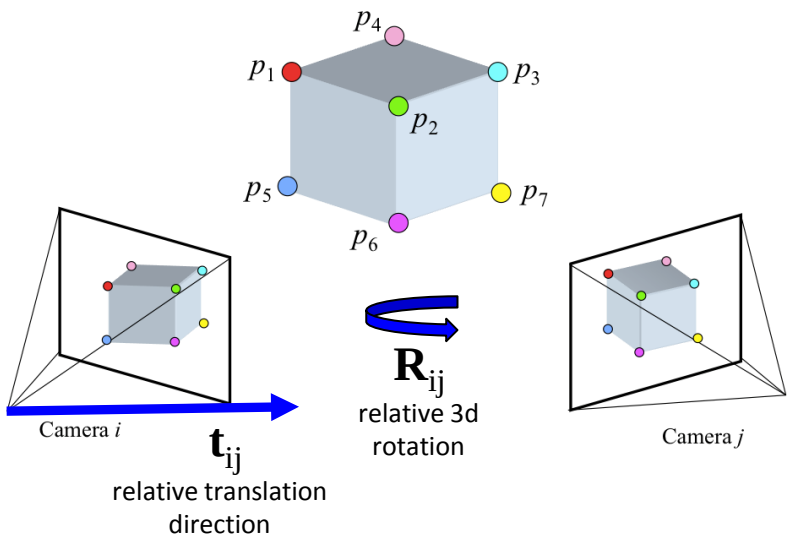
$$\mathbf{R}_{ij} = \mathbf{R}_i^\top \mathbf{R}_j$$

rotation consistency

$$\lambda_{ij}\mathbf{t}_{ij} = \mathbf{R}_i^\top(\mathbf{t}_j - \mathbf{t}_i)$$

translation direction consistency

Constraints on camera pairs



- Define robust error functions to use as pairwise potentials:

$$d^{\mathbf{R}}(\mathbf{R}_{ij}, \mathbf{R}_i^\top \mathbf{R}_j)$$

$$d^{\mathbf{R}}(\mathbf{R}_a, \mathbf{R}_b) = \rho_R(||\mathbf{R}_a - \mathbf{R}_b||)$$

rotation consistency ρ_R *is* truncated quadratic

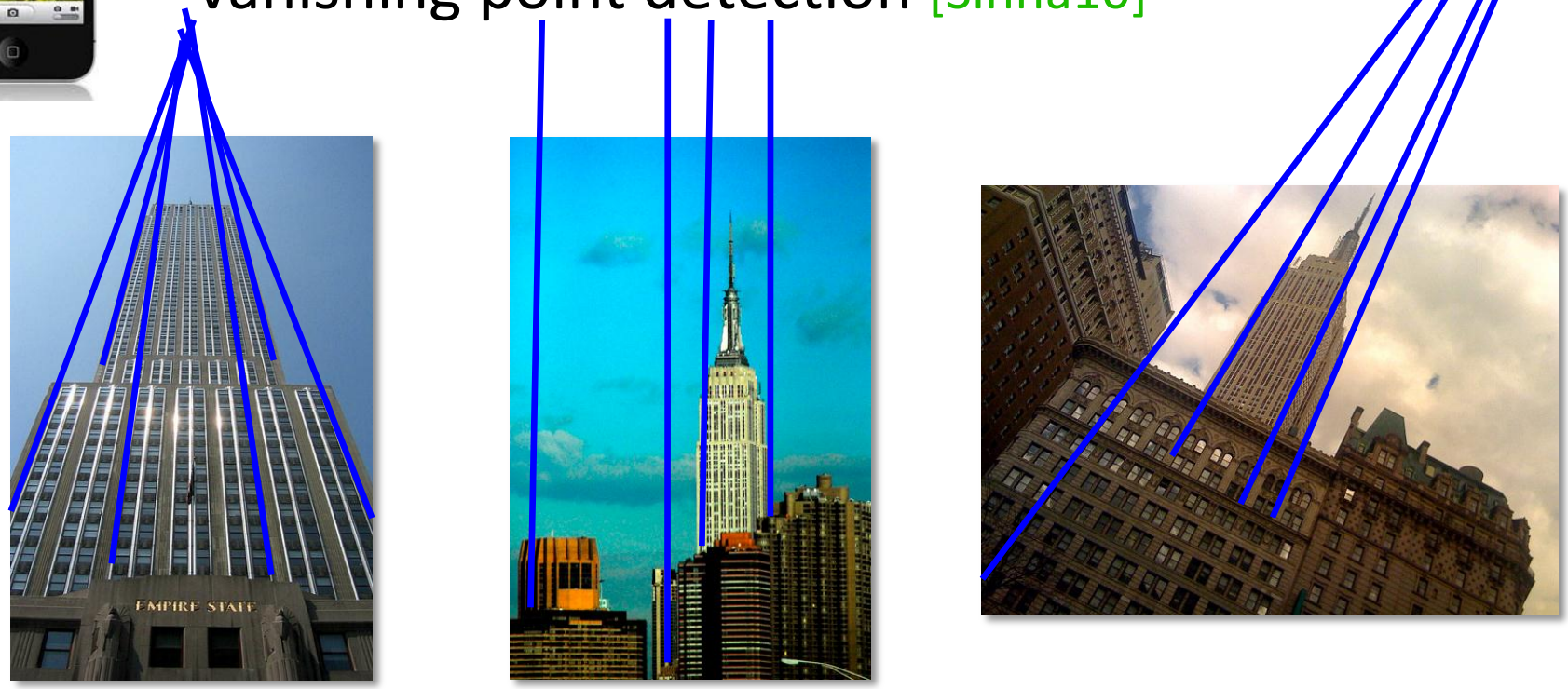
$$d^{\mathbf{T}}(\mathbf{t}_j - \mathbf{t}_i, \mathbf{R}_i \mathbf{t}_{ij})$$

$$d^{\mathbf{T}}(\mathbf{t}_a, \mathbf{t}_b) = \rho(\text{angleof}(\mathbf{t}_a, \mathbf{t}_b))$$

translation direction consistency

Data term: pose information

- *Noisy* absolute pose info for some cameras
  - 2D positions from geotags (GPS coordinates)
  - Orientations (tilt & twist angles) from vanishing point detection [Sinha10]



Prior pose information

- *Noisy* absolute pose info for some cameras
  - 2D positions from geotags (GPS coordinates)
  - Orientations (tilt & twist angles) from vanishing point detection

Rotations unary potential

$$d_i^{\mathbf{O}}(\mathbf{R}_i) = d_i^{\theta}(\mathbf{R}_i) + d_i^{\psi}(\mathbf{R}_i) + d_i^{\phi}(\mathbf{R}_i)$$

d^{ϑ}, d^{φ}, d^{ϕ} measure robust error *wrt* prior pose estimate

Translations unary potential

$$d_i^{\mathbf{G}}(\mathbf{t}_i) = \rho_T(||\,\mathrm{en}(\mathbf{g}_i) - \pi(\mathbf{t}_i)||)$$

$\mathbf{g}_i$ is a GPS coordinate
en & π project into a common coordinate system
ρ_T is a robust distance function

Overall optimization problem

- Given pairwise and unary pose constraints, solve for absolute camera poses simultaneously
  - for *n* cameras, estimate

    $\mathcal{R}$ = ($\mathbf{R}_1$, $\mathbf{R}_2$, …, $\mathbf{R}_n$) and $\mathcal{T}$ = ($\mathbf{t}_1$, $\mathbf{t}_2$, …, $\mathbf{t}_n$)

    so as to minimize total error over the entire graph

$$D^{\mathbf{R}}(\mathcal{R}) = \sum_{e_{ij} \in E_C} \boxed{d^{\mathbf{R}}\left(\mathbf{R}_{ij}, \mathbf{R}_i^\top \mathbf{R}_j\right)} + \alpha_1 \sum_{I_i \in \mathcal{I}} \boxed{d_i^{\mathbf{O}}(\mathbf{R}_i)}$$

pairwise rotation consistency **unary rotation consistency**

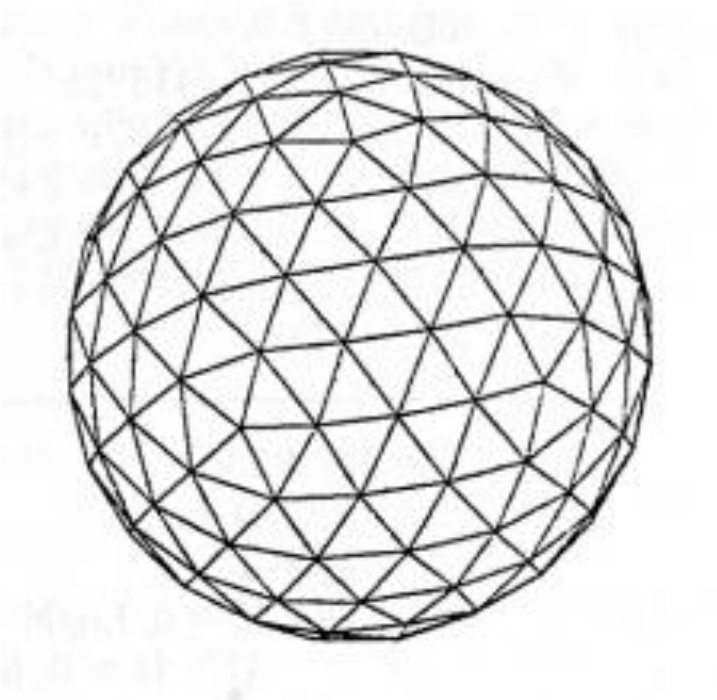
$$D^{\mathbf{T}}(\mathcal{T}, \mathcal{R}) = \sum_{e_{ij} \in E_C} \boxed{d^{\mathbf{T}}(\mathbf{t}_j - \mathbf{t}_i, \mathbf{R}_i \mathbf{t}_{ij})} + \alpha_2 \sum_{I_i \in \mathcal{I}} \boxed{d_i^{\mathbf{G}}(\mathbf{t}_i)}$$

pairwise translation consistency **unary translation consistency**

Solving the MRF

- Use discrete loopy belief propagation
  - Reduced by solving Rotation and Translation separately
  - Up to 1,000,000 nodes (cameras and points)
  - Up to 5,000,000 edges (constraints between cameras and points)

Discrete BP: Rotations

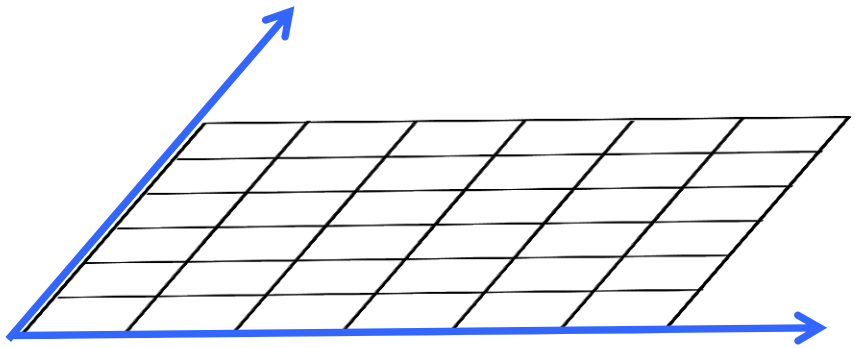


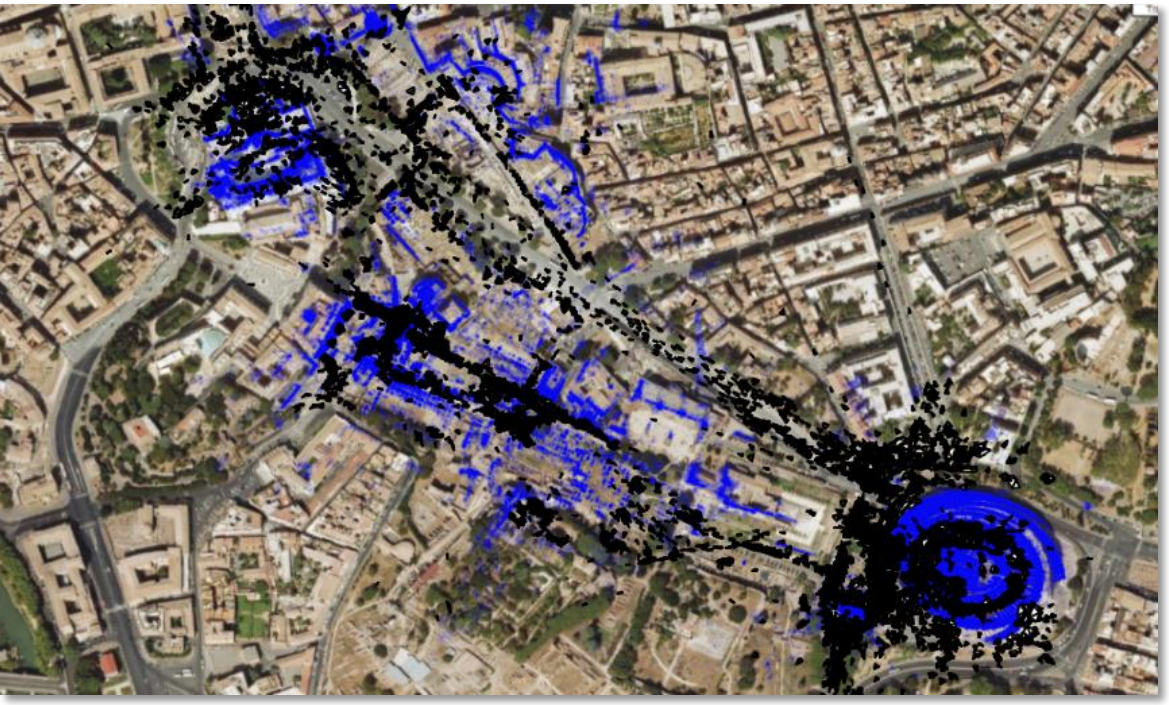
- **Uniform Grid on the sphere:** http://vision.princeton.edu/code.html#icosahedron

- Parameterize viewing directions as points on unit sphere
  - Discretize into 10x10x10 = 1,000 possible labels
  - Measure rotational errors as robust Euclidean distances on sphere (to allow use of distance transform)

Discrete BP: Translations

- Parameterize positions as 2D points in plane
  - Use approximation to error function (to allow use of distance transforms)
  - Discretize into up to 300 x 300 = 90,000 labels





Central Rome

Reconstructed images: 14,754

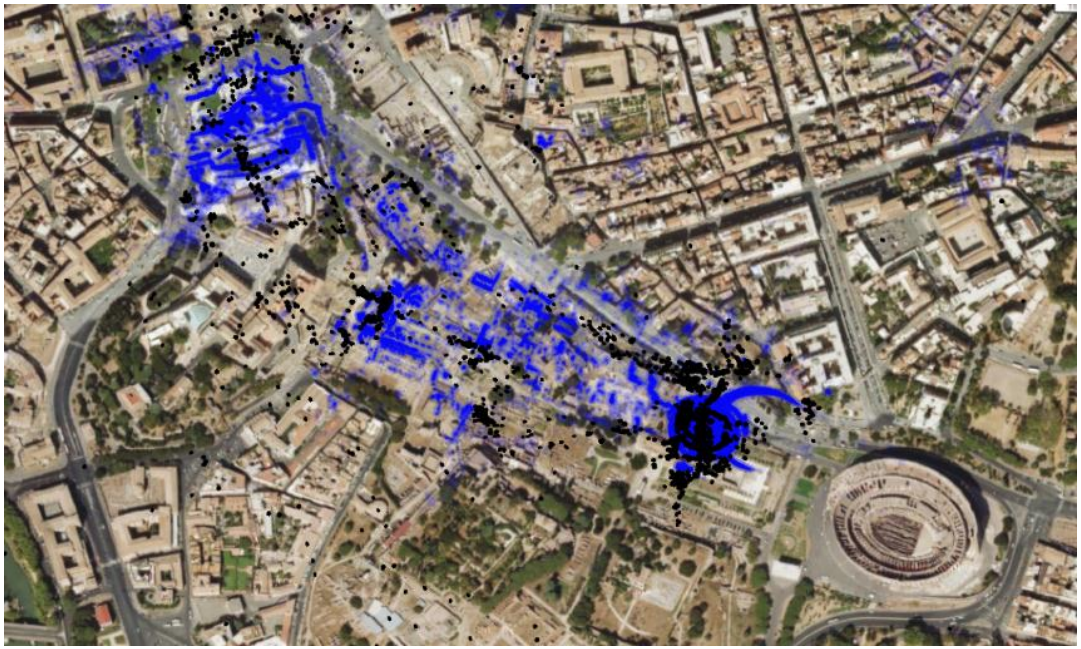
Edges in MRF: 2,258,416

Median camera pose difference wrt IBA: 25.0m

Our result

Dataset	Our approach	Incremental BA
Acropolis	0.2 hours	0.5 hours
Quad	7.7 hours	62 hours
Dubrovnik	5.5 hours	28 hours
CentralRome	13.2 hours	82 hours

Incremental Bundle Adjustment [Agarwal09]



Markov Random Fields

- In MRF: $E(L \mid X) = E(X \mid L) + E(L)$

  posterior energy likelihood energy prior energy

- MRF assumes that the prior distribution is independent of the measurements
  - Specifically, the prior(smoothness) energy E(L) is independent of the observations X
- Sometimes, we hope each variable is based on a set of global observation data
  - No pure prior knowledge

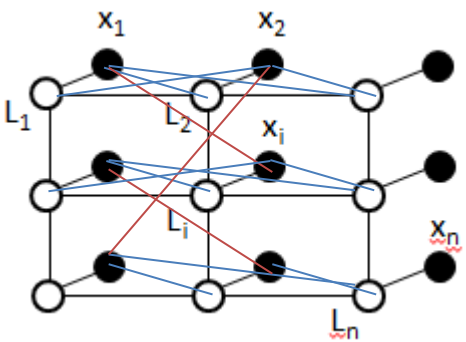
Why Conditional Random Fields?

- Object segmentation/recognition:
- Label set = {tree, people, water, sky, car}

Conditional Random Field

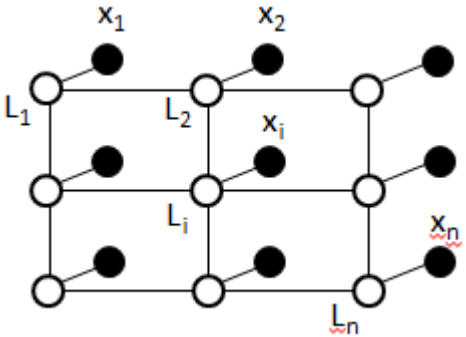
- Given observations X, (X, L) is said to be a Conditional Random Field (CRF) if, the random variables/labels L obey the Markov property with respect to the graph:

$$\mathrm{p}\left(L_i \middle| X, L_{j\neq i}\right) = p\left(L_i \middle| X, L_{neighbor(i)}\right)$$



- In MRF, we have very strict independence assumptions :

$$\mathrm{p}\left(L_i \middle| L_{j\neq i}\right) = p\left(L_i \middle| L_{neighbor(i)}\right)$$



CRF in supervised training problem

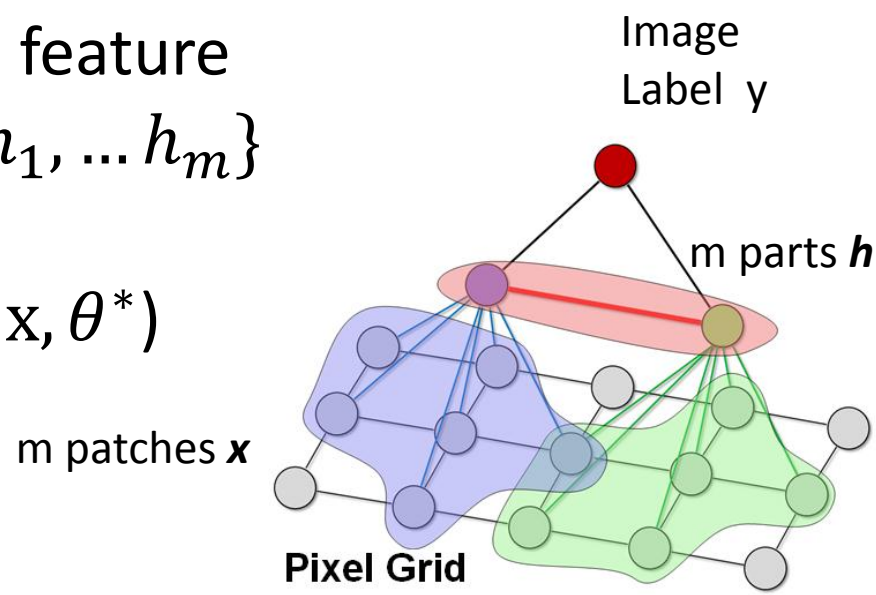
- Define the conditional distribution $P(y \mid x, \theta)$
- Parameter Learning

$$\theta^* = arg\, max_\theta \sum_i \log P(y_i \mid x_i, \theta)$$

Take the label $y^* = \arg max_y\, P(y \mid x, \theta^*)$

Object Recognition

- *Possible Image Label Set, i.e. Y* = {background*;* car*}*
- Given an image, define variable: label *y*∈ Y, "patches" $x = \{x_1, \dots x_m\}$
- Training set : n labeled images (x_i, y_i) where each $y_i \in Y$, and each $x_i = \{x_{i,1}, \dots x_{i,m}\}$
- $\emptyset(x_j) \in R^d$: representation feature
- Hidden variable: parts h= $\{h_1, \dots h_m\}$
- θ : parameter of the model
- Take the label $\mathrm{argmax_y}\ \mathrm{p(y|x}, \theta^*)$



Object Recognition

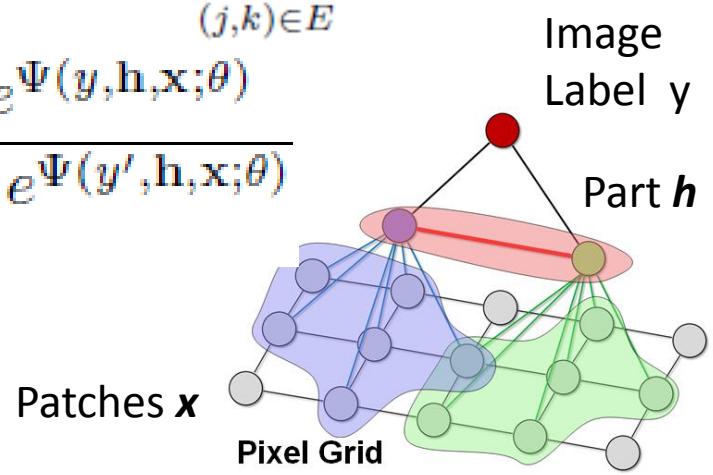
- We define

$$P(y, \mathbf{h} \mid \mathbf{x}, \theta) = \frac{e^{\Psi(y,\mathbf{h},\mathbf{x};\theta)}}{\sum_{y',\mathbf{h}} e^{\Psi(y',\mathbf{h},\mathbf{x};\theta)}}$$

where the potential function is

$$\Psi(y, \mathbf{h}, \mathbf{x}; \theta) = \sum_j \phi(x_j) \cdot \theta(h_j) + \sum_j \theta(y, h_j) + \sum_{(j,k)\in E} \theta(y, h_j, h_k)$$

$$\boxed{P(y \mid \mathbf{x}, \theta) = \sum_{\mathbf{h}} P(y, \mathbf{h} \mid \mathbf{x}, \theta)} = \frac{\sum_{\mathbf{h}} e^{\Psi(y,\mathbf{h},\mathbf{x};\theta)}}{\sum_{y',\mathbf{h}} e^{\Psi(y',\mathbf{h},\mathbf{x};\theta)}}$$



Object Recognition

- The object function in training the parameter:

$$L(\theta) = \sum_i \log P(y_i \mid \mathbf{x}_i, \theta) - \frac{1}{2\sigma^2}||\theta||^2$$

where the first term is likelihood term, and the second term is the log of a Gaussian prior with variance σ^2

$$\theta^* = \operatorname{argmax}_\theta \quad L(\theta)$$

- Optimization: gradient ascent

Using BP

- $\text{argmax}_{\text{y}}\ \text{p}(\text{y}|\text{x}, \theta^*)$

$$\forall y \in \mathcal{Y}, \quad Z(y \mid \mathbf{x}, \theta) = \sum_{\mathbf{h}} \exp\{\Psi(y, \mathbf{h}, \mathbf{x}; \theta)\}$$

- $\dfrac{\partial L(\theta)}{\partial \theta}$

$$\forall y \in \mathcal{Y}, j \in 1 \ldots m, a \in \mathcal{H}, \quad P(h_j = a \mid y, \mathbf{x}, \theta) = \sum_{\mathbf{h}: h_j = a} P(\mathbf{h} \mid y, \mathbf{x}, \theta)$$

$$\forall y \in \mathcal{Y}, (j, k) \in E, a, b \in \mathcal{H}, \quad P(h_j = a, h_k = b \mid y, \mathbf{x}, \theta) = \sum_{\mathbf{h}: h_j = a, h_k = b} P(\mathbf{h} \mid y, \mathbf{x}, \theta)$$

Results

Data set	5 parts	10 parts
Car Side	94 %	99 %
Car Rear	91 %	91.7 %

Summary

- MRF
- CRF
- Graph Cut Algorithm
  - Binary
  - Multi-label
- Belief Propagation Algorithm

- Modeling Vision Problem is hard because
  - Optimization algorithm doesn’t help you to find global maxima
  - Model itself

Thank you!