Denmark'08
Tutorial

Markov Random Fields for Vision and Graphics

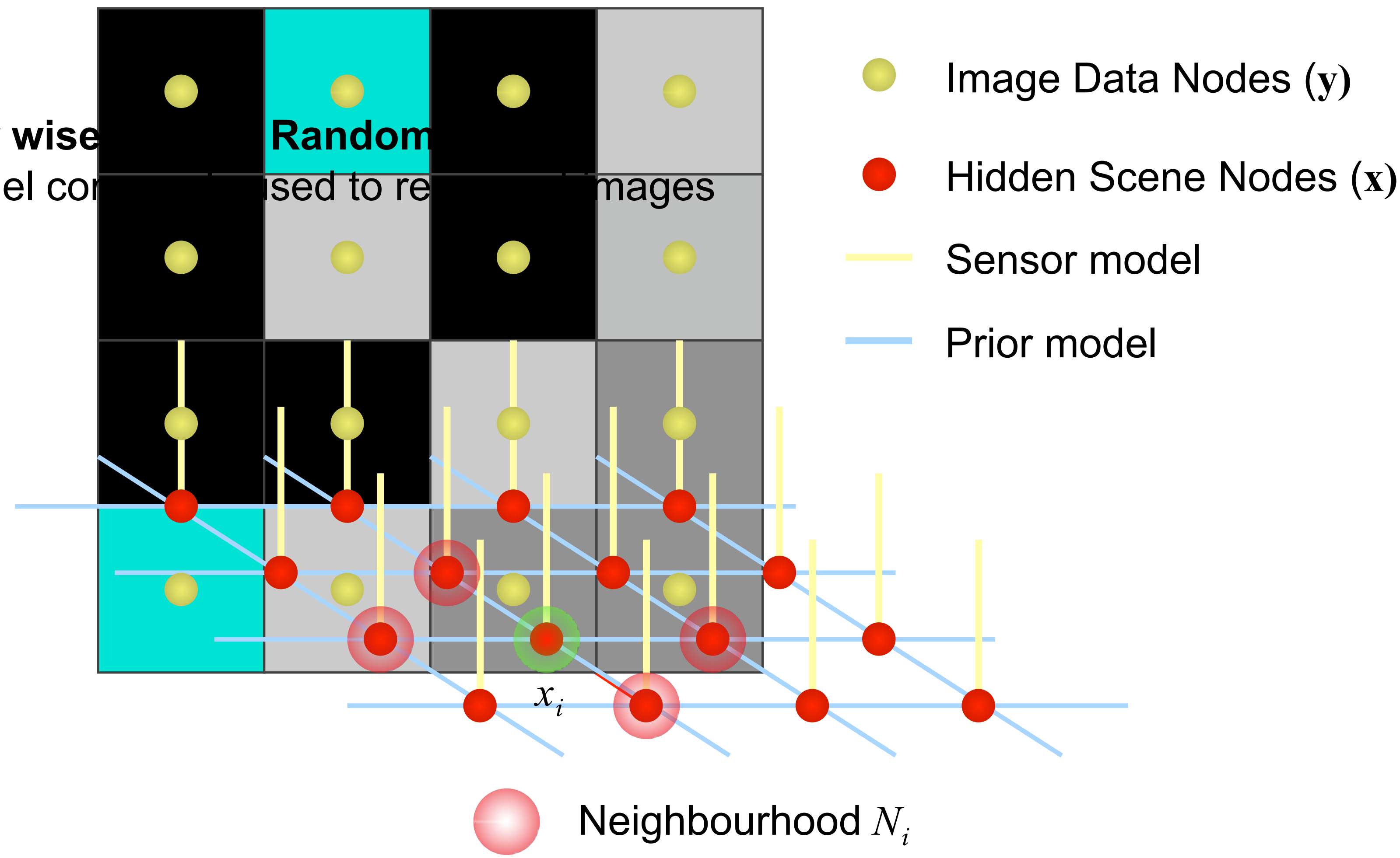
Phil Torr,

Oxford Brookes University, http://cms.brookes.ac.uk/research/visiongroup/

thanks to

Manish Jethwa, Pushmeet Kohli, Pawan Kumar, Yuri Boykov, George Vogliatzsis, Chria Bishop, Bill Freeman and many others.

Markov Random Fields



$$p(x_i \mid \mathbf{x}) = p(x_i \mid \{x_k\}), \quad x_k \in N_i$$

Slide Stealing!!!!

- Many many thanks to Yuri Boykov for allowing me to use his slides

“Normal” MRF

- This is a pairwise MRF
- First order according to Besag
- Second order pseudo Boolean function
- Higher order cliques???

Part I Shortest Path:

Dynamic Programming

Applications

Shortest paths on graphs (examples)

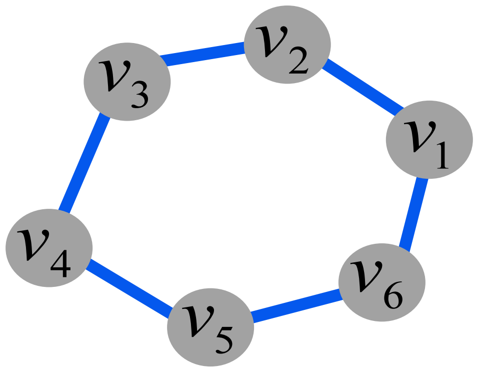
- Object extraction (Segmentation)
  - live-wire [*Falcao, Udupa, Samarasekera, Sharma* 1998]
  - intelligent scissors [*Mortensen, Barrett* 1998]
- Scan line stereo (& Optic Flow)
  - Ohta & Kanade, 1985
  - Cox, Hingorani, Rao, 1996
- Texture Synthesis
  - Efros & Freeman, 2001

Dynamic Programming (examples)

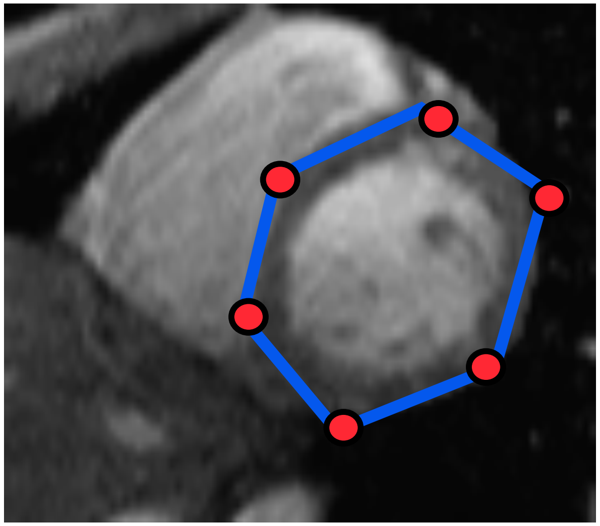
- Snakes
  - Amini, Weymouth, Jain, 1990
- Scan-line stereo
  - e.g. Ohta&Kanade'85, Cox at.al.'96
- Object Matching / Registrations
  - Pictorial structures [Felzenszwalb & Huttenlocher, 2000]

Discrete Snakes

[Amini, Weymouth, Jain, 1990]



control points



First-order interactions

$$E(v_1, v_2, \ldots, v_n) = E_1(v_1, v_2) + E_2(v_2, v_3) + \ldots + E_{n-1}(v_{n-1}, v_n)$$

Second-order interactions

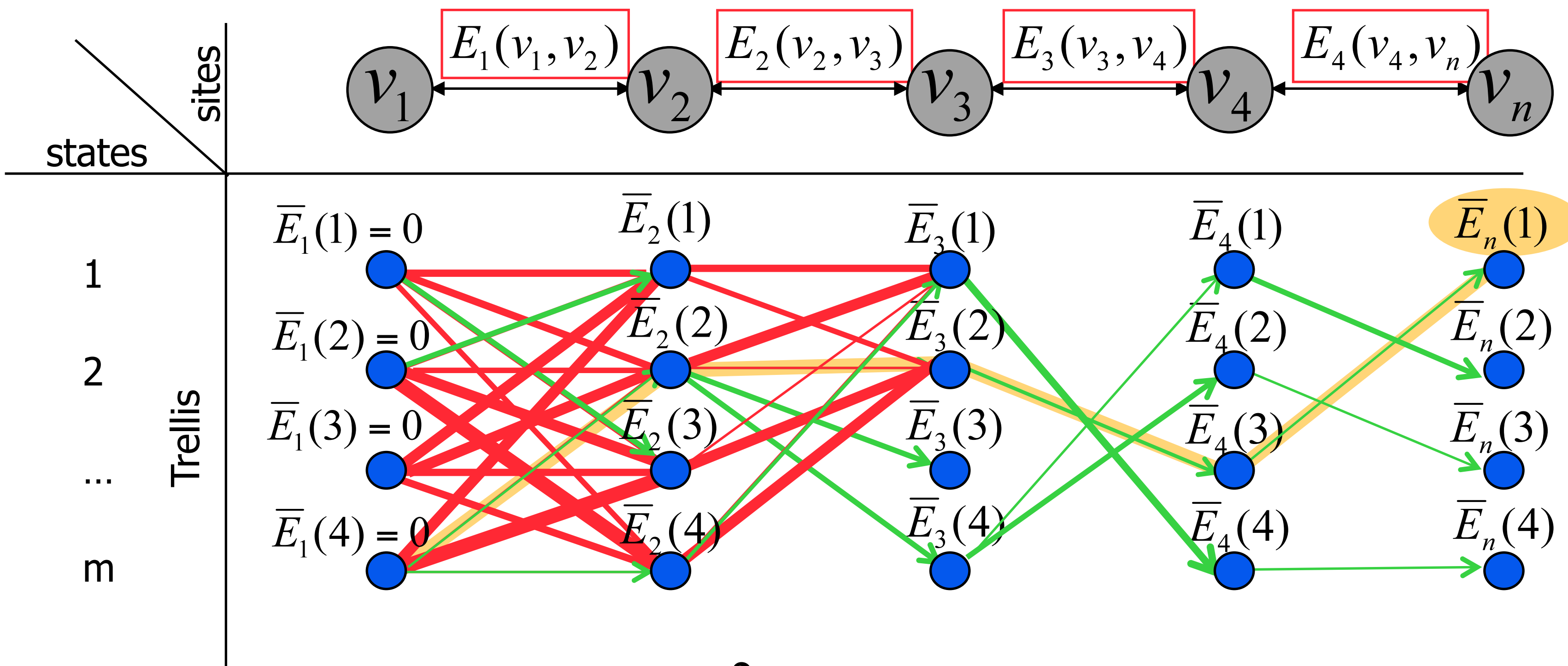
$$+ E_1(v_1, v_2, v_3) + E_2(v_2, v_3, v_4) + \ldots + E_{n-2}(v_{n-2}, v_{n-1}, v_n)$$

Energy E is minimized via Dynamic Programming

Dynamic Programming (DP)

In this talk we mainly concentrate on **first-order interactions**

$$E_1(v_1, v_2) + E_2(v_2, v_3) + \ldots + E_{n-1}(v_{n-1}, v_n)$$



Complexity: $O(nm^2)$, Worst case = Best Case

Note: Similar to Viterbi Algorithm

- Viterbi, developed in speech to solve HMM's

Higher Order Interaction

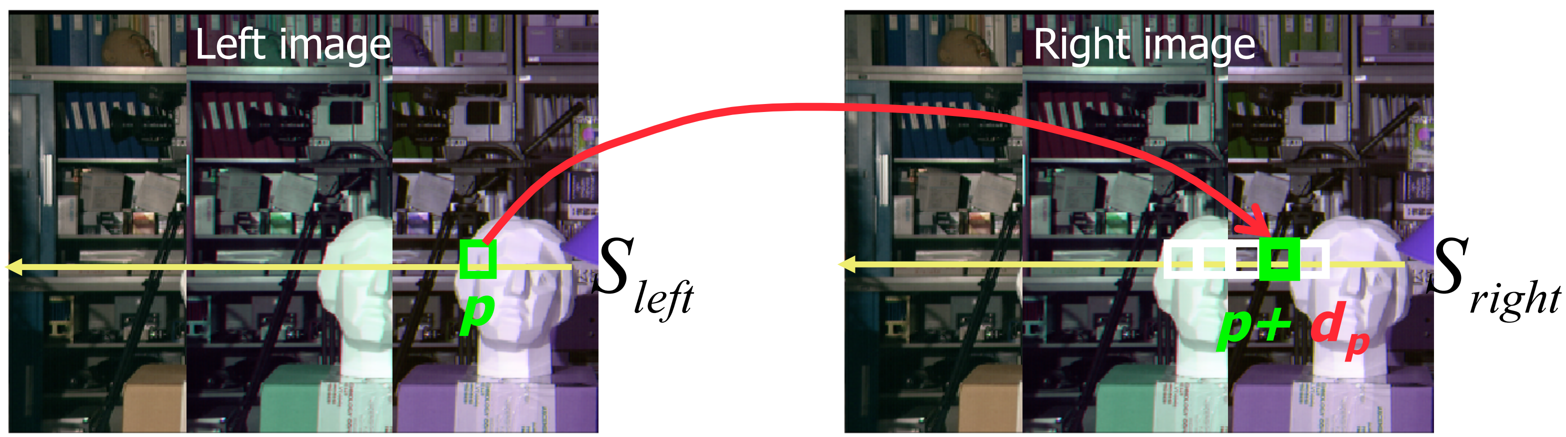
- Second order can be done by considering trellises where states corresponding to the joint labelling of two points.
- Problem m^2 labels and complexity in square of labels.
- Similar idea for higher order interactions…

N.B. 2nd & 3rd order PSB fcns!

DP in vision: Scan-line stereo

- Baker & Binford 1981
- Ohta & Kanade, 1985
- Geiger at.al. 1992
- Belhumeur & Mumford 1992
- Cox at.al. 1996
- Scharstein & Szelisky 2001

Example:



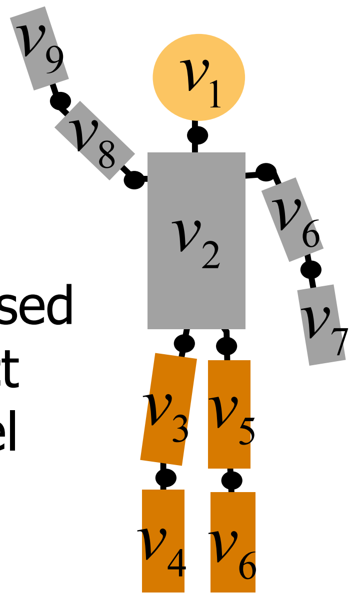
$$E(d_1, d_2, \dots, d_n) = \sum_{p \in S_{left}} E_p(d_p, d_{p+1})$$

Disparities of pixels in the scan line S_{left}

DP in vision: Object matching / registration

Pictorial structures [Felzenszwalb & Huttenlocher, 2000]

Tree-based object model



DP can be applied to trees!

FH’00 beat O(nm^2) complexity of DP for a class of first-order energies

Hierarchical Part-Based Human Body Pose Estimation
BMVC 2005

* Ramanan Navaratnam

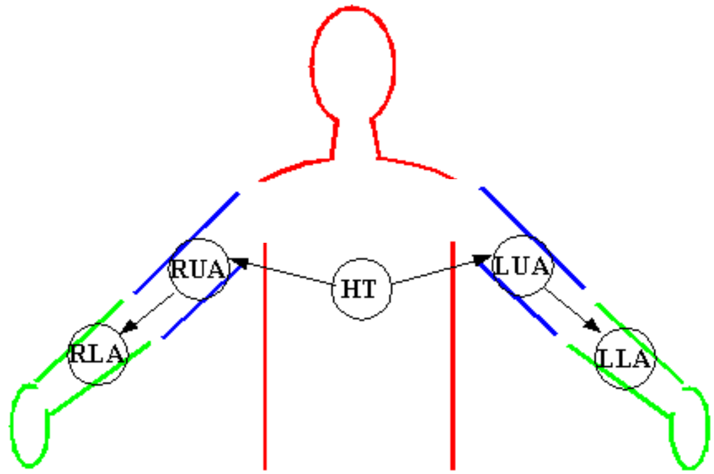
* Arasanathan Thayananthan

† Prof. Phil Torr

* Prof. Roberto Cipolla

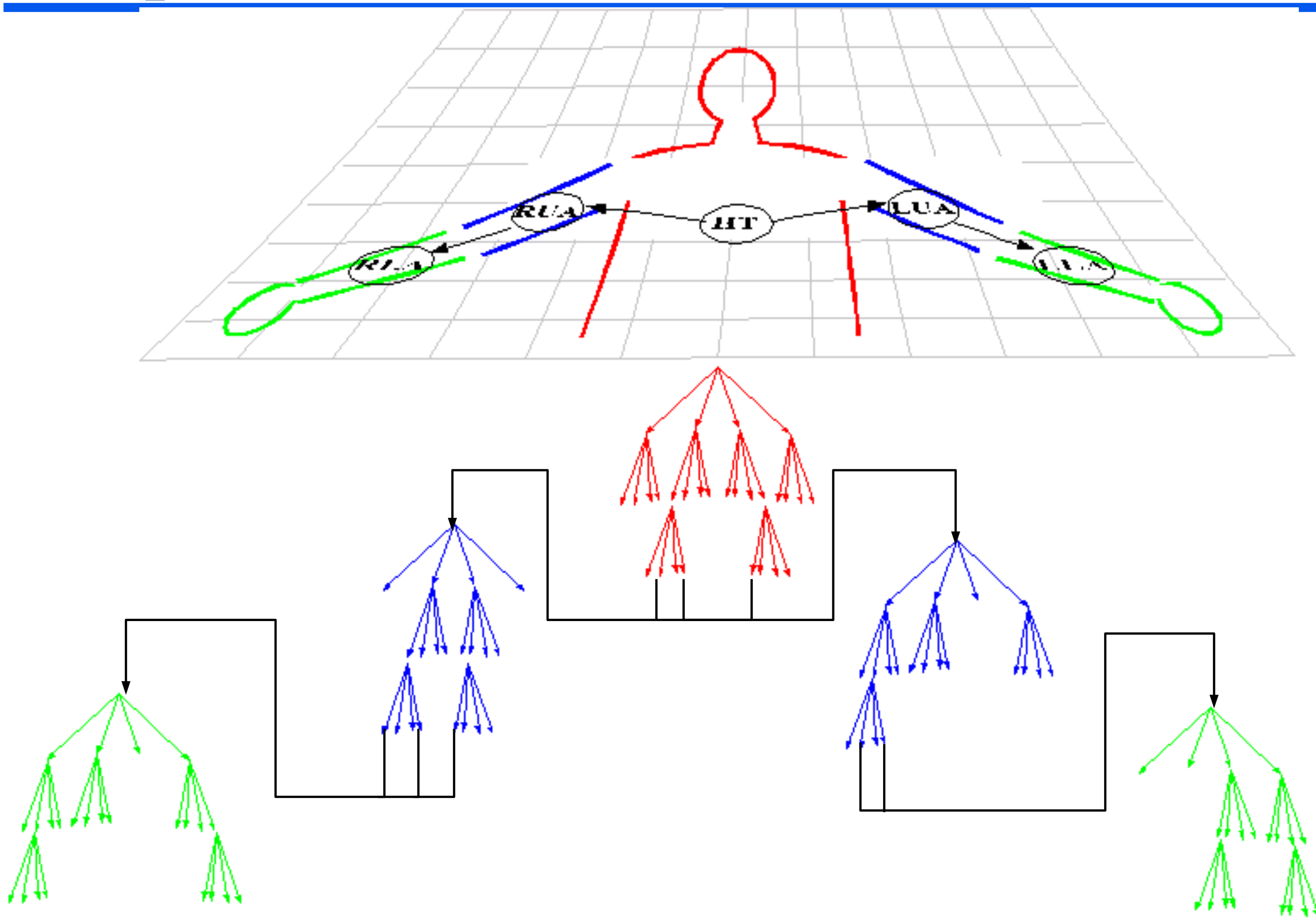
* University Of Cambridge † Oxford Brookes University

Hierarchical Parts

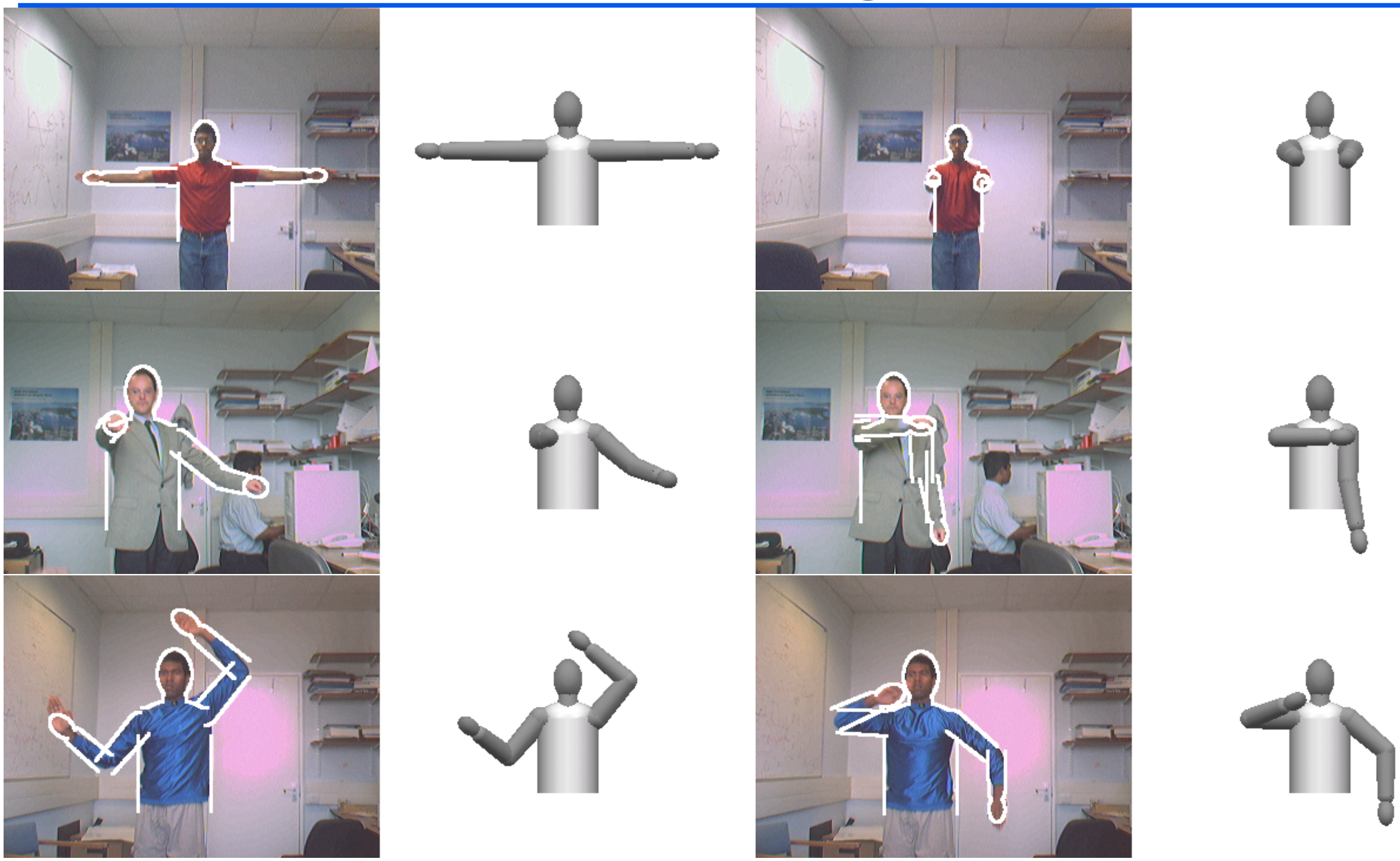


$$p(\mathbf{x}|\mathbf{z}) = \frac{p(\mathbf{x})p(\mathbf{z}|\{x_i\})}{p(\mathbf{z})}$$
$$= \frac{p(x_{HT})p(x_{RUA}|x_{HT})p(x_{LUA}|x_{HT})p(x_{RLA}|x_{RUA})p(x_{LLA}|x_{LUA})p(\mathbf{z}|\{x_i\})}{p(\mathbf{z})}$$

Template Search

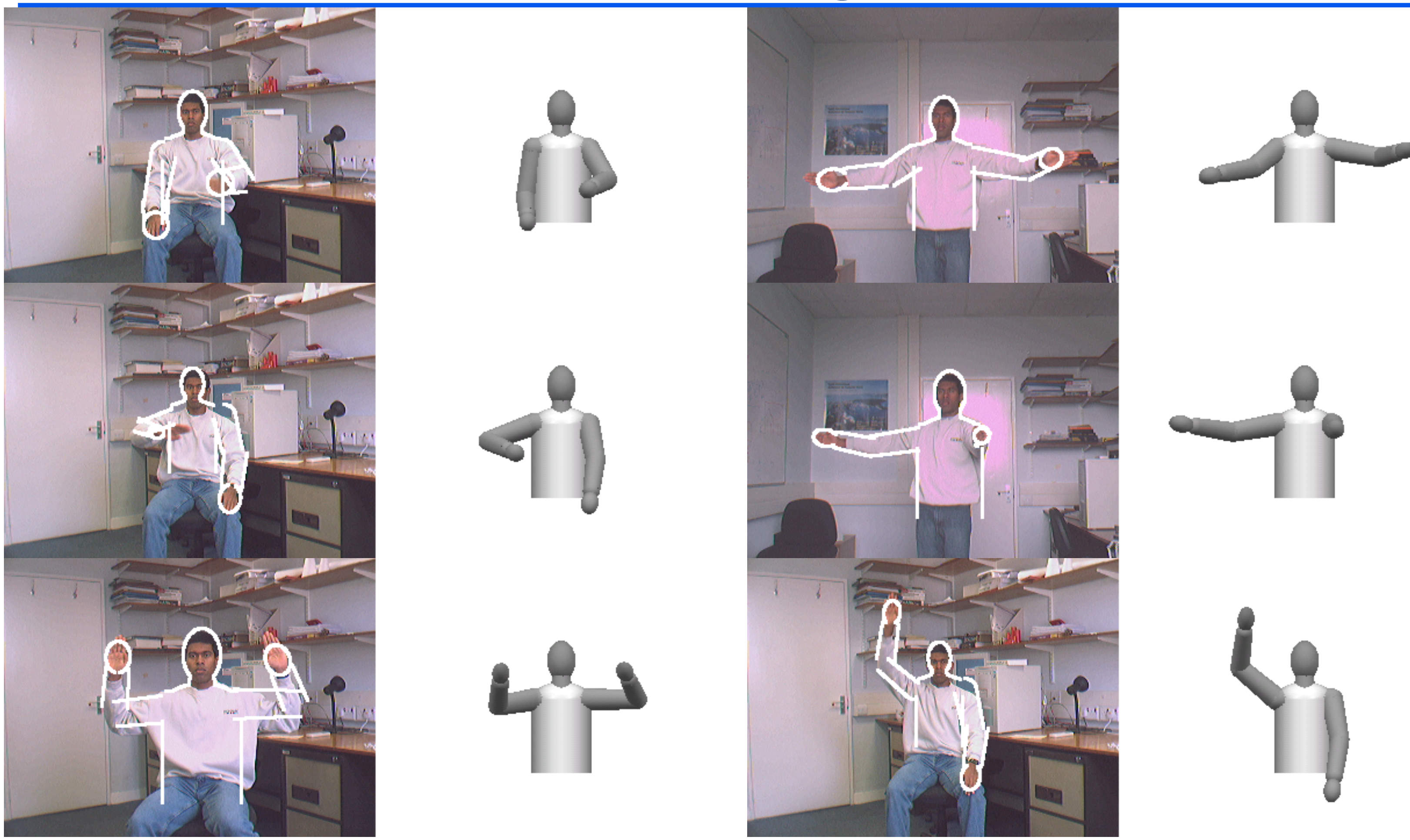


Pose Estimation in a Single Frame



OXFORD BROOKES UNIVERSITY

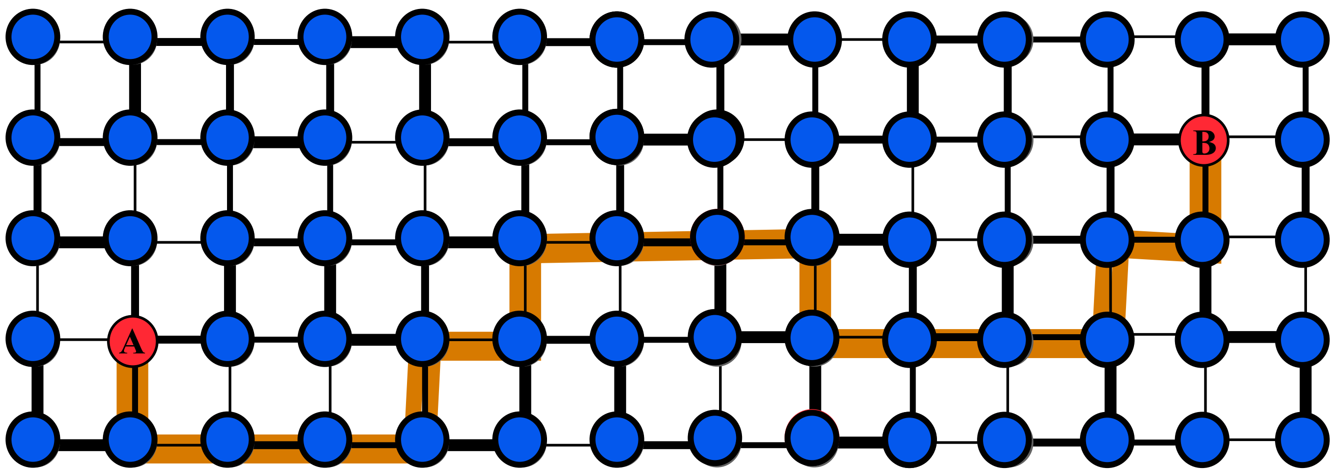
Pose Estimation in a Single Frame



DP => graph algorithms

DP can be "implemented" via
Shortest Path algorithm for graphs

Shortest Paths on weighted graphs



- processed nodes (distance to A is known)
- active nodes (front)
- active node with the smallest distance value

Dijkstra algorithm

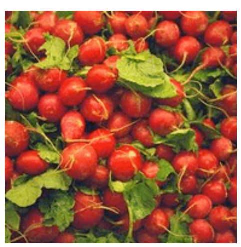
Shortest paths on graphs (examples)

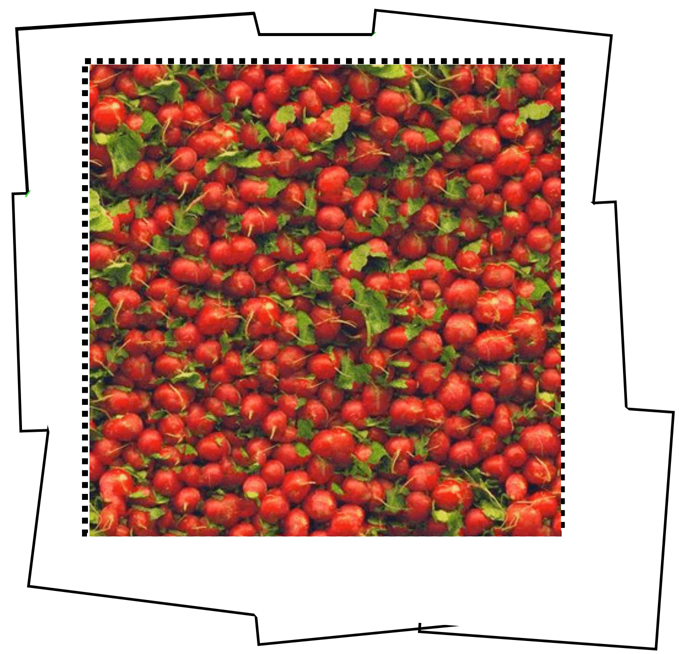
- Texture Synthesis
  - Image quilting [Efros & Freeman, 2001]
- Object extraction
  - live-wire [Falcao, Udupa, Samarasekera, Sharma 1998]
  - intelligent scissors [Mortensen, Barrett 1998]
- Scan line stereo
  - Ohta & Kanade, 1985
  - Cox, Hingorani, Rao, 1996

Shortest paths: Texture synthesis

"Image quilting"

Efros & Freeman, 2001

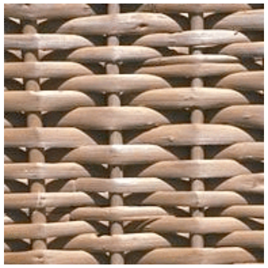


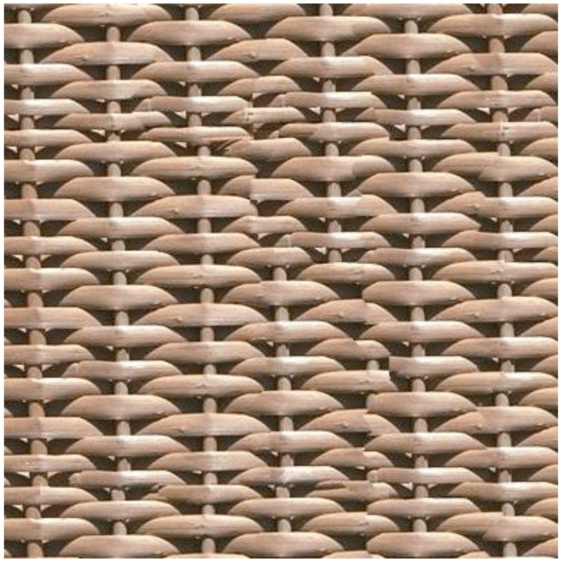


Shortest paths: Texture synthesis

"Image quilting"

Efros & Freeman, 2001

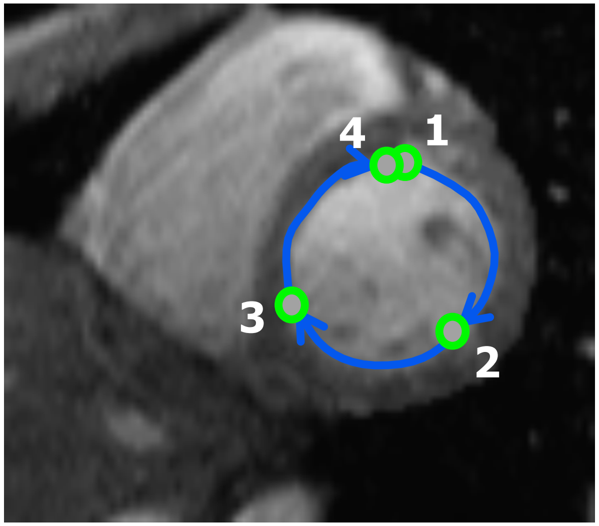




Shortest paths: object extraction

- live-wire [Falcao, Udupa, Samarasekera, Sharma 1998]
- intelligent scissors [Mortensen, Barrett 1998]

Example:

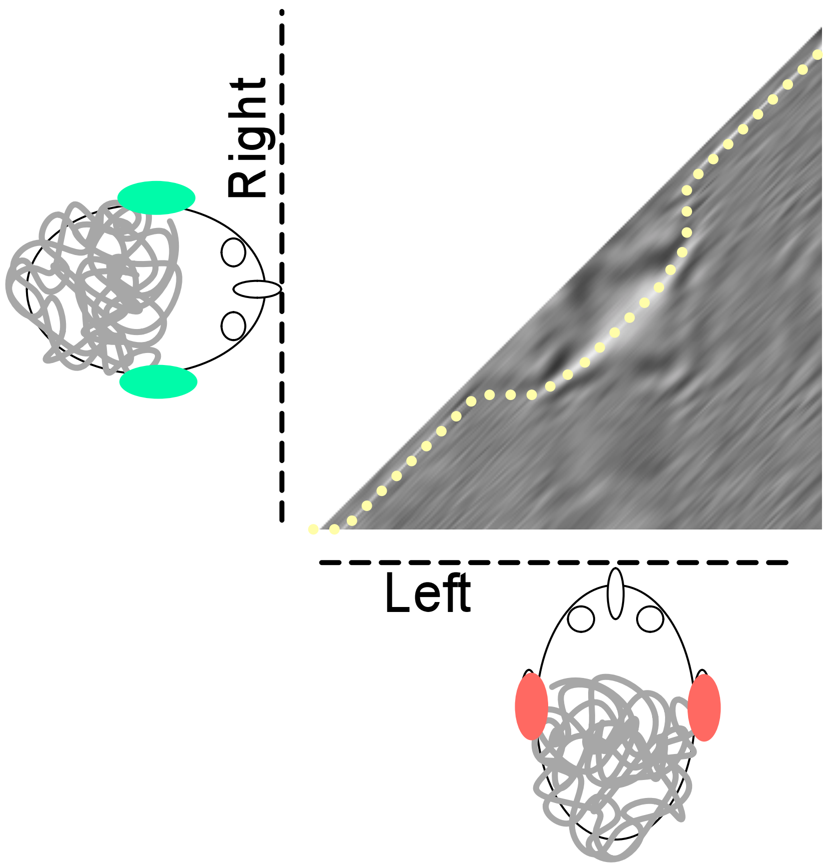


Graph edges are "cheap" in places with high intensity gradients

Matching space
(one scanline pair)

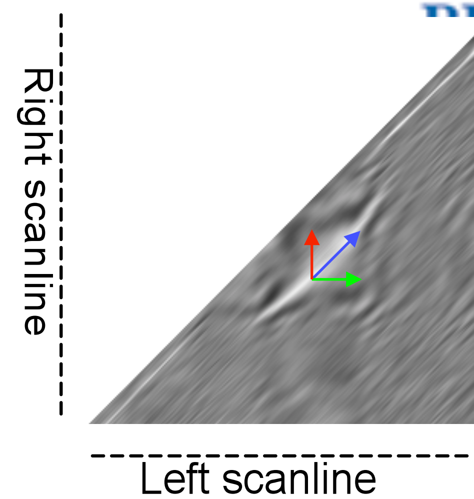
- compare every pixel in the left scanline with every pixel in right scanline (subject to **r≤l**)
  - use a similarity metric, $\boldsymbol{M_n(l, r)}$, for comparisons
    - *e.g.* sum of squared differences (SSD) over 3x3 windows centred on the pixels in question
    - normalise s.t. $\mathbf{0 \le M_n \le 1}$

- High similarity ⇒ low cost
  (light points)
- Low similarity ⇒ high cost
  (dark points)

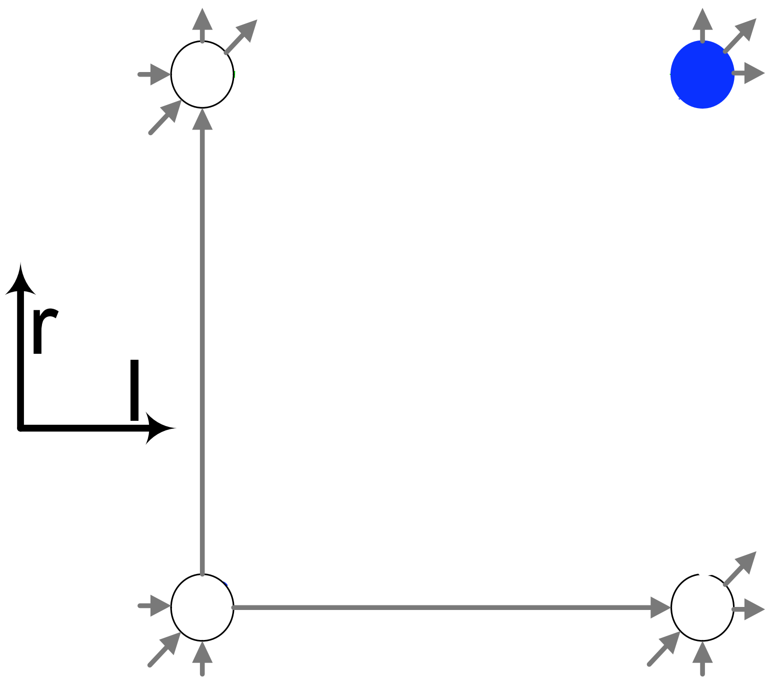


State of the art: *'Three-move' graph*

- Path planning on graph



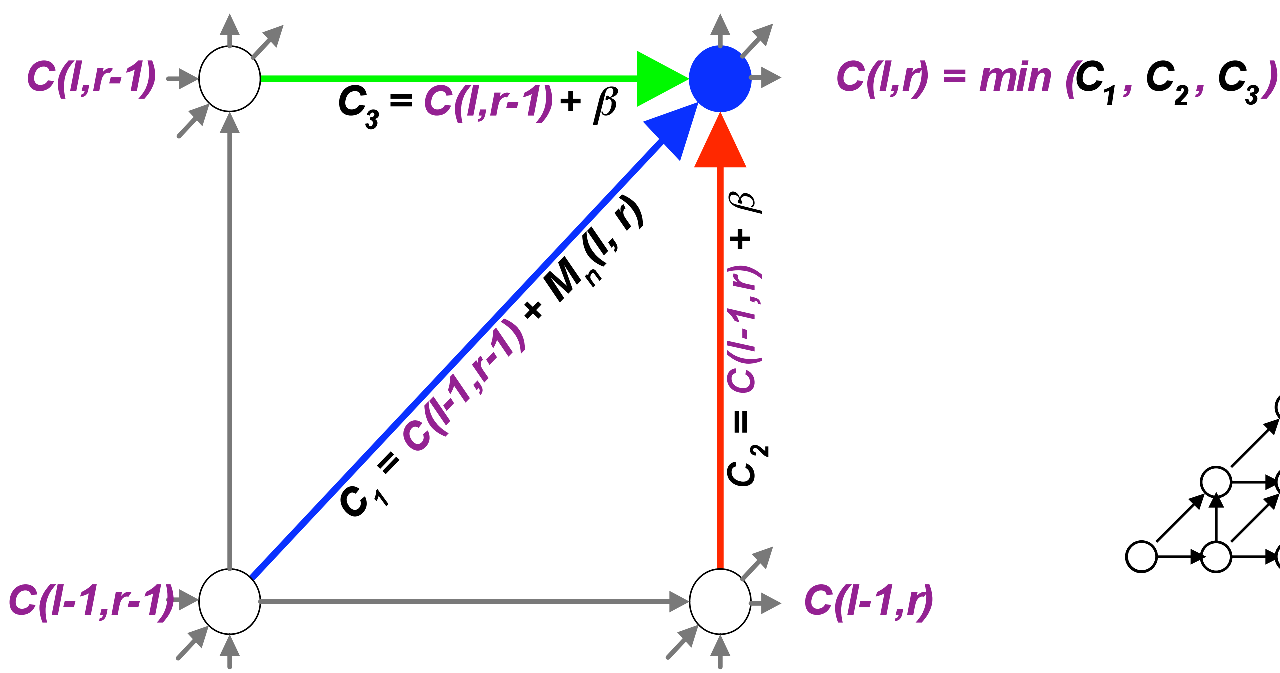
- Three edges into each node (l, r):
  - **Matched** from (l-1, r-1)
    - weight $\boldsymbol{M_n(l, r)}$
  - **Left Occlusion** from (l, r-1)
    - weight β ('occlusion cost')
  - **Right Occlusion** from (l-1, r)
    - weight β ('occlusion cost')

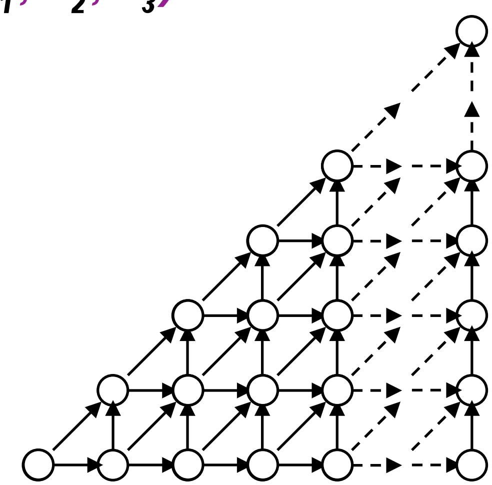


Three-move model

Minimum cost path:
Dynamic programming step

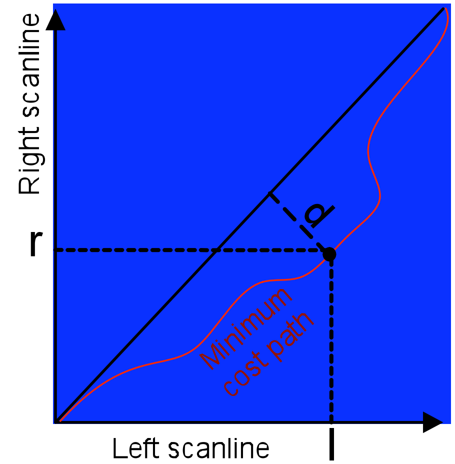
- Build up minimum cost graph in ordered, greedy fashion from bottom left to top right
- Maintain backward links table





Minimum cost path

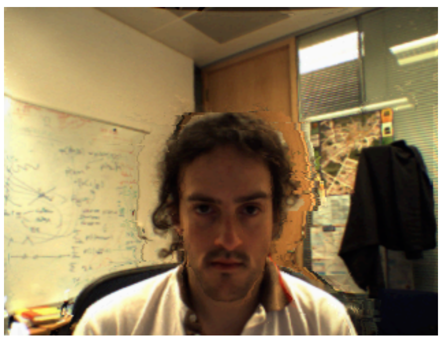
- Following backward links gives minimum cost path through graph



- Over all scanlines these paths form a depth map

Disparity (depth) map

- Create synthetic new view ***(Criminsi, Torr ICCV 2003)***



Synthetic new view

DP ~ *Shortest Paths*

Snakes (via DP)	Live wire / Int. scissors
Scan-line stereo (via DP)	Scan-line stereo (via Shortest paths)

- DP and *Shortest paths* are comparable optimization tools

  -1D structures only

  -first- and second-order interactions: (DP~ $O(nm^2), O(nm^3)$)

  -similar theoretical complexities

Note about practical complexities:

(DP) "Average Case" = "Worst Case"

(SP) "Average Case" < "Worst Case"

DP / Shortest-paths

- Efficient global optimization tools ☺
- Good for 1-D optimization problems only ☹
  - Can’t deal with dependent scan-lines in stereo
  - Can’t handle bubbles (N-D snakes) or object boundaries in volumetric data

- Graph-Cuts can be seen as a “generalization” to N-D

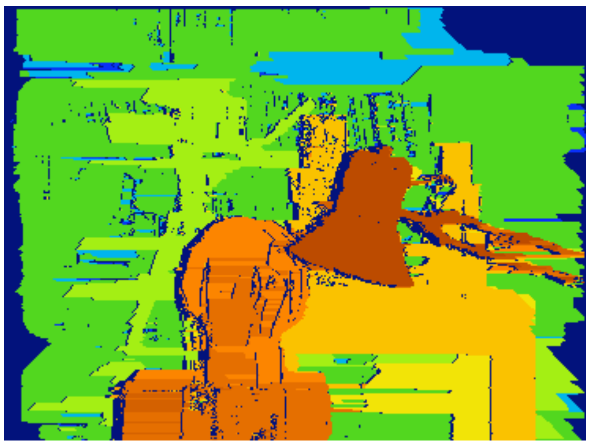
Summary Part I

- Dynamic Programming
- Shortest path algorithm
- Examples
  - Snakes
  - Textures
  - Stereo

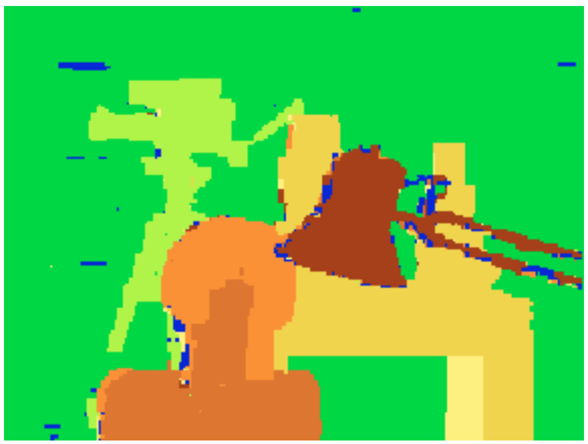
Part II Graph Cuts

Graph Cuts vs Shortest Path
Augmenting Path Algorithm
Given an energy how to create the graph to be cut.
Multi Way Cuts
Examples

Stereo example



Independent
scan-lines
(via DP)



Multi-scan line
(via Graph Cuts)

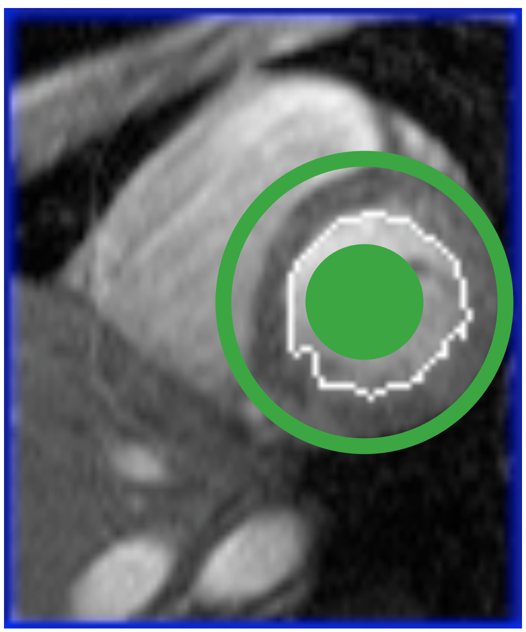
Ground truth

Graph Cuts for Solving MRF's

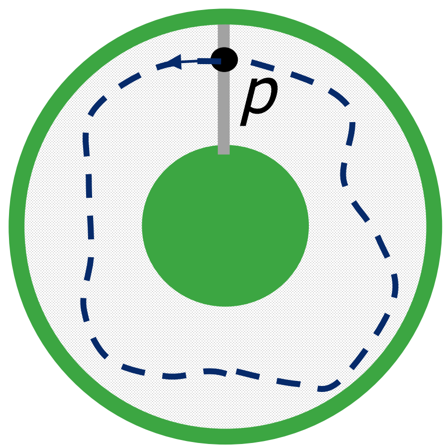
- segmentation, object extraction, stereo, motion, image restoration, pattern recognition, shape reconstruction, object matching/recognition, augmented reality, texture synthesis, …

1D Graph cuts = shortest paths

Example:
find the shortest closed contour in a given domain of a graph

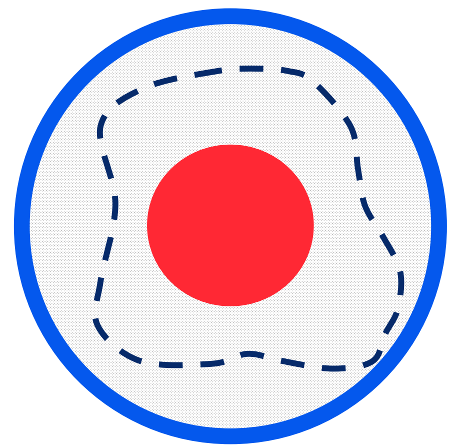


Shortest paths approach



Compute the *shortest path* *p* -> *p* for a point *p*. Repeat for all points on the gray line. Then choose the optimal contour.

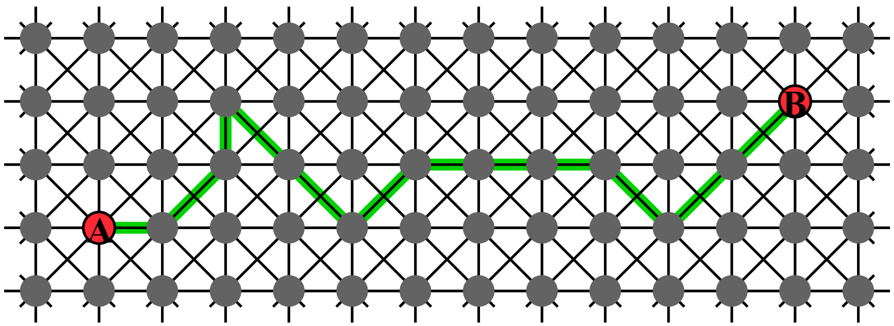
Graph Cuts approach



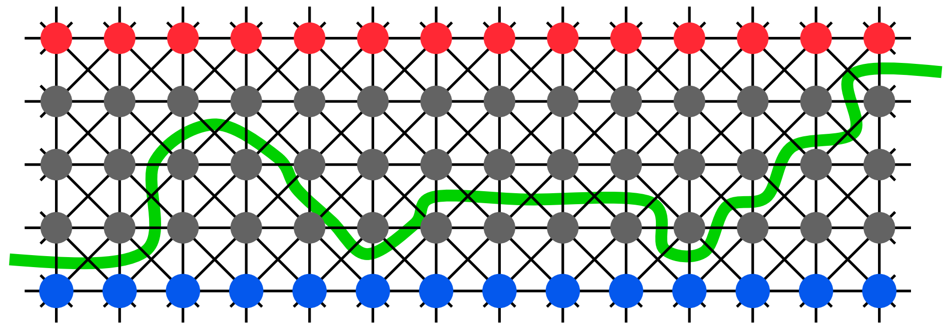
Compute the *minimum cut* that separates red region from blue region

Graph cuts vs. *shortest paths*

- On 2D grids *graph cuts* and *shortest paths* give optimal 1D contours.



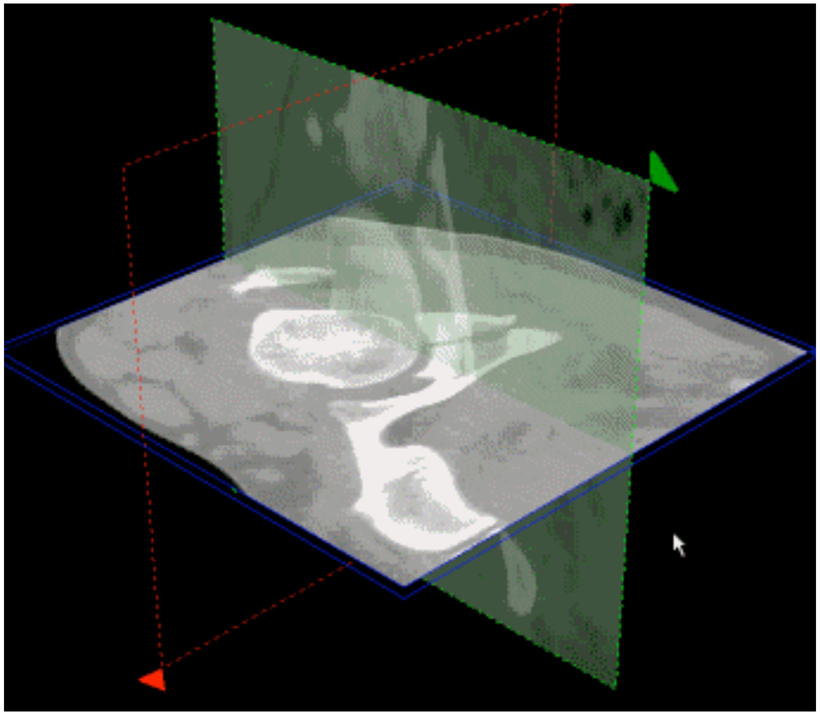
A Path connects points



A Cut separates regions

- *Shortest paths* still give optimal 1-D **contours** on N-D grids
- *Min-cuts* give optimal **hyper-surfaces** on N-D grids

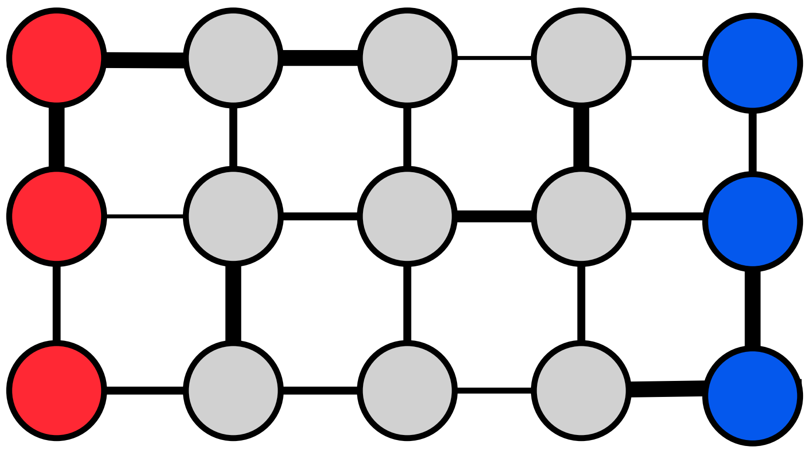
Graph Cuts as hyper-surface in 3D



Object extraction [Boykov, Jolly, Funkalea 2001, 2004]

Graph Cuts Basics
(simple 2D example)

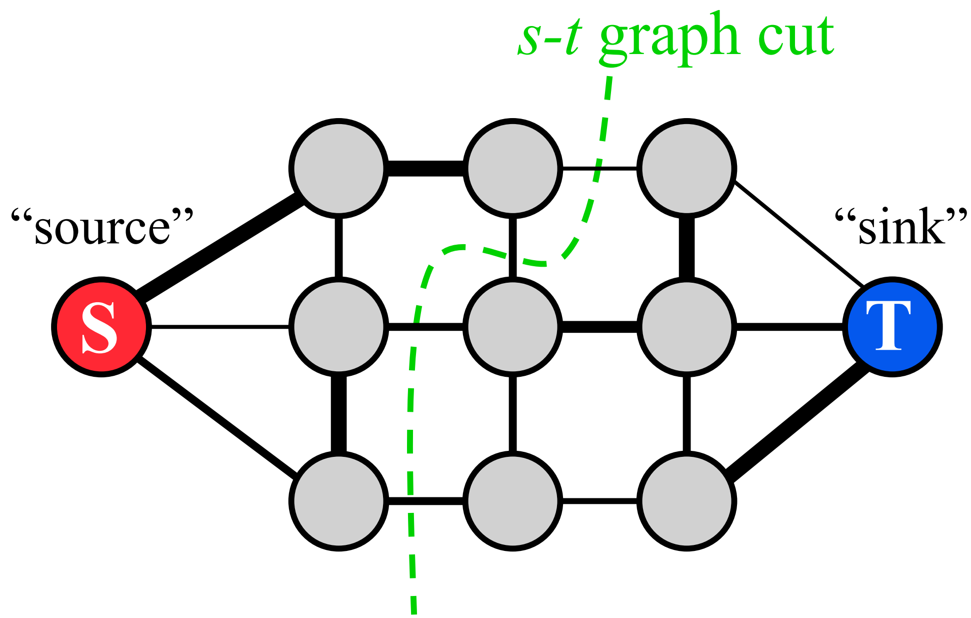
Goal: divide the graph into two parts separating red and blue nodes



Red/blue nodes can be "identified" into two super nodes (terminals)

Graph Cuts Basics
(simple 2D example)

Goal: divide the graph into two parts separating red and blue nodes



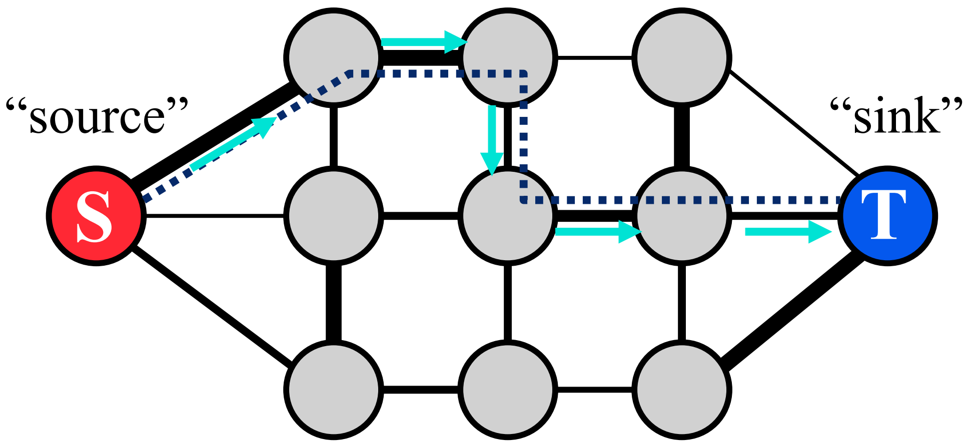
A graph with two terminals *S* and *T*

- Cut cost is a sum of severed edge weights
- **Minimum cost *s-t* cut can be found in polynomial time**

Minimum *s-t* cuts algorithms

- Augmenting paths [Ford & Fulkerson, 1962]
- Push-relabel [Goldberg-Tarjan, 1986]

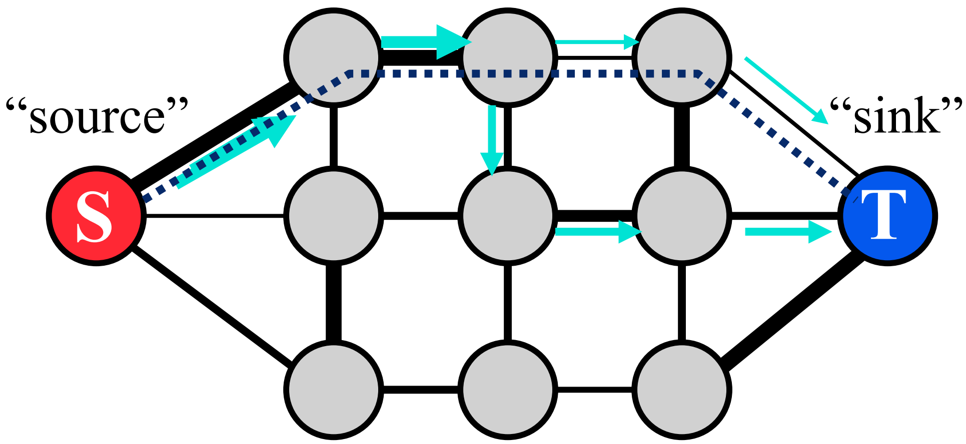
"Augmenting Paths"



A graph with two terminals

- Find a path from S to T along non-saturated edges
- Increase flow along this path until some edge saturates

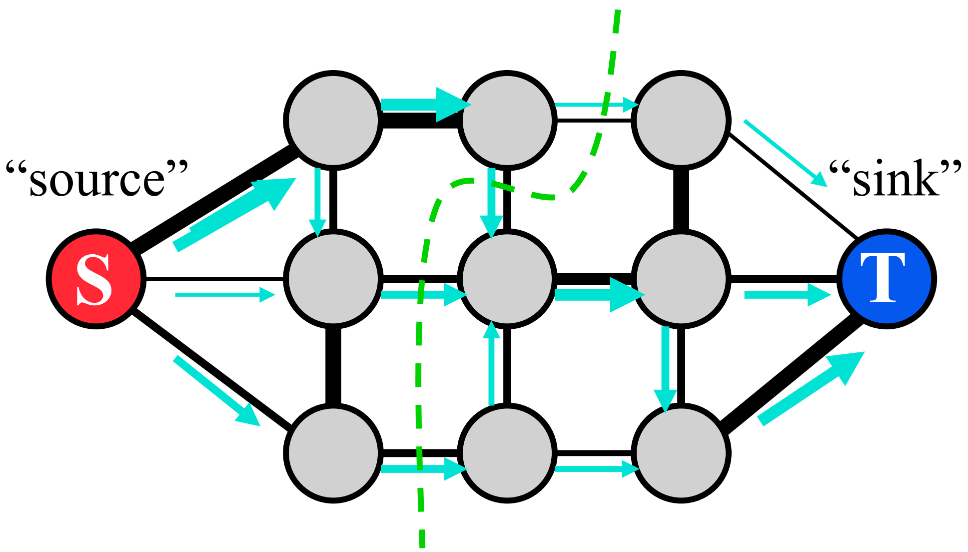
“Augmenting Paths”



A graph with two terminals

- Find a path from S to T along non-saturated edges
- Increase flow along this path until some edge saturates
- Find next path…
- Increase flow…

“Augmenting Paths”



A graph with two terminals

MAX FLOW ⇔ MIN CUT

- Find a path from S to T along non-saturated edges
- Increase flow along this path until some edge saturates

Iterate until … all paths from S to T have at least one saturated edge

Implementation notes (sequential version)

- Boykov & Kolmogorov, EMMCVPR 2001
  PAMI 2004

  - empirical comparison of different versions of *augmenting paths* and *push-relabel* algorithms on grid-graphs typical in vision

  - "tuned" version of augmenting paths is proposed
    (**freely available implementation**)

  - graph cuts can be used for problems in vision in near real time

  - empirical complexity is near linear with respect to image size

Implementation notes (parallel version)

- *Push-relabel* algorithm can be implemented in parallel on all graph nodes [e.g. Goldberg 86]

- Parallel *Push-Relabel* algorithm and typical in vision grids give a perfect combination for GPU (graphics card) hardware acceleration [work in progress…]

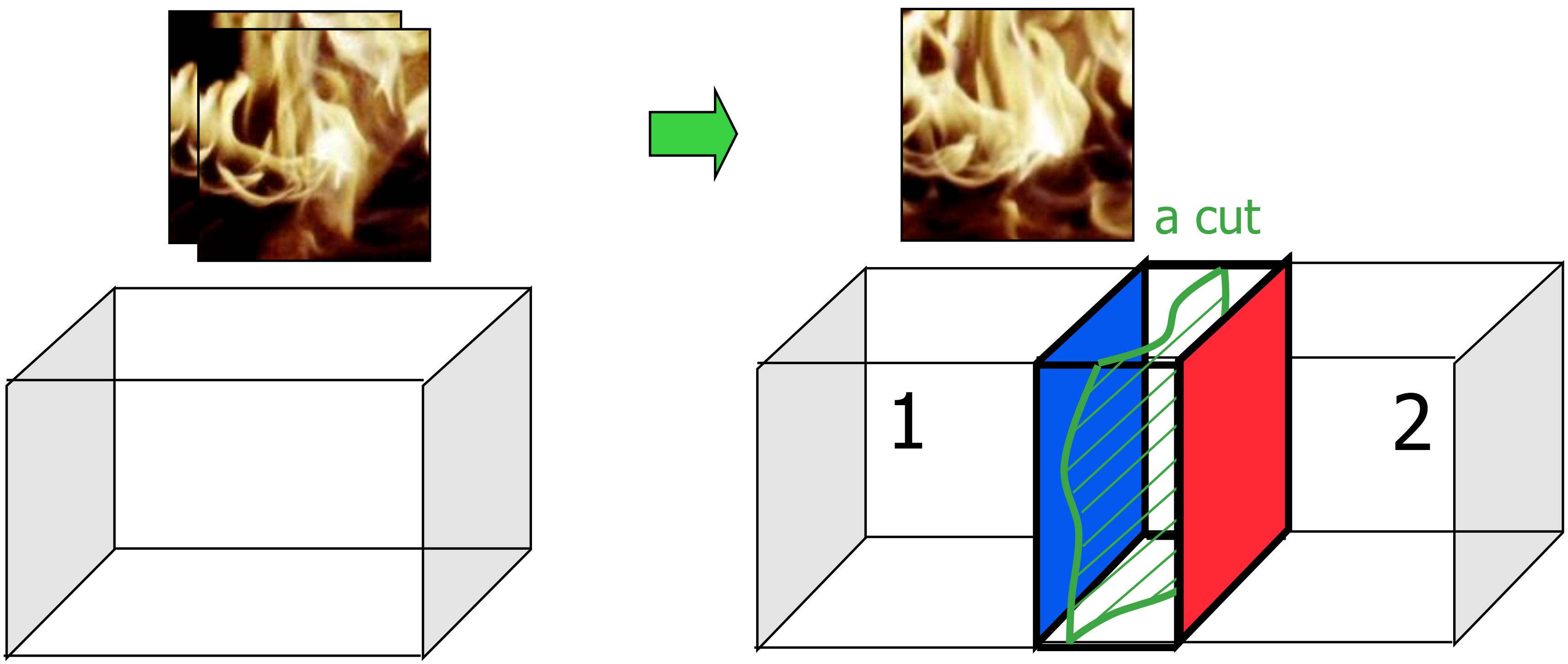
Examples of Graph-Cuts in vision

- Image Restoration (e.g. Greig at.al. 1989)
- Segmentation
  - Wu & Leahy 1993
  - Nested Cuts, Veksler 2000
- Multi-scan-line Stereo, Multi-camera stereo
  - Roy & Cox 1998, 1999
  - Ishikawa & Geiger 1998, 2003
  - Boykov, Veksler, Zabih 1998, 2001
  - Kolmogorov & Zabih 2002, 2004
- Object Matching/Recognition (Boykov & Huttenlocher 1999)
- N-D Object extraction (photo-video editing, medical imaging)
  - Boykov, Jolly, Funka-Lea 2000, 2001, 2004
  - Boykov & Kolmogorov 2003
  - Rother, Blake, Kolmogorov 2004
- Texture synthesis (Kwatra, Schodl, Essa, Bobick 2003)
- Shape reconstruction (Snow,Viola,Zabih 2000)
- Motion (e.g. Xiao, Shah 2004)

s-t graph cuts for video textures

Graph-cuts video textures

(Kwatra, Schodl, Essa, Bobick 2003)



Short video clip

Long video clip

3D generalization of "**image-quilting**" (Efros & Freeman, 2001)

Graph Cut Textures

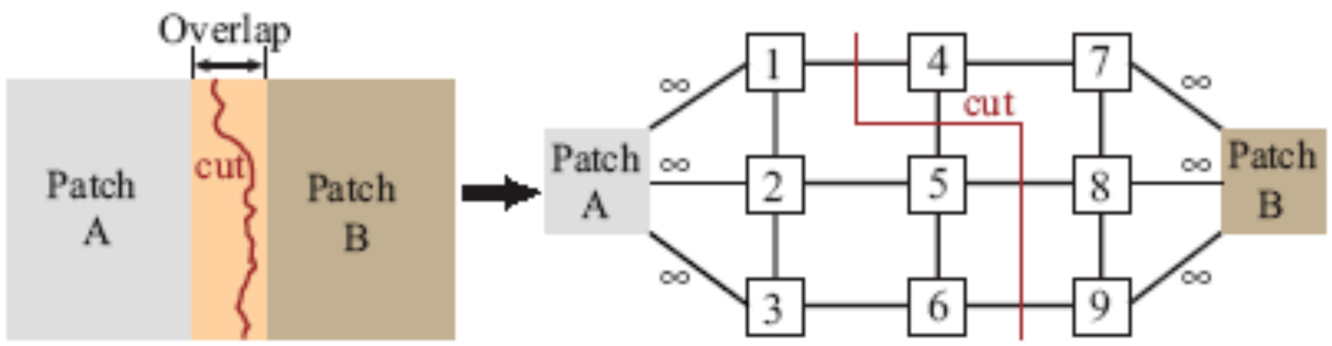


Figure 2: (Left) Schematic showing the overlapping region between two patches. (Right) Graph formulation of the seam finding problem, with the red line showing the minimum cost cut.

$$M(s,t,\mathbf{A},\mathbf{B}) = \|\mathbf{A}(s) - \mathbf{B}(s)\| + \|\mathbf{A}(t) - \mathbf{B}(t)\|$$

s-t graph cuts for video textures

Graph-cuts video textures

(Kwatra, Schodl, Essa, Bobick 2003)

original short clip

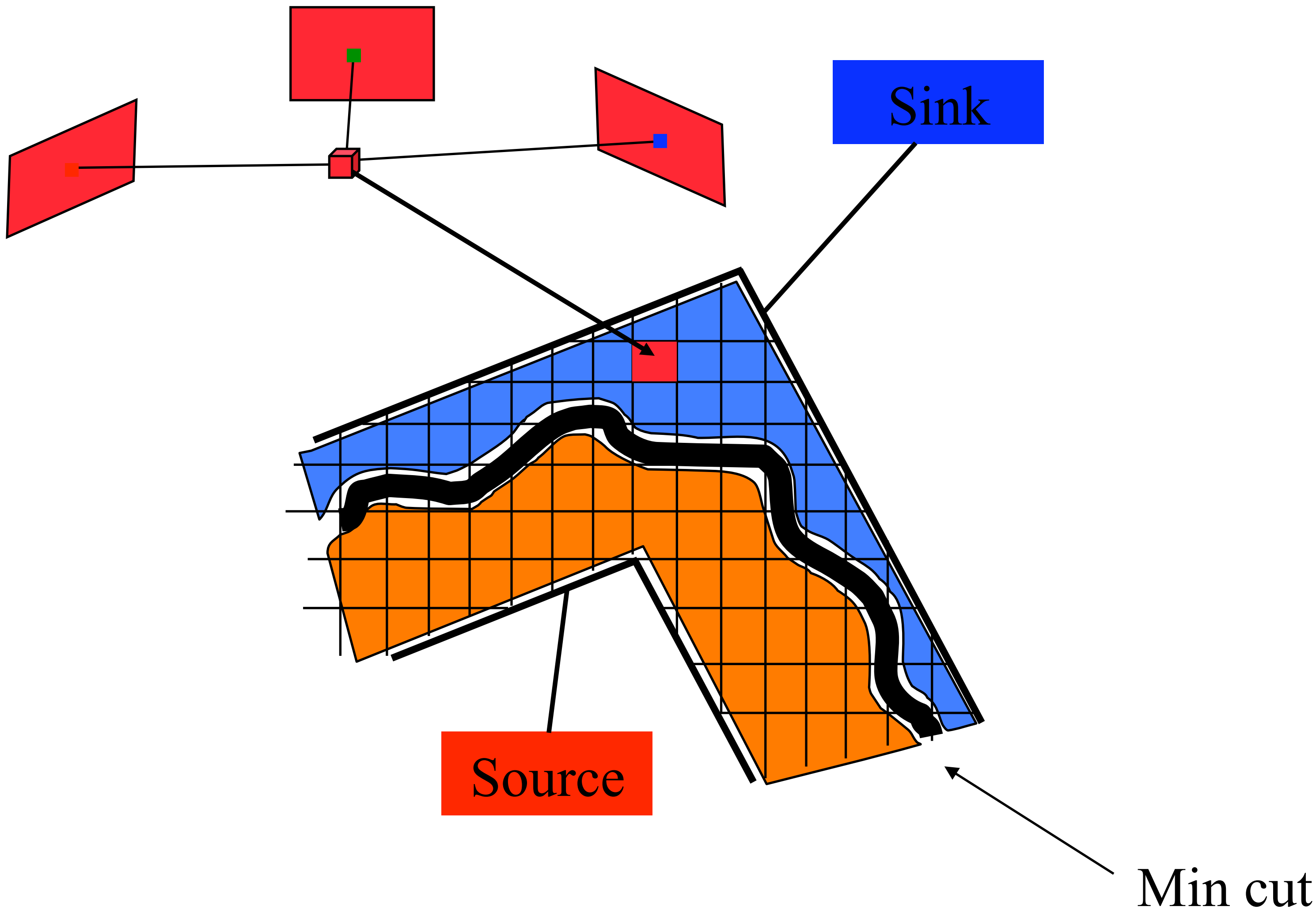
synthetic infinite texture

Multi-view Stereo via Volumetric Graph-cuts

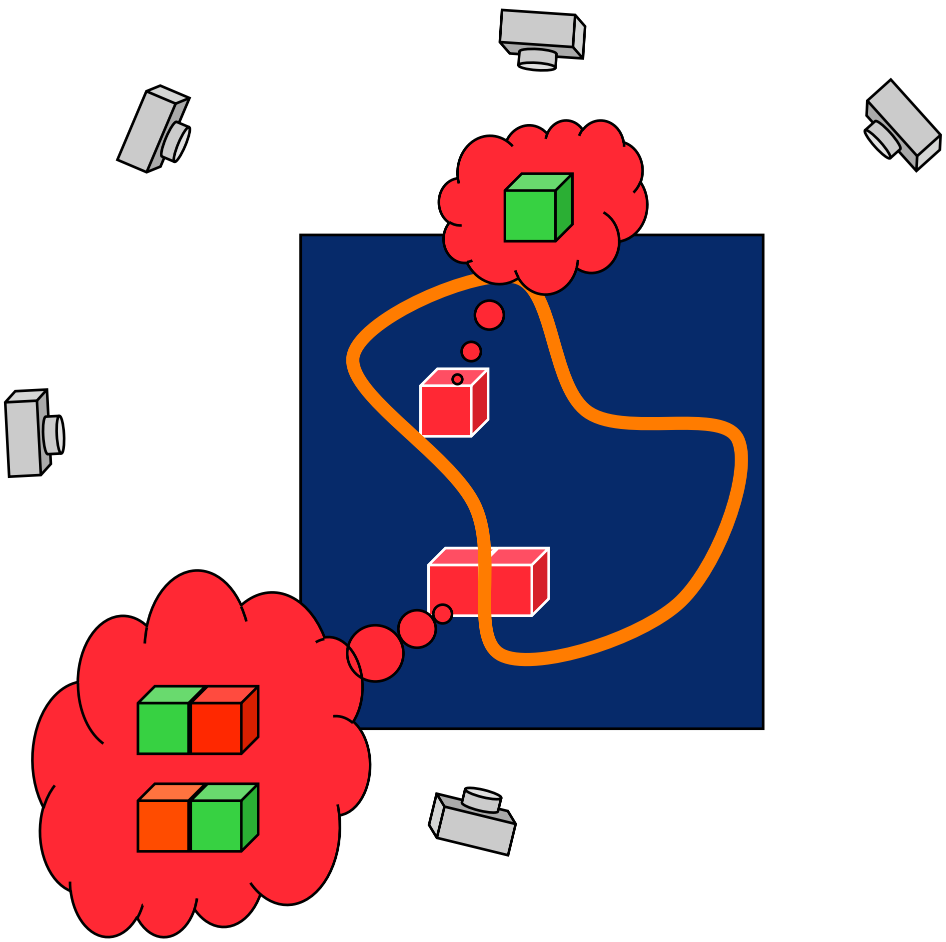
George Vogiatzis, Philip H. S. Torr
Roberto Cipolla

CVPR 2005

Volumetric Graph cuts

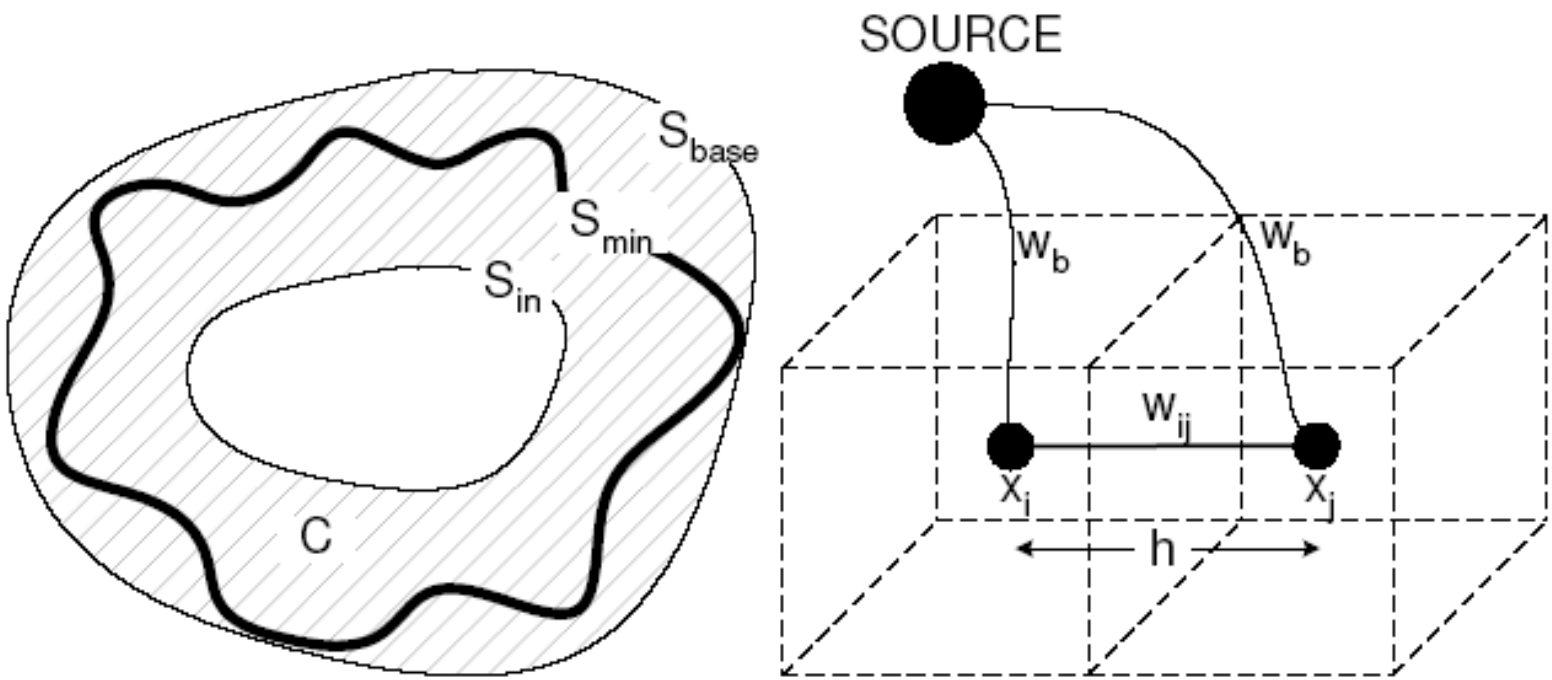


3D MRF for 3D modelling

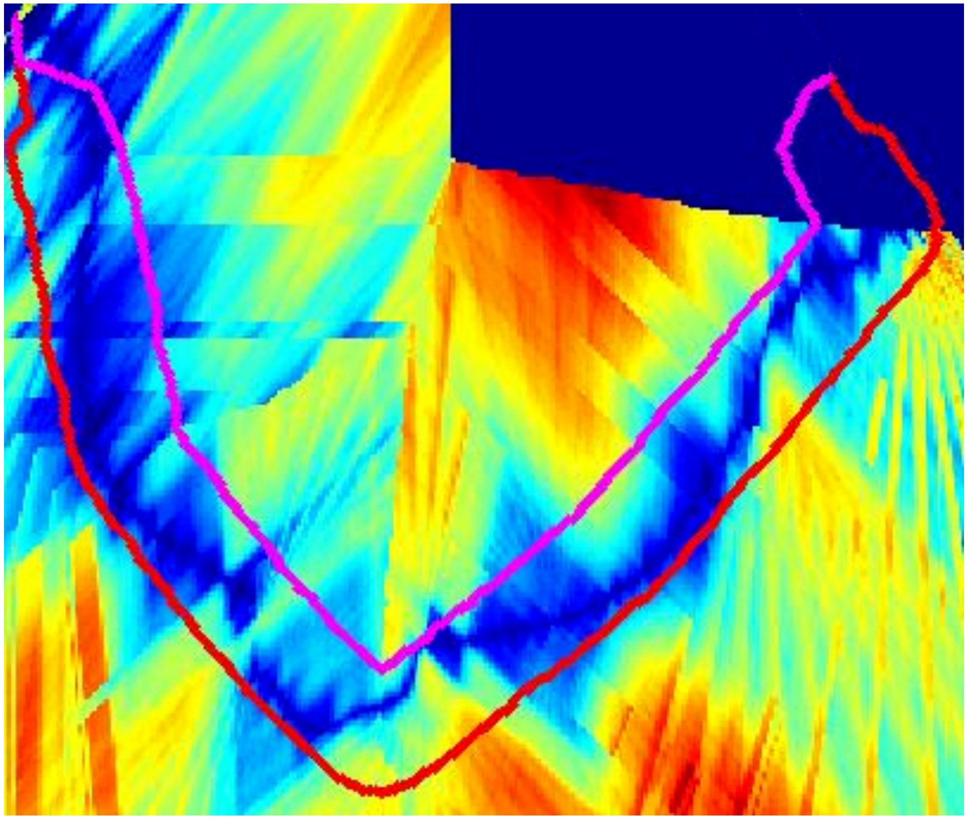


- Labelling cost:
  - Constant bias towards being **foreground**
- Compatibility cost:
  - Pair of neighbour voxels prefers having opposite labels if photo-consistent region is between them

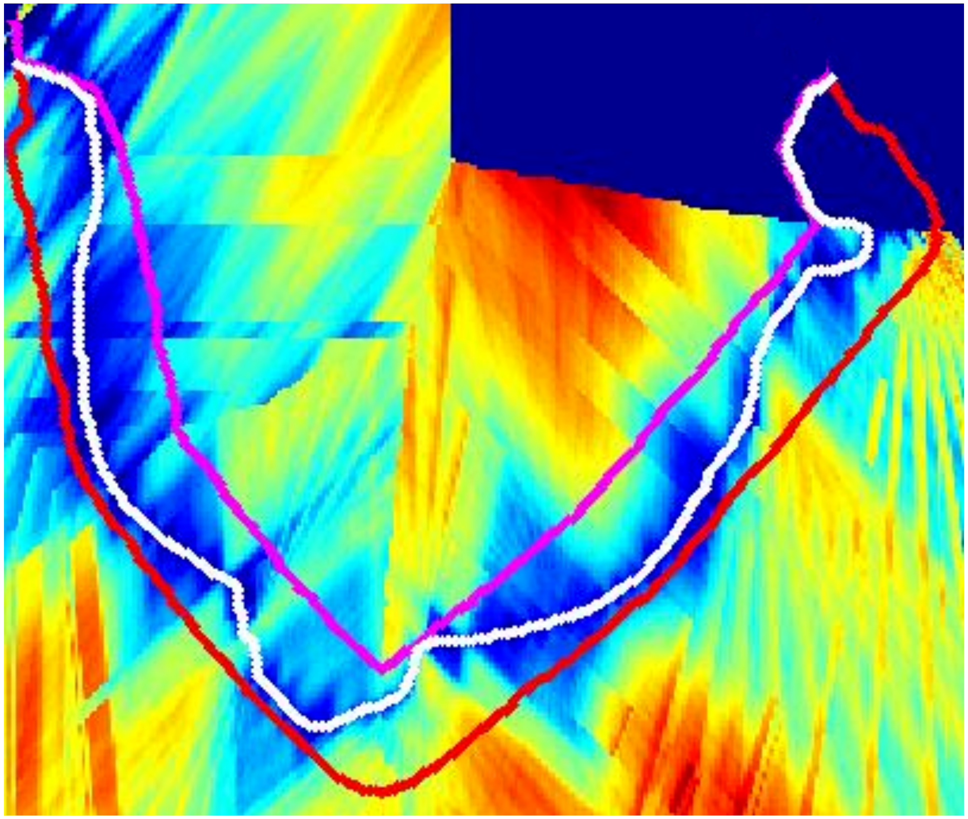
Graph



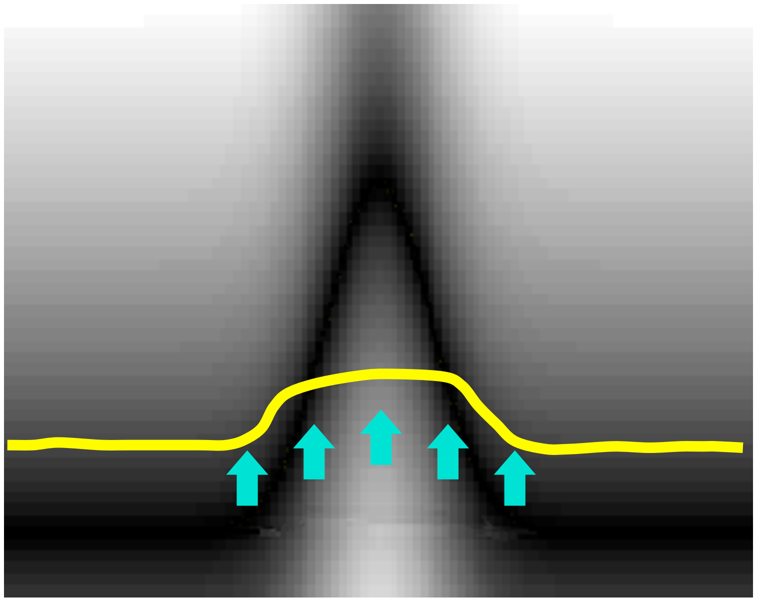
Face - Slice



Face - Slice with graph cut



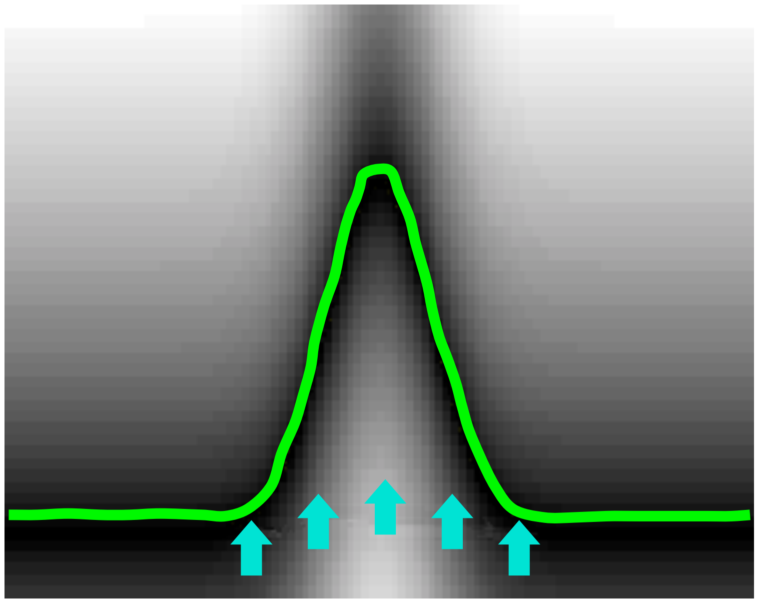
Protrusion problem



- ‘Ballooning’ force
  - favouring bigger volumes

L.D. Cohen and I. Cohen. Finite-element methods for active contour models and balloons for 2-d and 3-d images. *PAMI*, 15 (11):1131–1147, November 1993.

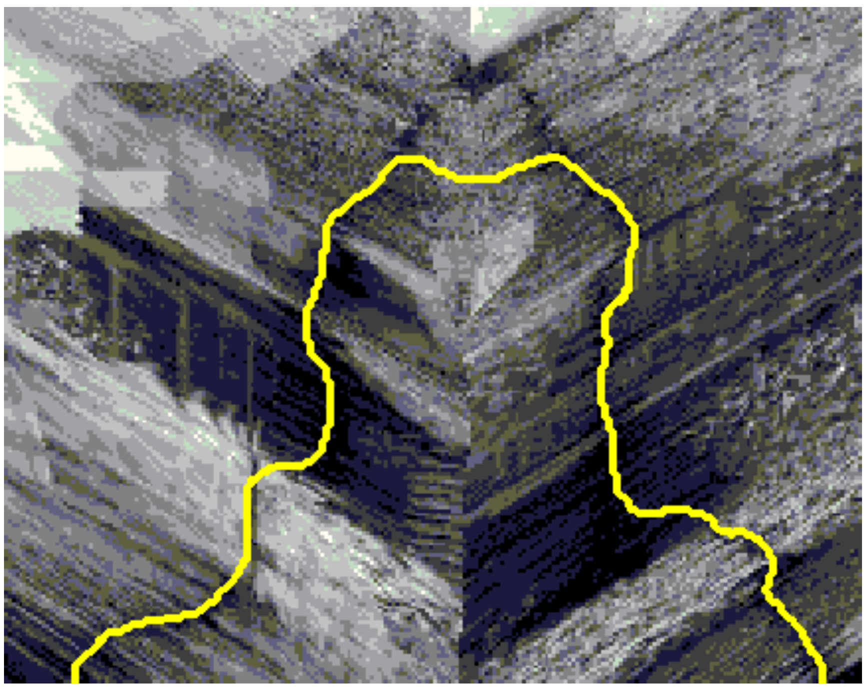
Protrusion problem

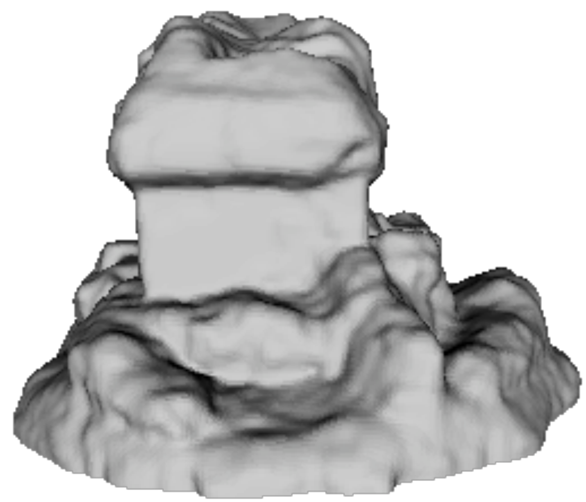


- ‘Ballooning’ force
  - favouring bigger volumes

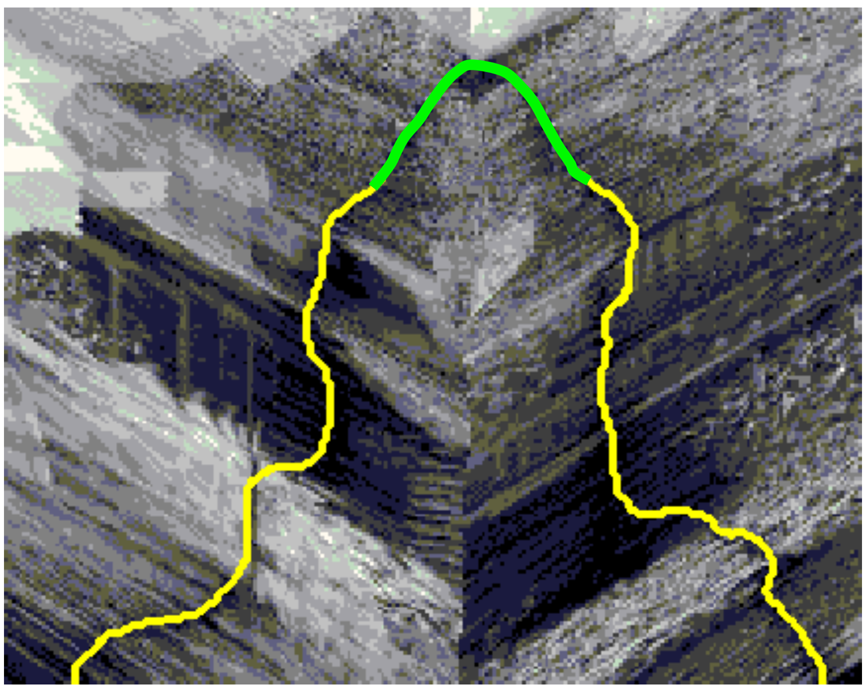
L.D. Cohen and I. Cohen. Finite-element methods for active contour models and balloons for 2-d and 3-d images. *PAMI*, 15 (11):1131–1147, November 1993.

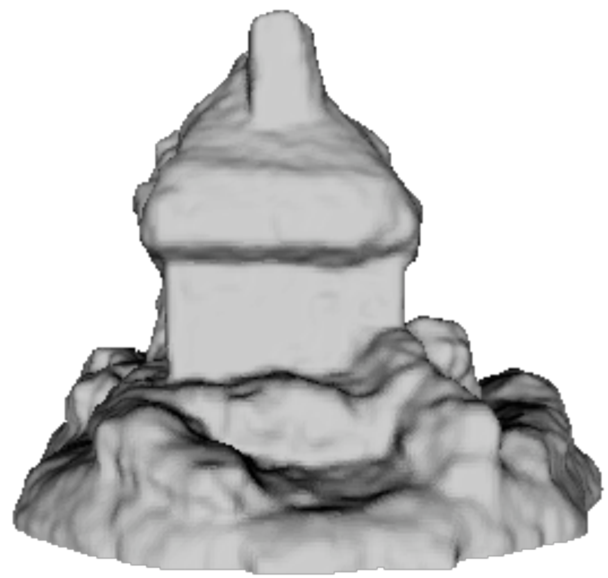
Protrusion problem



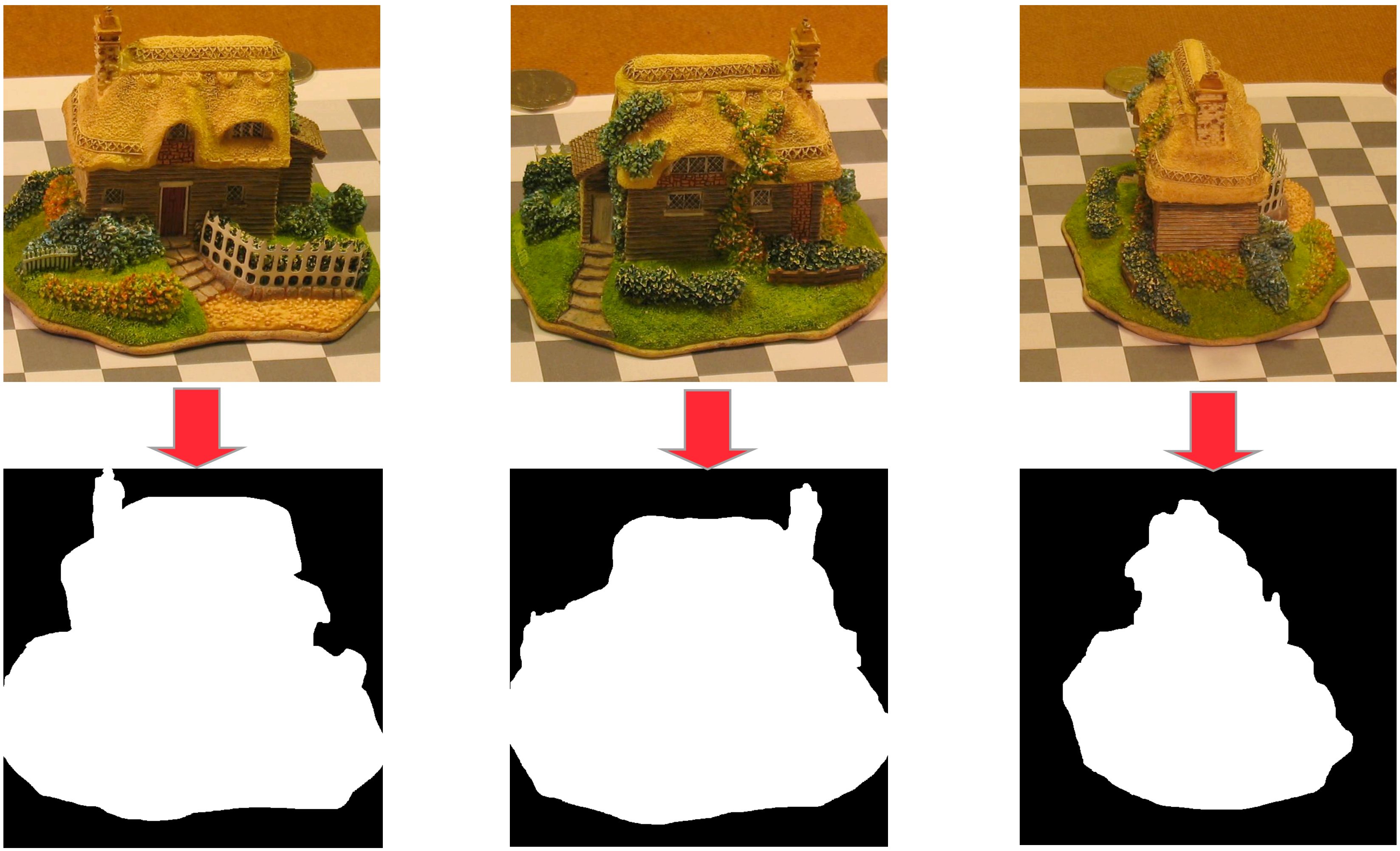


Protrusion problem

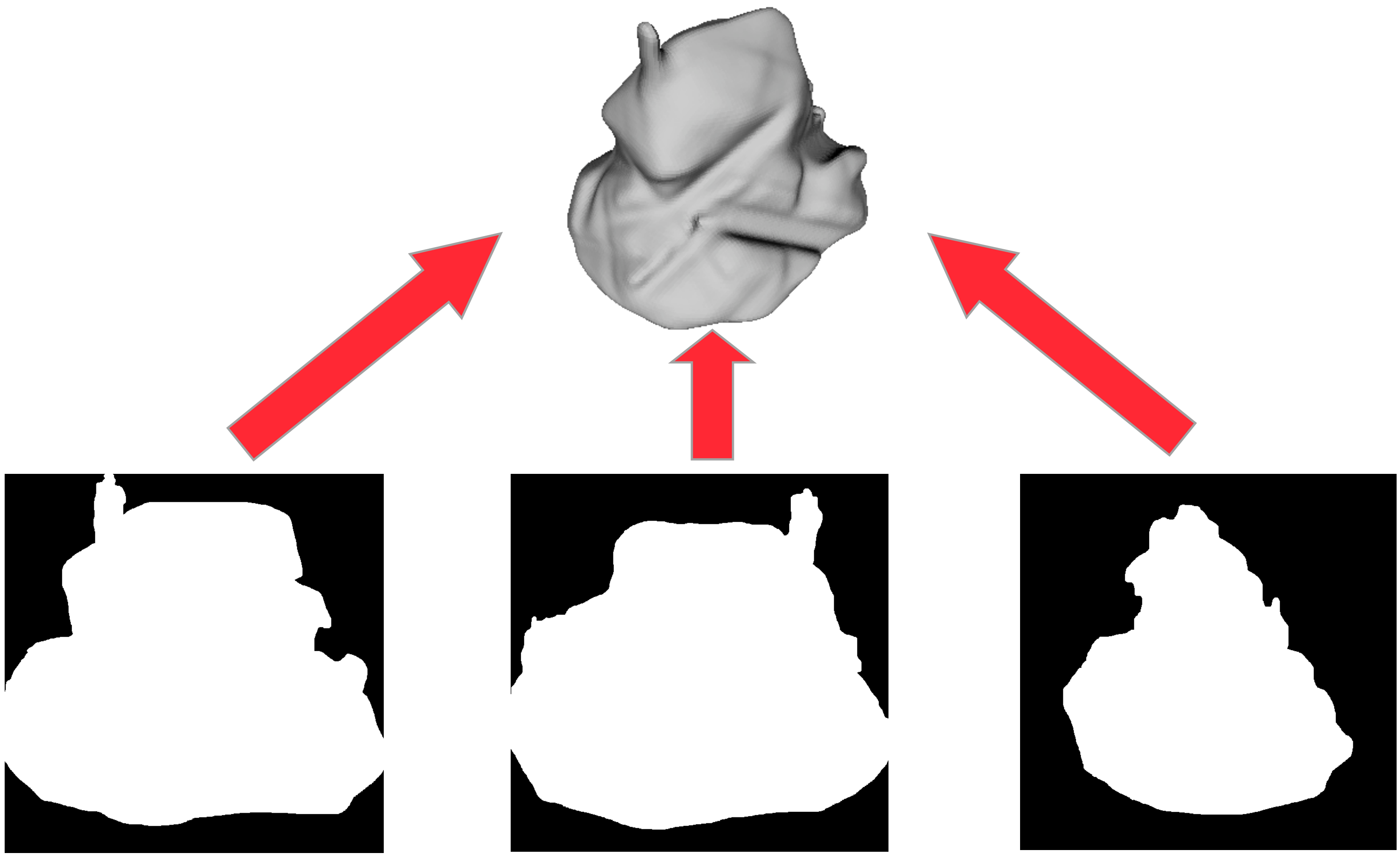




Results, Model House

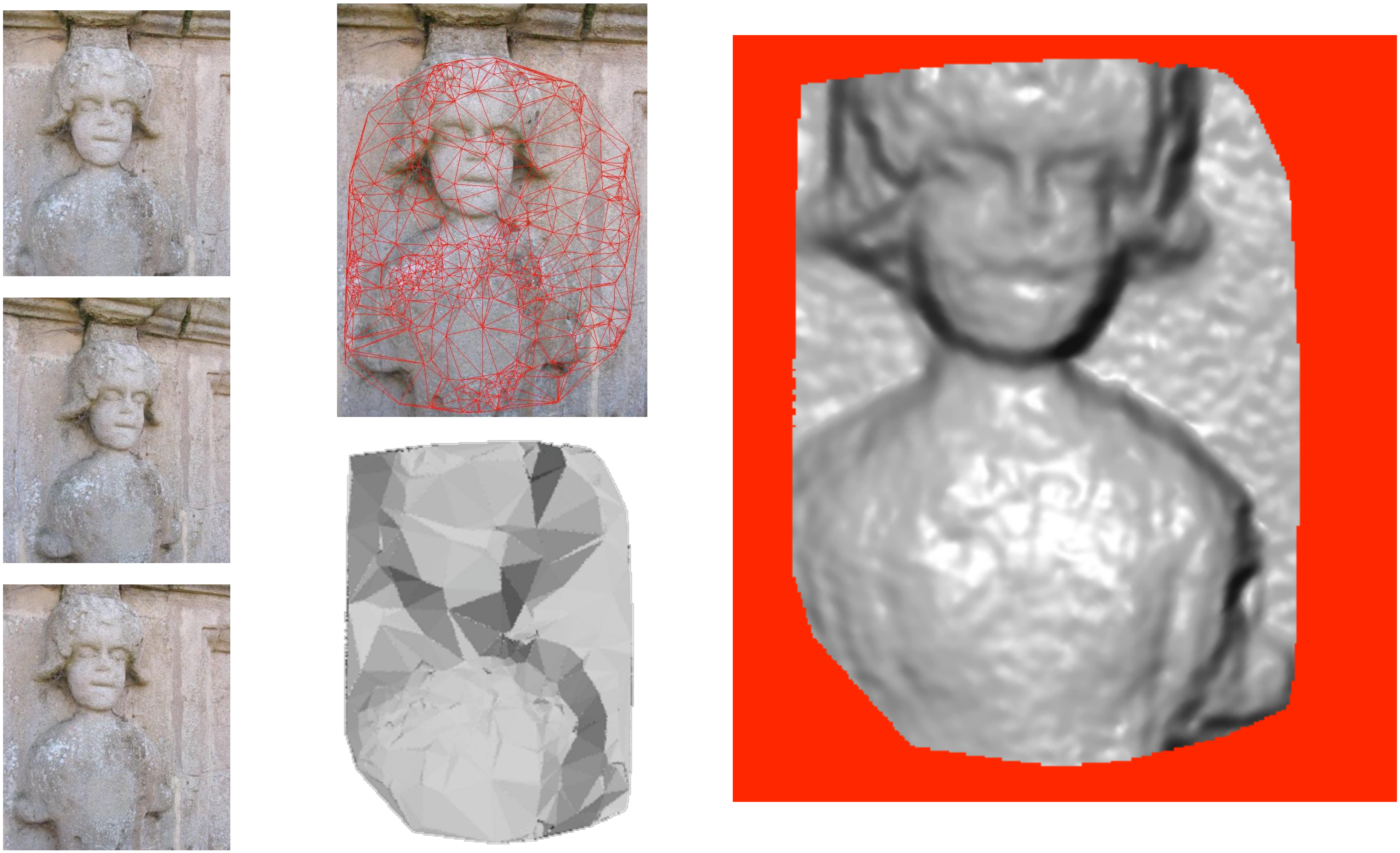


Results, Model House – Visual Hull

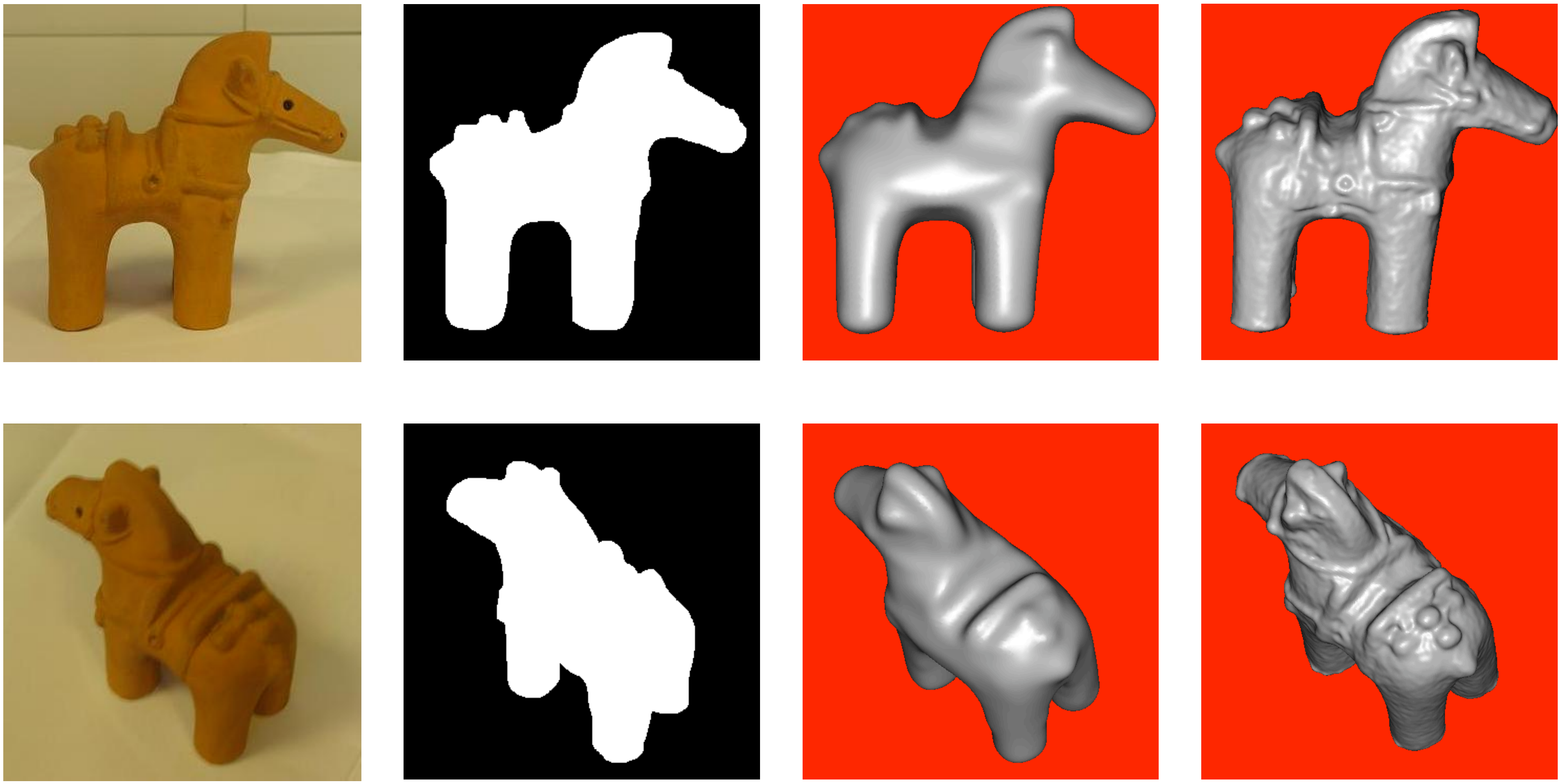


Results, Model House

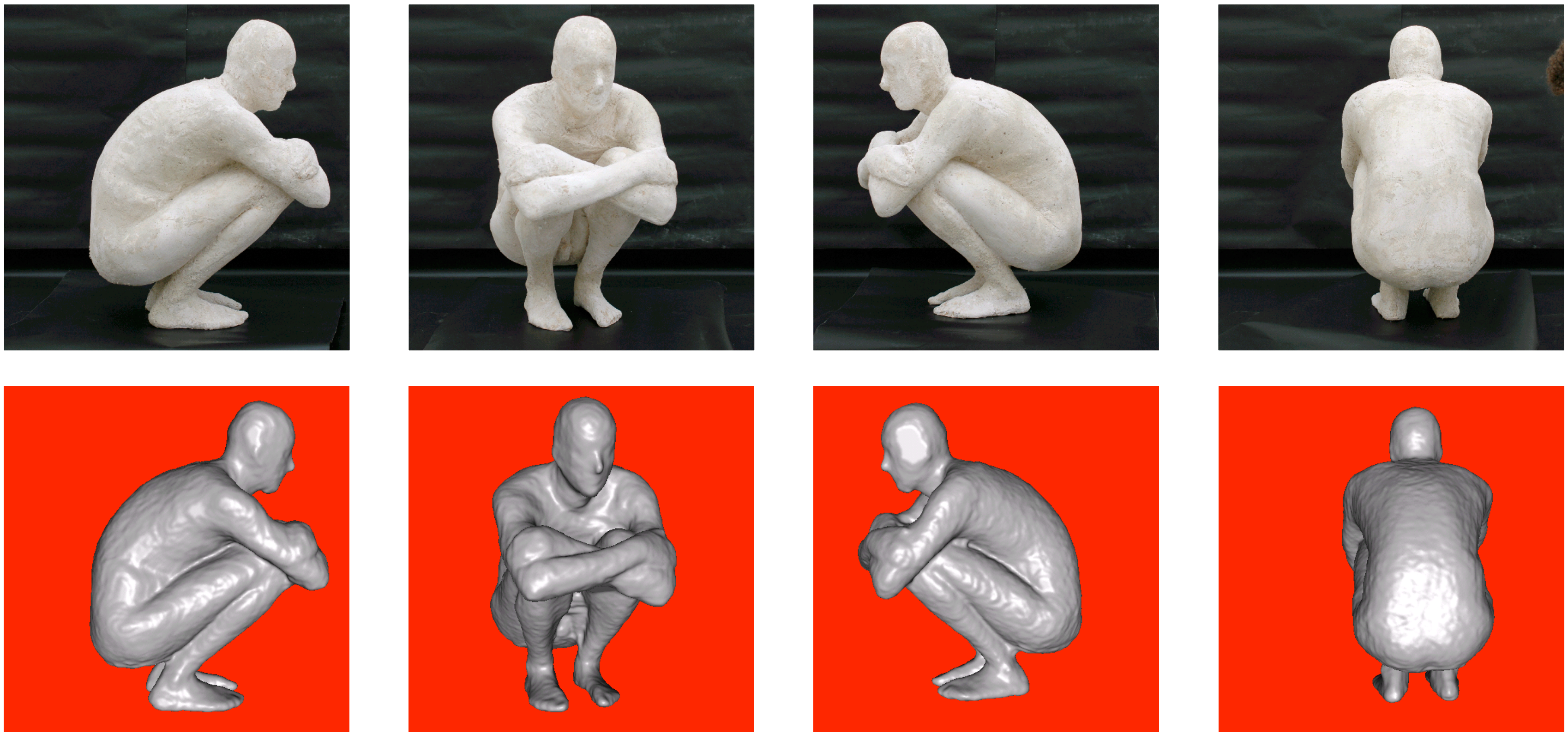
Results, Stone carving



Results, Haniwa

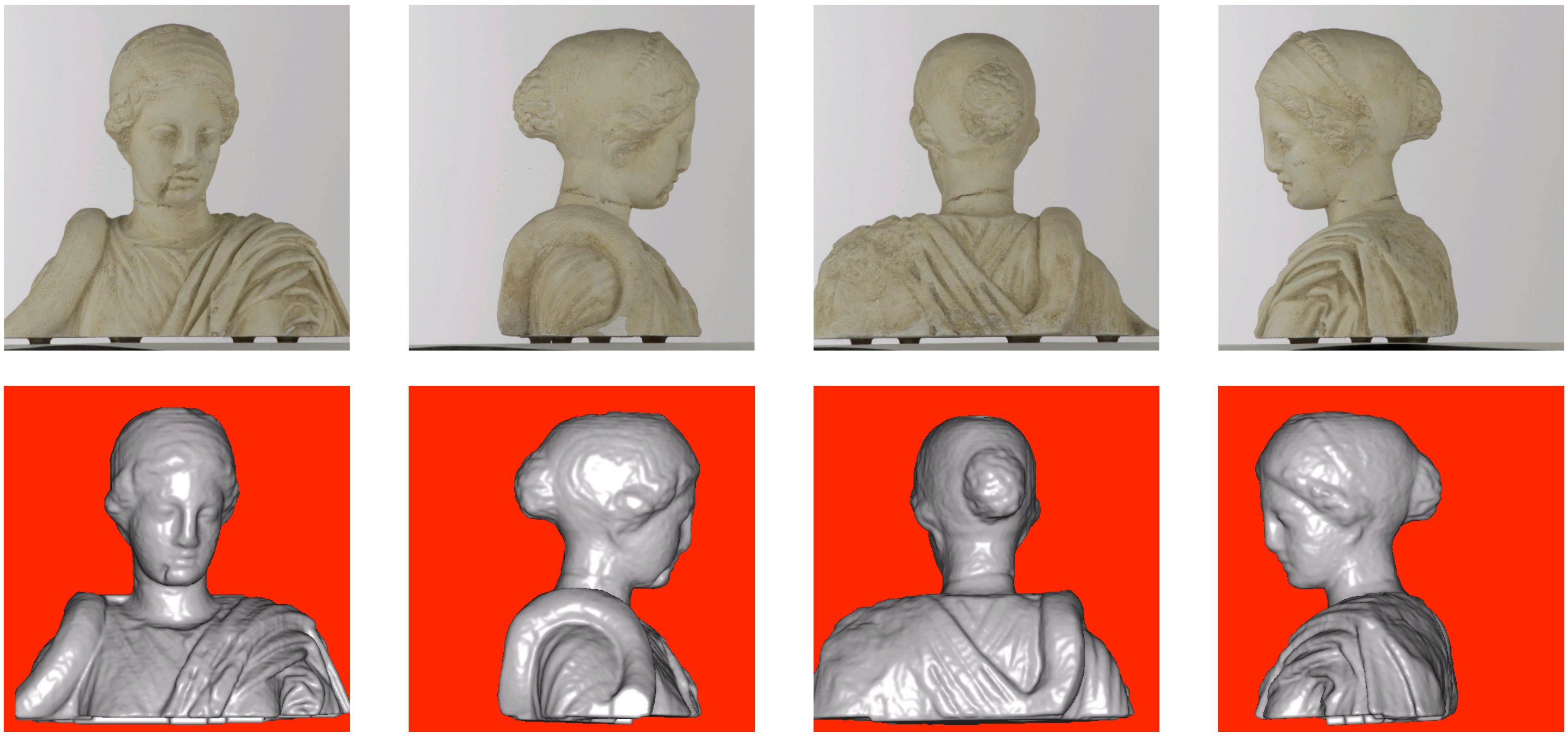


3D Models



OXFORD BROOKES UNIVERSITY

3D models



Middlebury evaluation (temple)

	Accuracy / Completeness		
	Full (312 images)	Ring (47 images)	SparseRing (16 images)
Hernandez [10]	0.36mm / 99.7%	0.52mm / 99.5%	0.75mm / 95.3%
Goesele [9]	0.42mm / 98.0%	0.61mm / 86.2%	0.87mm / 56.6%
Hornung [12]	0.58mm / 98.7%	–	–
Pons [20]	–	0.60mm / 99.5%	0.90mm / 95.4%
Furukawa [8]	0.65mm / 98.7%	0.58mm / 98.5%	0.82mm / 94.3%
Vogiatzis [29]	1.07mm / 90.7%	0.76mm / 96.2%	2.77mm / 79.4%
Present method	**0.50mm / 98.4%**	**0.64mm / 99.2%**	**0.69mm / 96.9%**

Advantages

- Accurate
  - sub-millimetre accuracy on sequence with ground truth
- Simple
  - Can work with about 15-30 images
- Fast
  - Approximately 45’ of computation for these models
  - We believe we can bring this down to few minutes

Extensions

- Boykov and co workers-flux
- Pollefeys and co workers-constraining graph cuts via the occluding contours…

User Assist: Research Issues

- We are allowed to take the video and pre-compute all our favourite SFM primitives
  - Calibration
  - Point Tracks
  - Volumetric Stereo
- Questions
  - What are the best user edits for this?
  - What algorithms could be useful for the edits?

Graph Cuts for Image Segmentation

Part I: *Boykov and Jolly ICCV 2001*

Part II: How to set up graphs from MRF

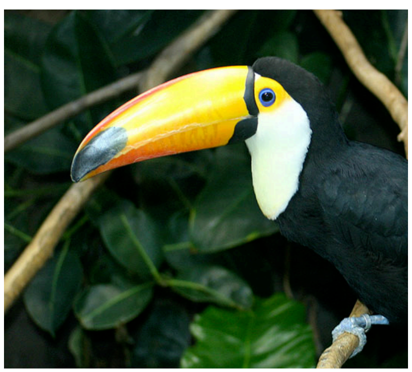
MRF for Image Segmentation

Boykov et al. [ICCV 2001], Blake et al. [ECCV 2004]

$$\text{Energy}_{\text{MRF}} = E(\mathbf{x}) = \sum_{i \in S} \left(\phi(\mathbf{D}|\mathbf{x}_i) + \sum_{j \in N_i} \left(\phi(\mathbf{D}|\mathbf{x}_i, \mathbf{x}_j) + \psi(\mathbf{x}_i, \mathbf{x}_j) \right) \right) + \text{const}$$

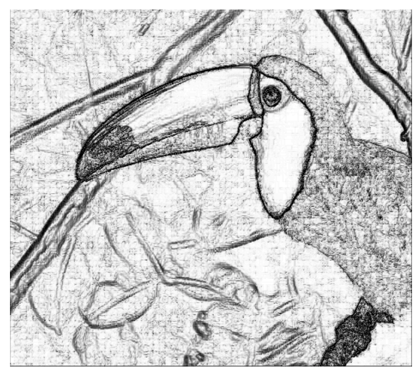
Unary likelihood Contrast Term Potts Model Prior

MAP solution $\mathbf{x}^* = \arg\min_{\mathbf{x}} E(\mathbf{x})$



Data (**D**)

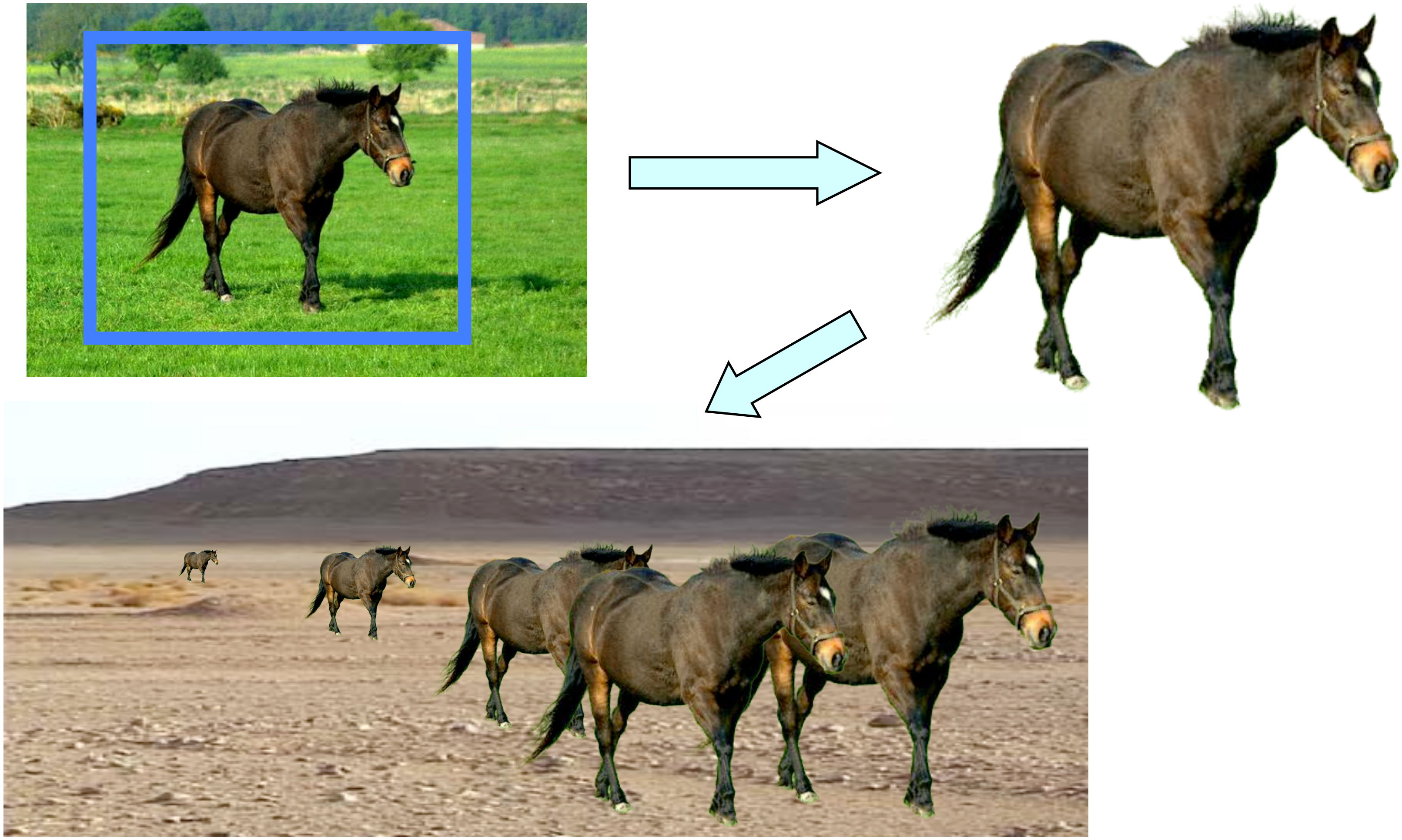
Unary likelihood



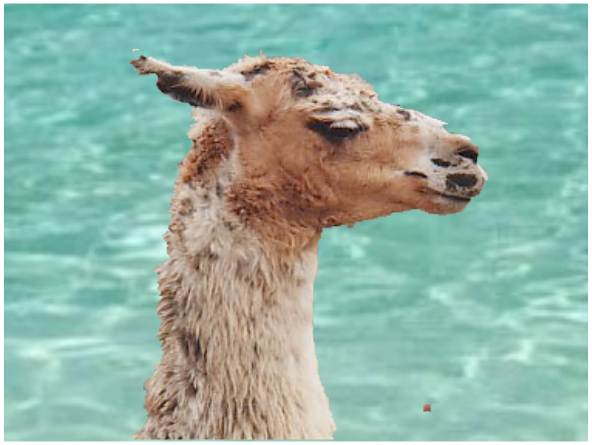
Pair-wise Terms

MAP Solution

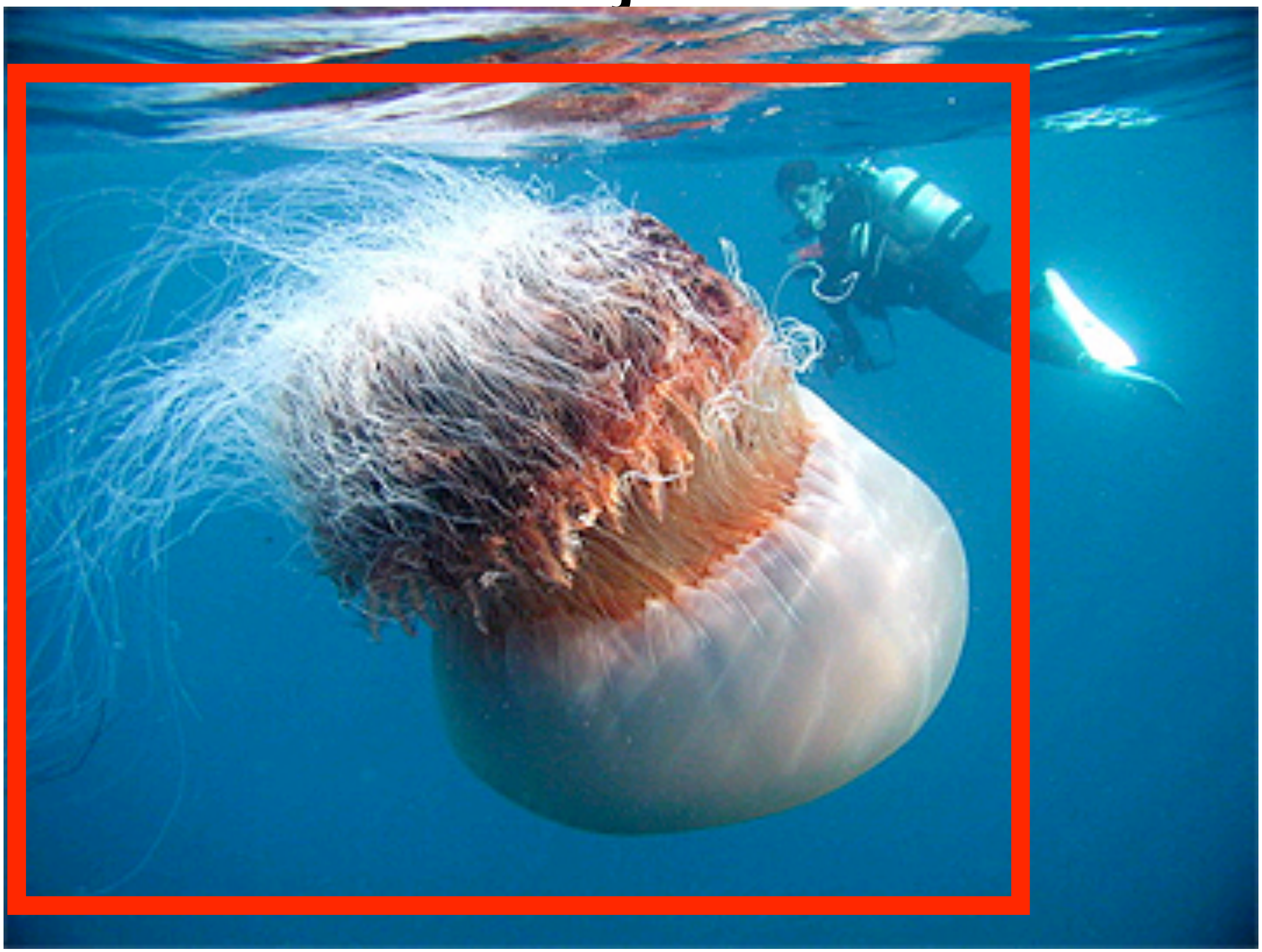
Segmentation with Alpha Matting



Grabcut demo



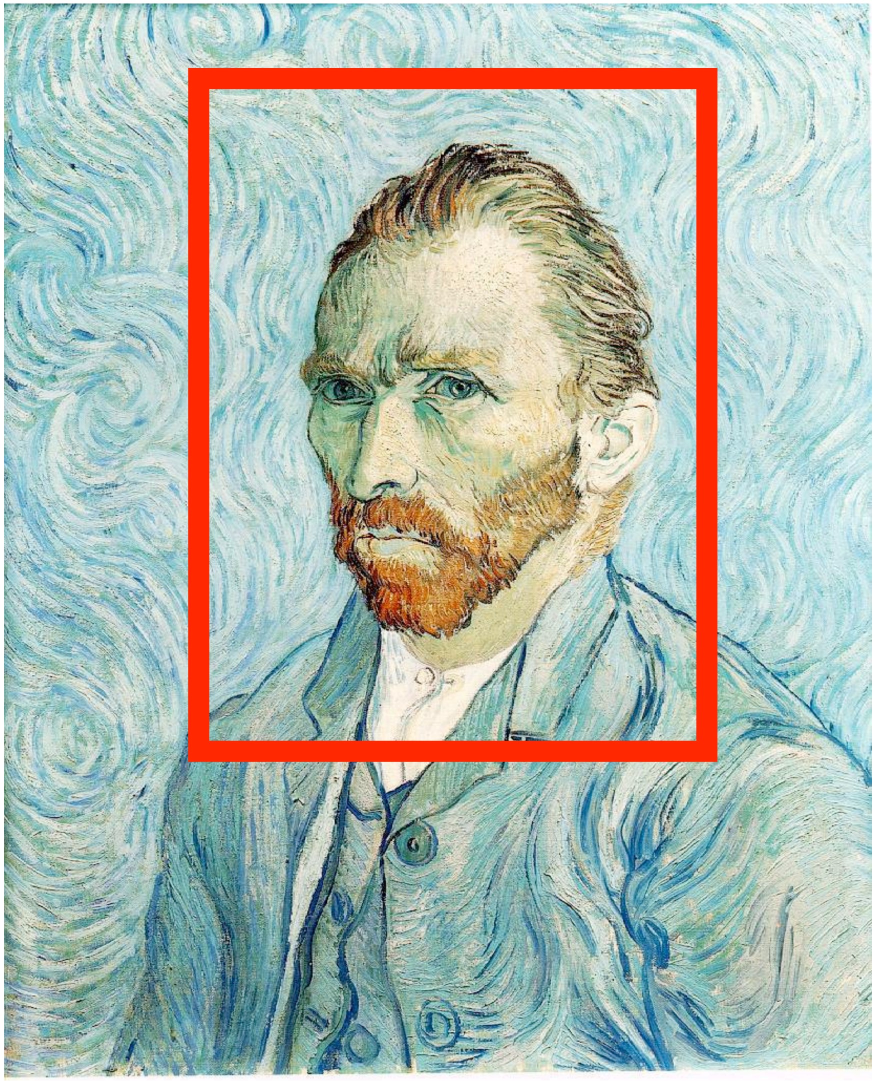
First place rectangle round object



Object plus alpha mask segmented by new secret method.

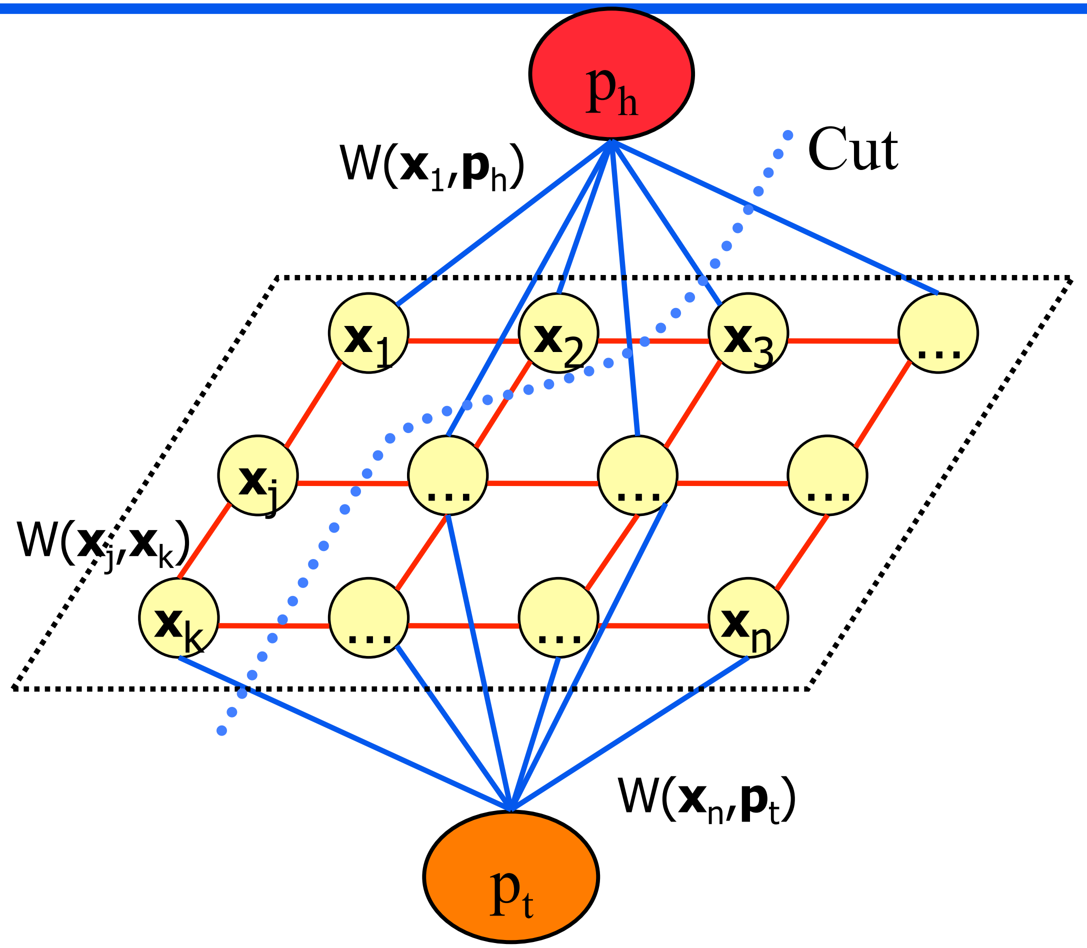
Watch out they are coming!!…note alpha mask

Chop off Van Gogh's head with one click!!



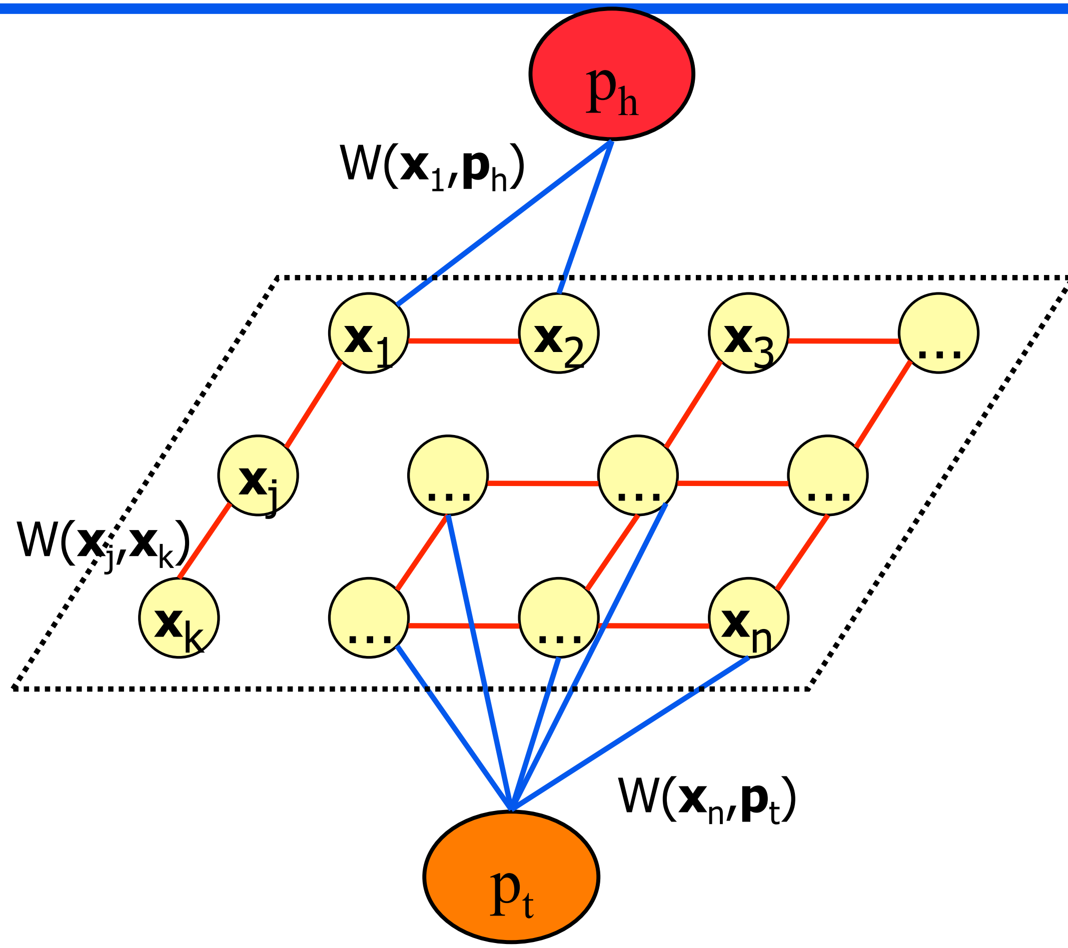
Graph Cuts

Consider the case of two segments.



- $W(\mathbf{x}_i,p_j) \propto$ appearance component
- $W(\mathbf{x}_j,\mathbf{x}_k) \propto$ boundary component

Graph Cuts

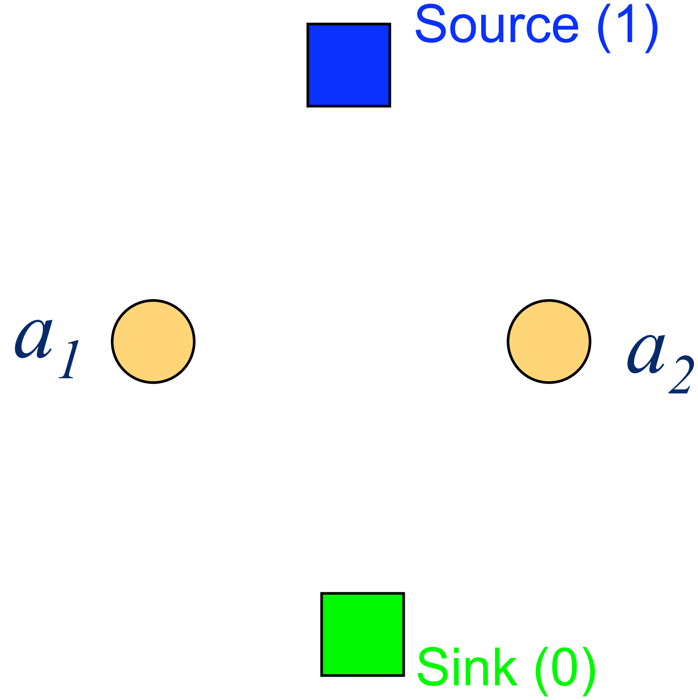


p_h
$W(\mathbf{x}_1,\mathbf{p}_h)$
$\mathbf{x}_1$
$\mathbf{x}_2$
$\mathbf{x}_3$
…
$\mathbf{x}_j$
…
…
…
$W(\mathbf{x}_j,\mathbf{x}_k)$
$\mathbf{x}_k$
…
…
$\mathbf{x}_n$
$W(\mathbf{x}_n,\mathbf{p}_t)$
p_t

Energy Minimization using Graph cuts

What really happens? Building the graph

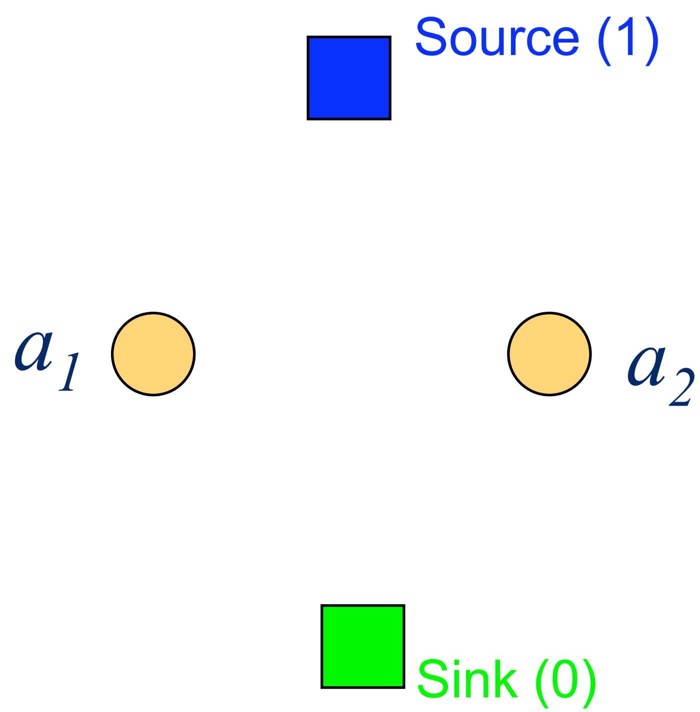
$\mathrm{E}_{\mathrm{MRF}}(a_1,a_2)$



Energy Minimization using Graph cuts

What really happens? Building the graph

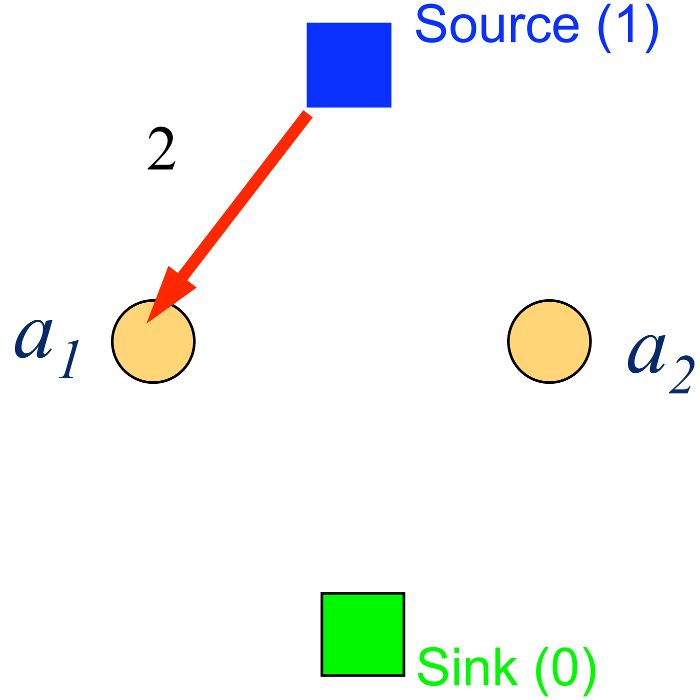
If we cut a link to source or sink this indicates label



Energy Minimization using Graph cuts

What really happens? Building the graph

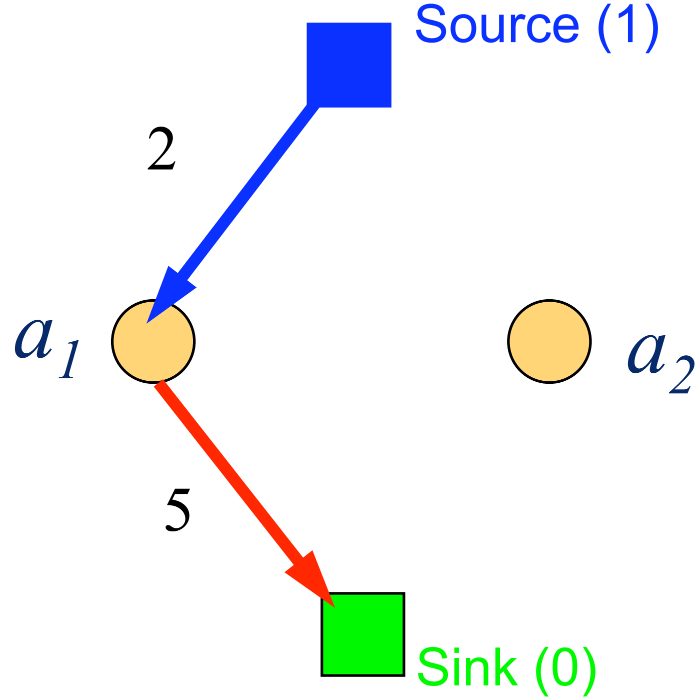
$$E_{MRF}(a_1,a_2) = 2a_1$$



Energy Minimization using Graph cuts

What really happens? Building the graph

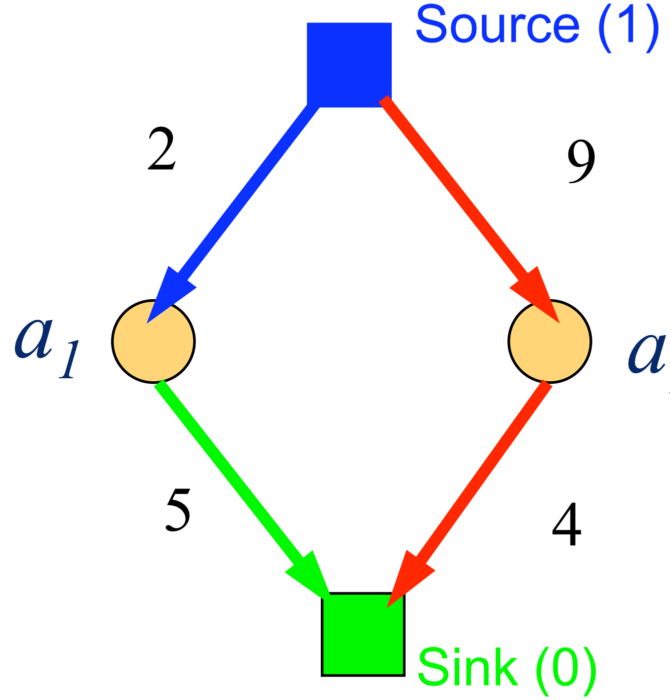
$$E_{MRF}(a_1,a_2) = 2a_1 + 5\bar{a}_1$$



Energy Minimization using Graph cuts

What really happens? Building the graph

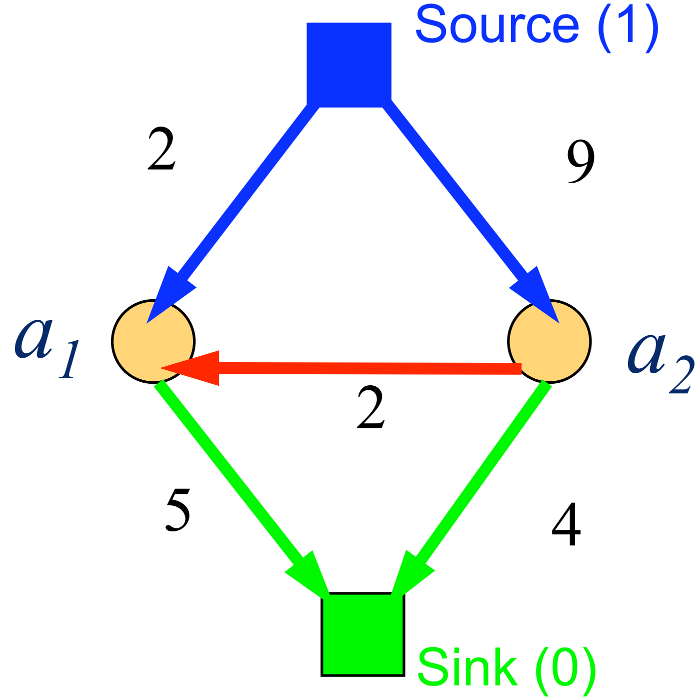
$$E_{\text{MRF}}(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2$$



Energy Minimization using Graph cuts

What really happens? Building the graph

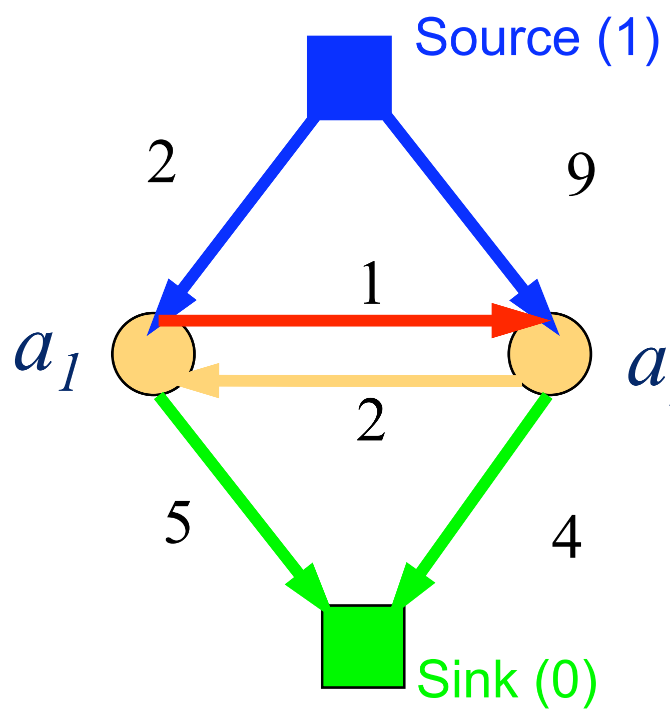
$$E_{MRF}(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2$$



Energy Minimization using Graph cuts

What really happens? Building the graph

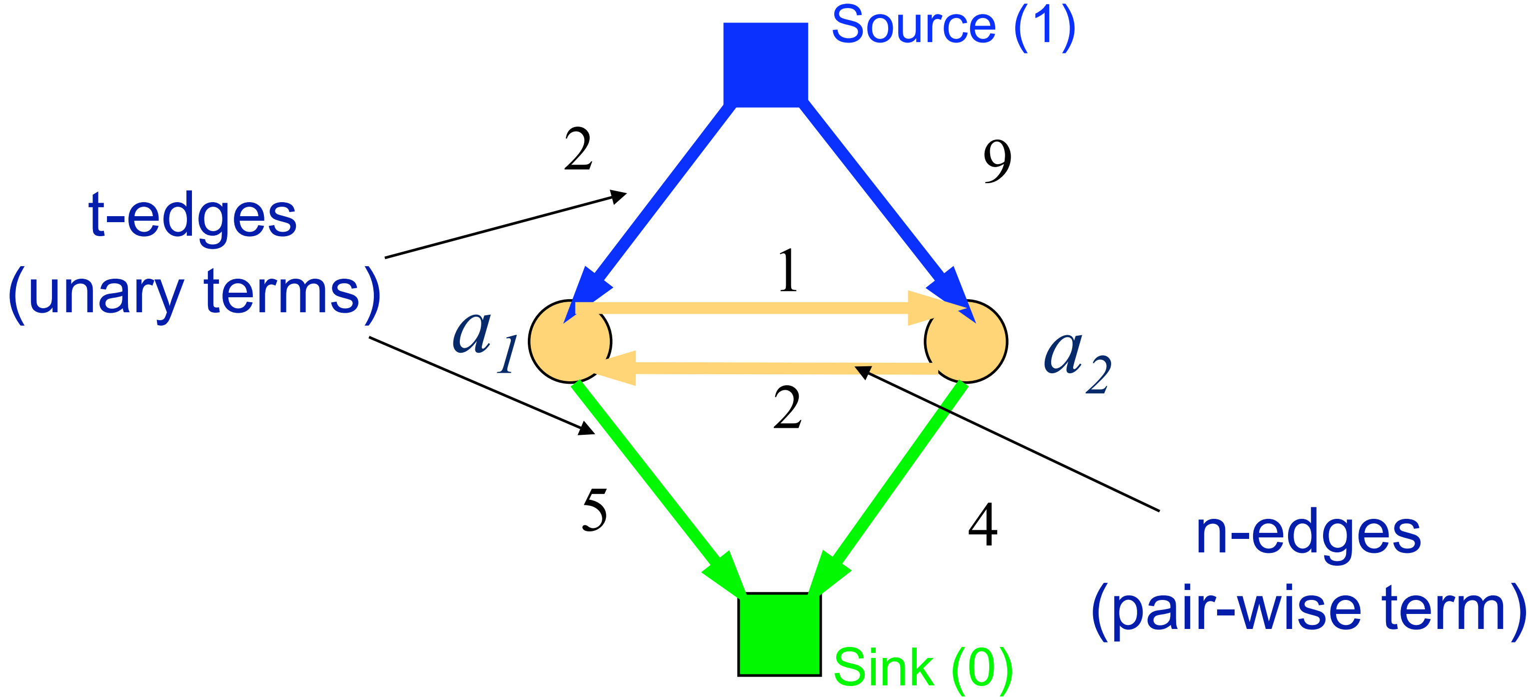
$$E_{MRF}(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1 a_2$$



Energy Minimization using Graph cuts

What really happens? Building the graph

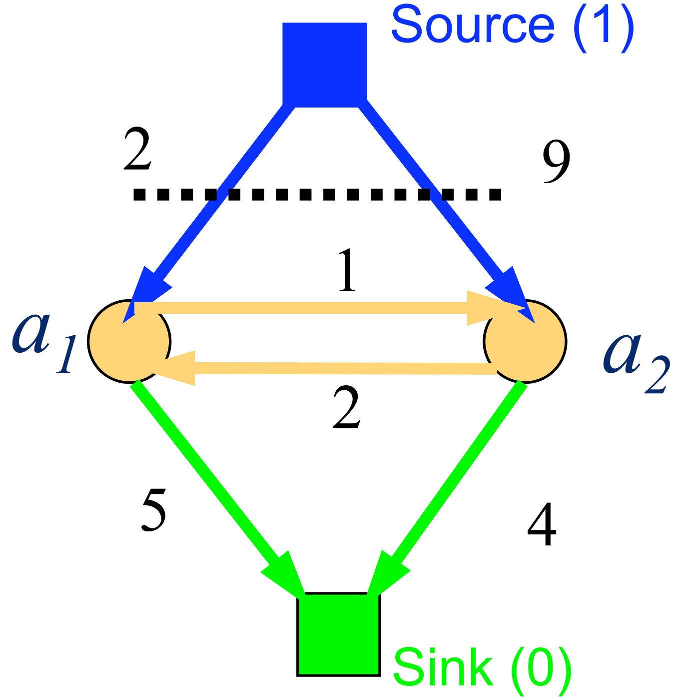
$$E_{MRF}(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1 a_2$$



Energy Minimization using Graph cuts

What really happens? Building the graph

$$E_{MRF}(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1 a_2$$



$a_1 = 1 \quad a_2 = 1$

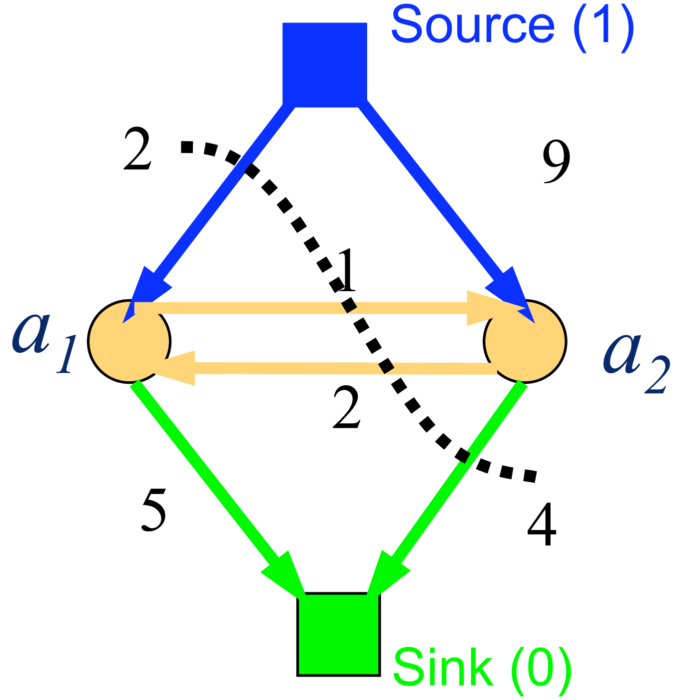
Cost of st-cut = 11

$E_{MRF}(1,1) = 11$

Energy Minimization using Graph cuts

What really happens? Building the graph

$$E_{MRF}(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1 a_2$$



$a_1 = 1 \quad a_2 = 0$

Cost of st-cut = 8

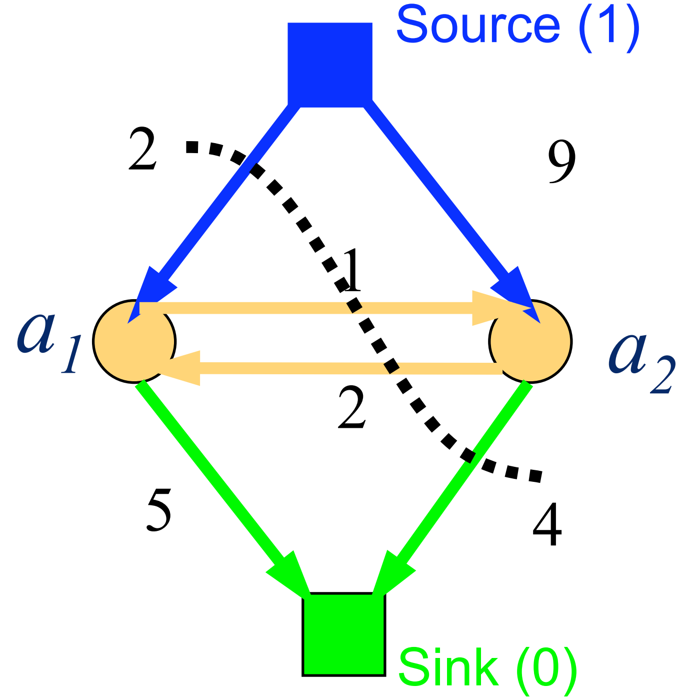
$E_{MRF}(1,0) = 8$

Energy Minimization using Graph cuts

What really happens? Building the graph

$$E_{MRF}(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1a_2$$

+ CONSTANT TERM K



$a_1 = 1 \quad a_2 = 0$

Cost of st-cut = 8

$E_{MRF}(1,0) = 8$

Energy Minimization using Graph cuts

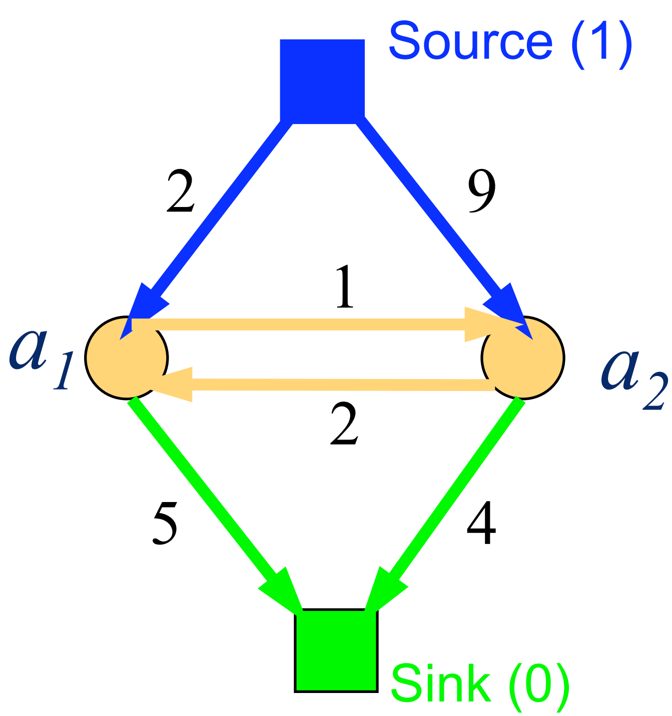
Posiform

$$E_{MRF}(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1 a_2$$

+ CONSTANT TERM K

Posiform is a multilinear polynomials in binary variables with positive coefficients

Computing the st-mincut using max-flow



Find the maximum flow from the source node s to the sink node t subject to the edge capacity and flow balance constraints:

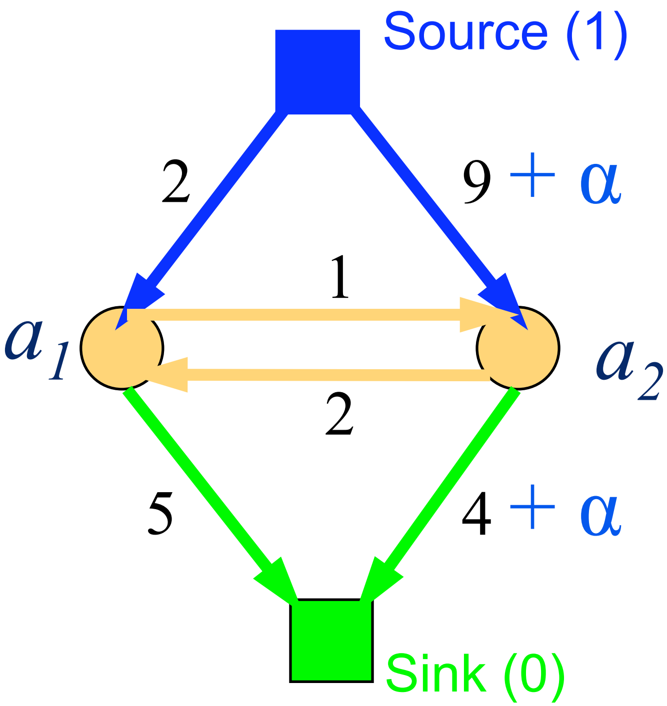
1

$$f_{ij} \leq c_{ij} \quad \forall (i,j) \in E$$

2

$$\sum_{i \in N(x) \backslash \{s,t\}} (f_{xi} - f_{ix}) = f_{sx} - f_{xt} \quad \forall x \in V \backslash \{s,t\}$$

The Max-flow problem



Key Observation

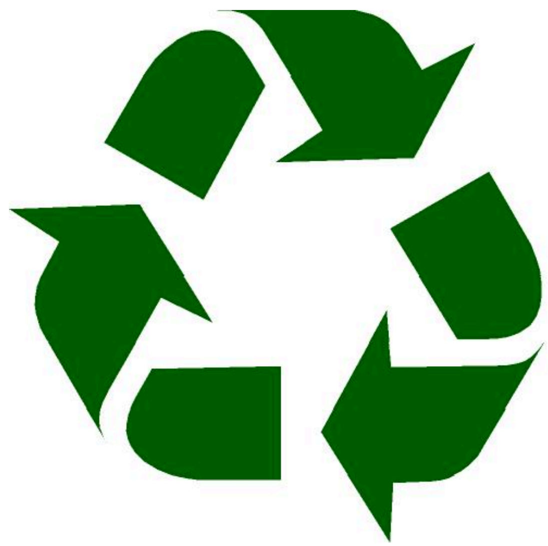
Adding a constant to both the t-edges of a node is equivalent to adding a constant amount of flow and does not change the edges constituting the st-mincut.

Efficiently Solving Dynamic Markov Random Fields using Graph Cuts (ICCV 2005)

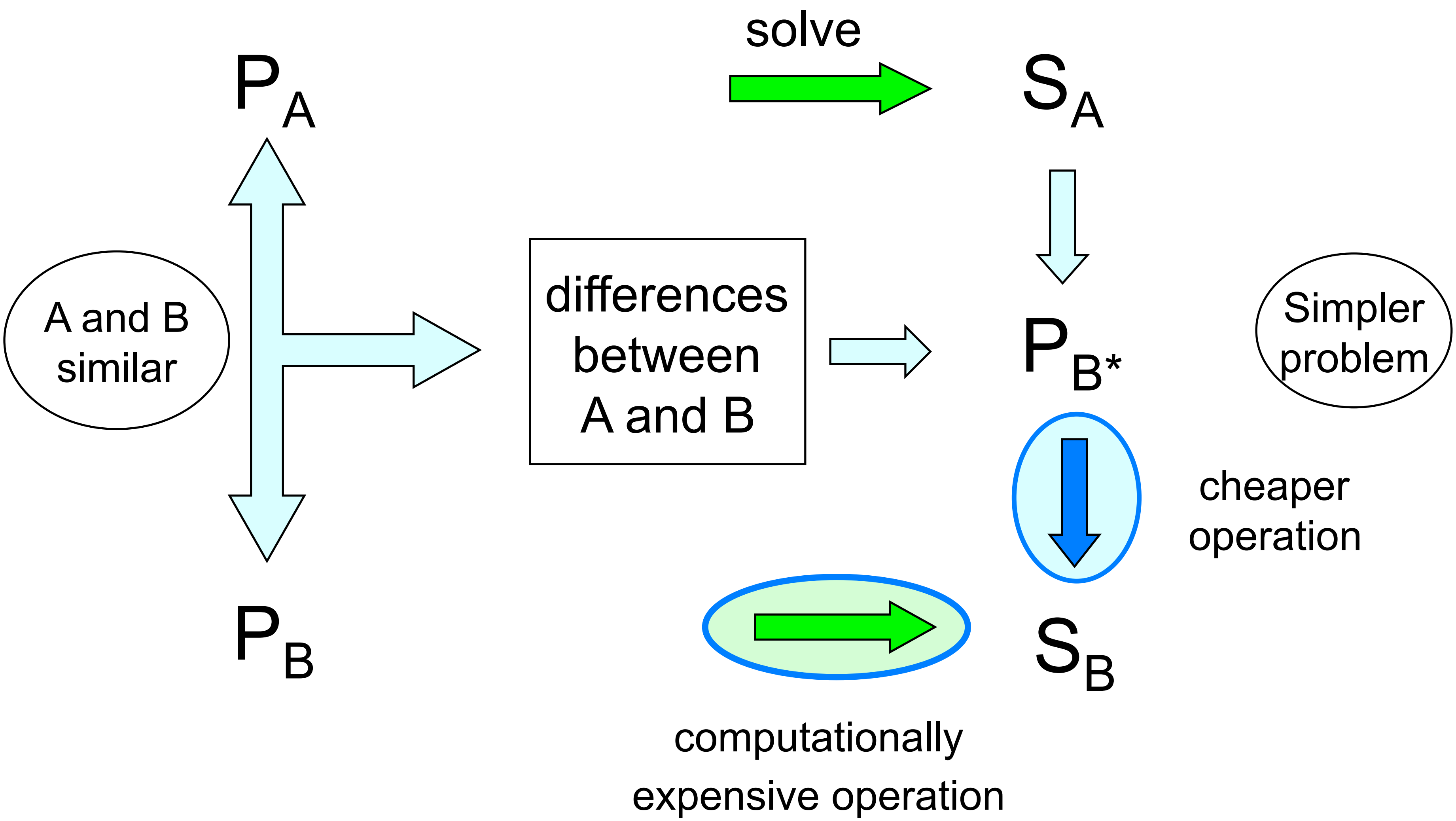
Pushmeet Kohli Philip H.S. Torr

Overview

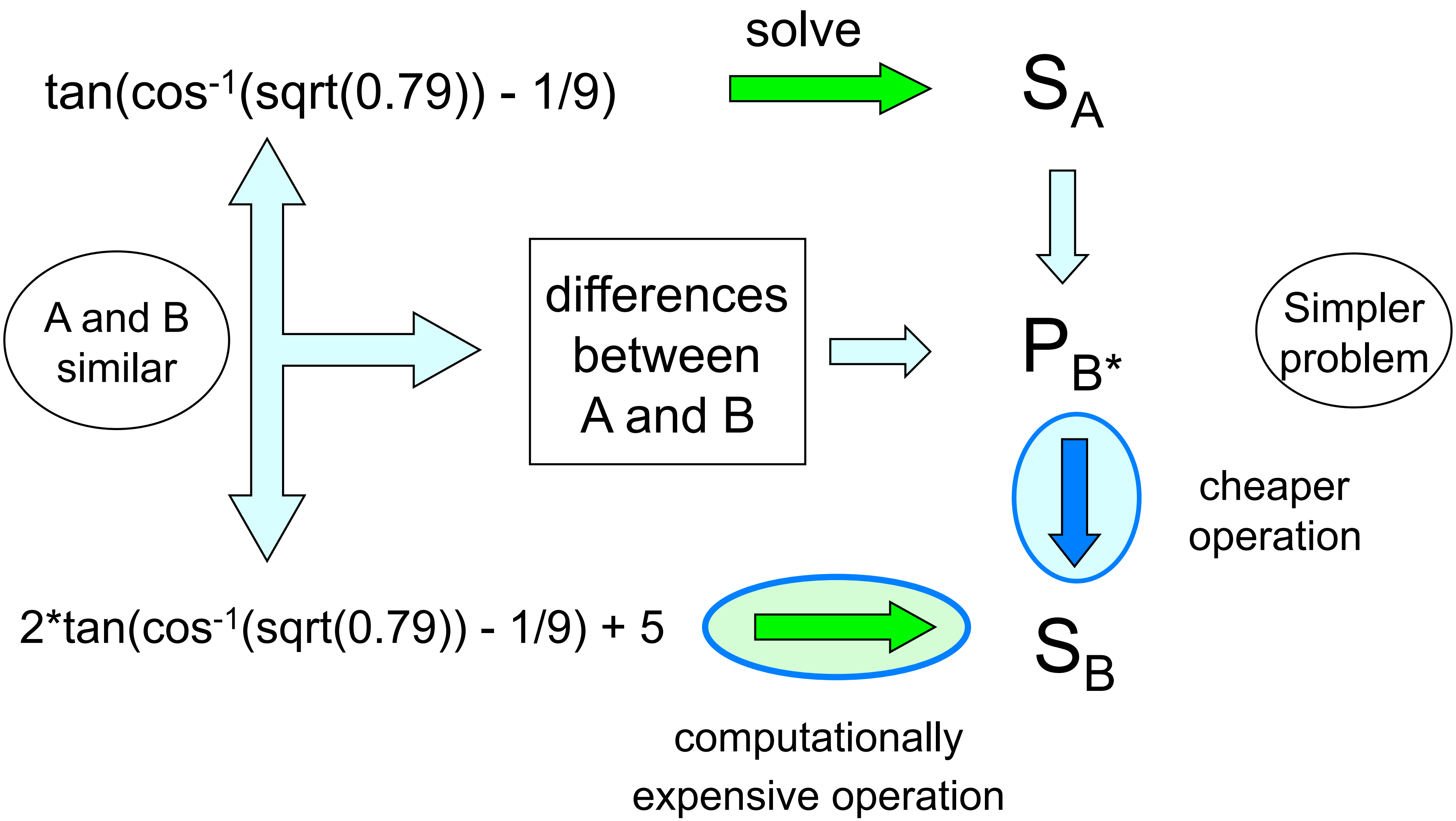
Recycling Computations



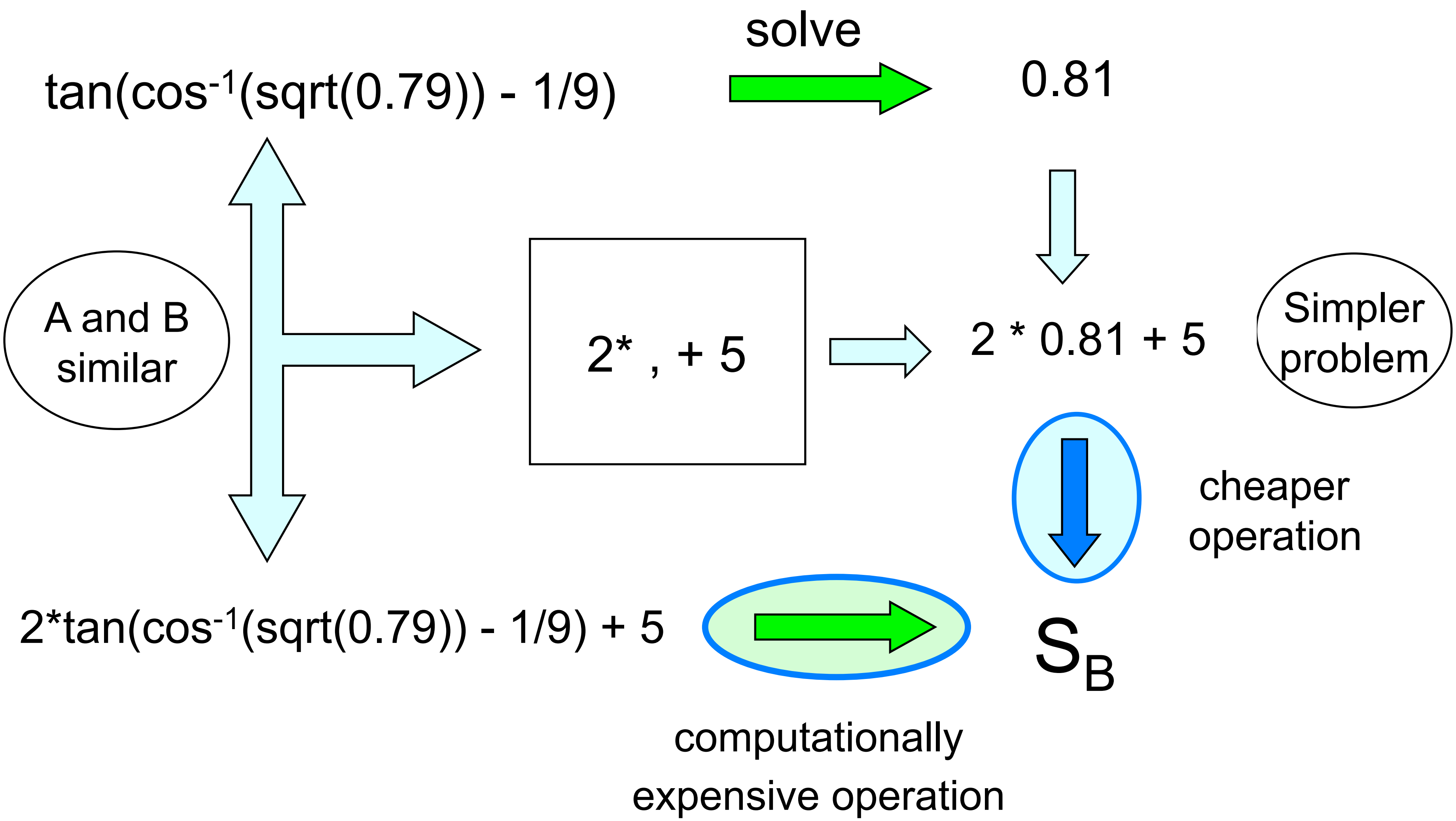
Overview


solve
P_A
S_A
A and B similar
differences between A and B
P_{B^*}
Simpler problem
cheaper operation
P_B
S_B
computationally expensive operation

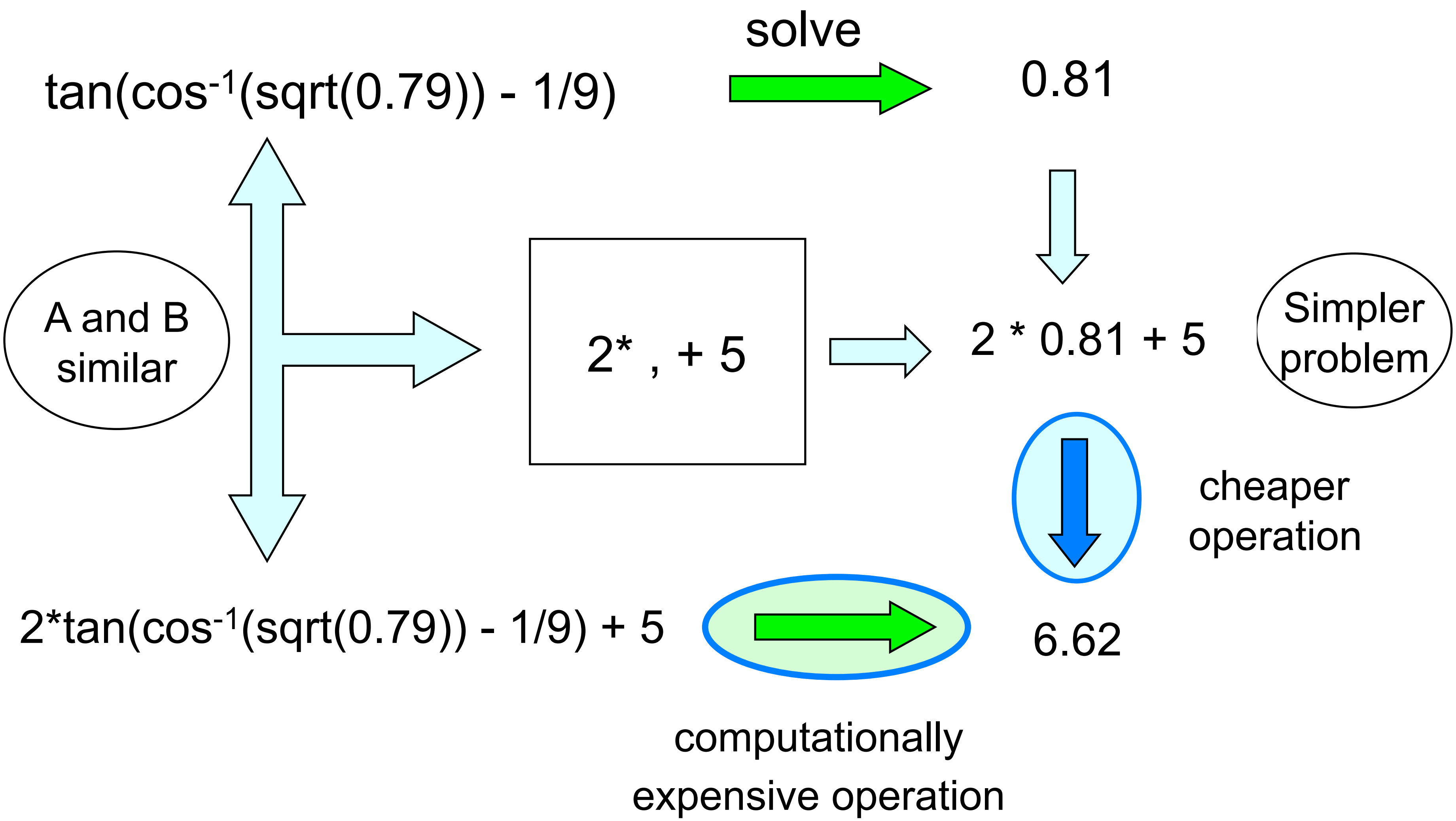
Overview


solve
tan(cos^{-1}(sqrt(0.79)) - 1/9)
S_A
A and B similar
differences between A and B
P_{B*}
Simpler problem
cheaper operation
2*tan(cos^{-1}(sqrt(0.79)) - 1/9) + 5
S_B
computationally expensive operation

Overview


solve
tan(cos-1(sqrt(0.79)) - 1/9)
0.81
A and B similar
2* , + 5
2 * 0.81 + 5
Simpler problem
cheaper operation
2*tan(cos-1(sqrt(0.79)) - 1/9) + 5
SB
computationally expensive operation

Overview


solve
tan(cos^{-1}(sqrt(0.79)) - 1/9)
0.81
A and B similar
2* , + 5
2 * 0.81 + 5
Simpler problem
cheaper operation
2*tan(cos^{-1}(sqrt(0.79)) - 1/9) + 5
6.62
computationally expensive operation

Our Contributions

- A fully dynamic algorithm for the st-mincut problem.

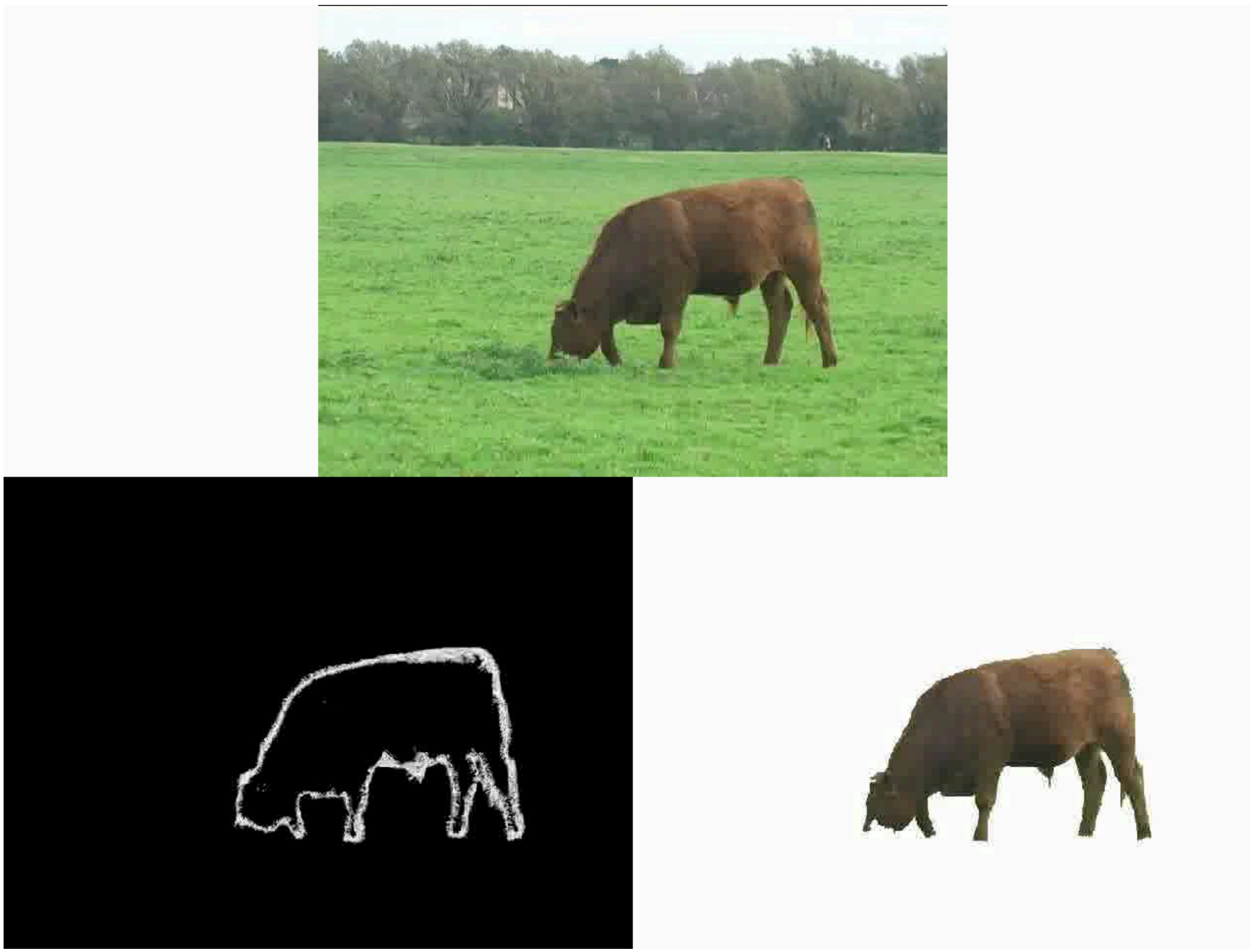
  - Exact
  - Minimize dynamic energy functions.
  - Running time proportional to the number of changes in the problem

- Efficient image segmentation in videos

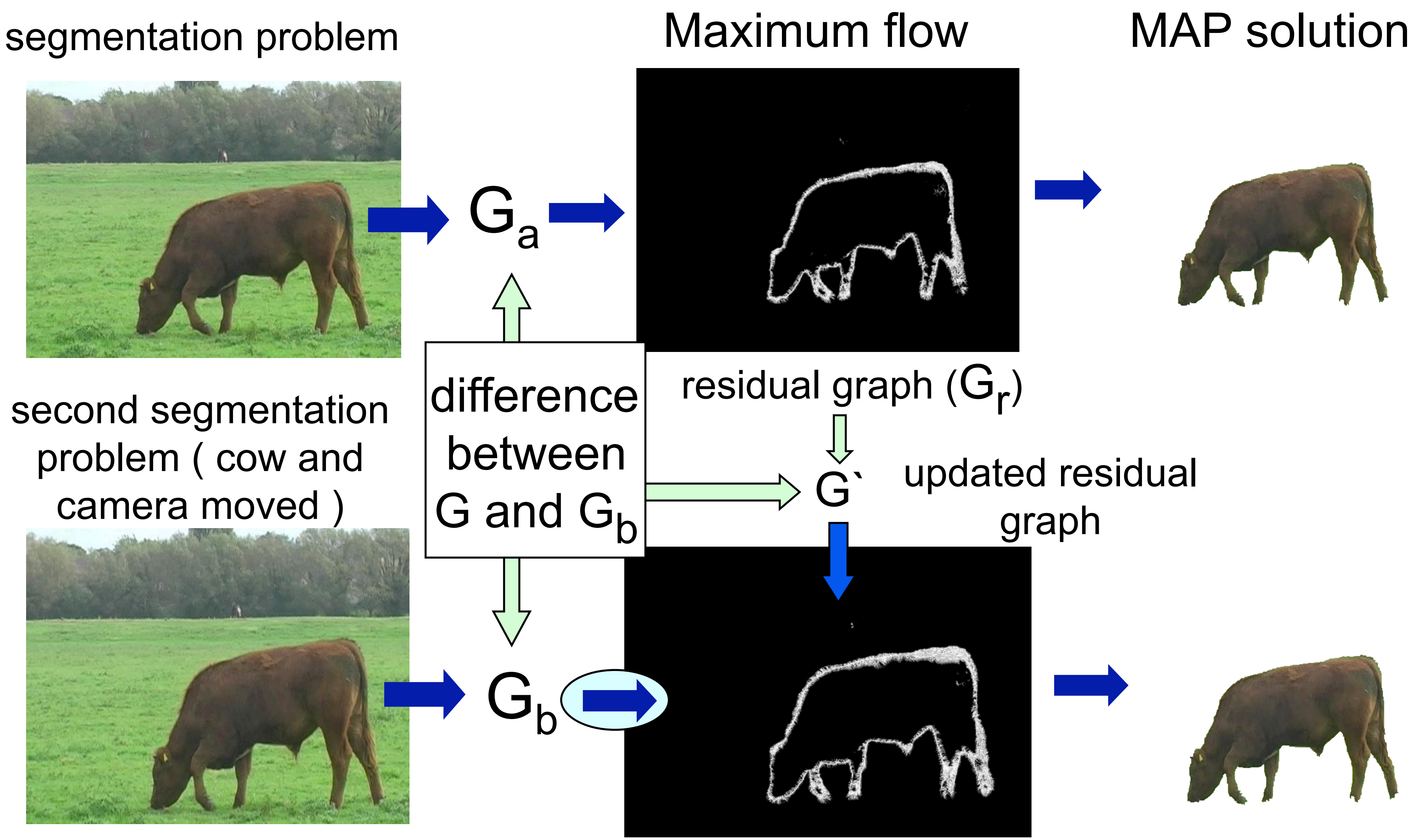
Minimizing dynamic energy functions

- Given a solution of [min E_a] compute [min E_b]
    - Compute the st-mincut on G_b using the flows in G_a
- Some flows may violate new edge capacity constraints!
    - if new edge capacities are less than the edge flow
- Re-parameterize the problem to satisfy constraints

Minimizing dynamic energy functions



Minimizing dynamic energy functions



Minimizing dynamic energy functions

Partial Solution

• Boykov and Jolly, Interactive Image Segmentation [ICCV01]

- limited to unary energy terms (t-edge capacities)

Our Contributions

• Arbitrary changes in the energy (graph)

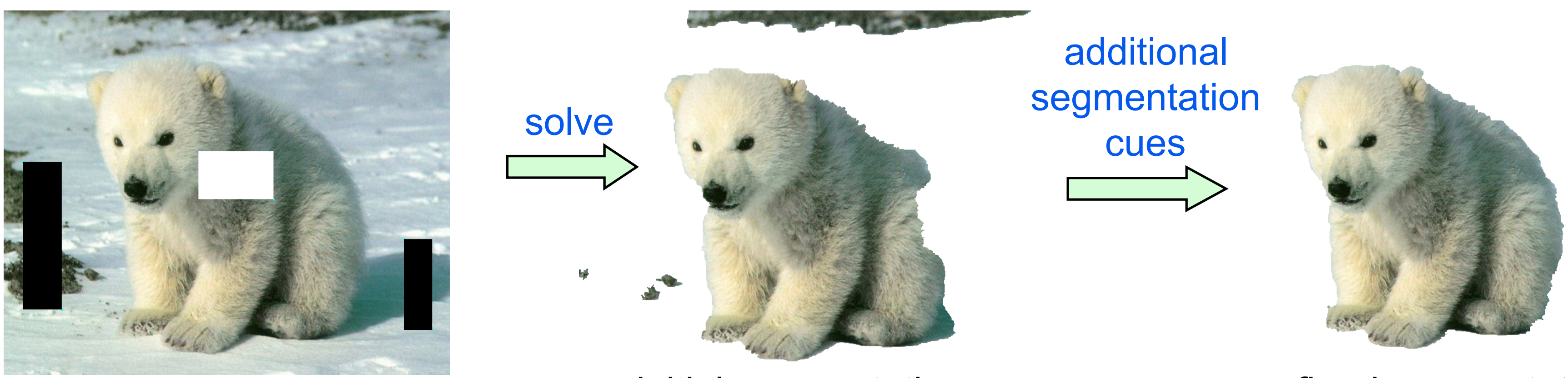
• Re-cycle search trees

- (substantial speedup observed)

Dynamic unary terms

$$\text{Energy}_{\text{MRF}} = \boxed{\sum_{i \in S} \left(\phi(\mathbf{D}|\mathbf{x}_i) \right.} + \sum_{j \in N_i} \left(\phi(\mathbf{D}|\mathbf{x}_i, \mathbf{x}_j) \; + \psi(\mathbf{x}_i, \mathbf{x}_j) \right) \Bigg)$$

- Corresponds to change in t-edge capacities
- Applications: Interactive Image Segmentation



Original Image

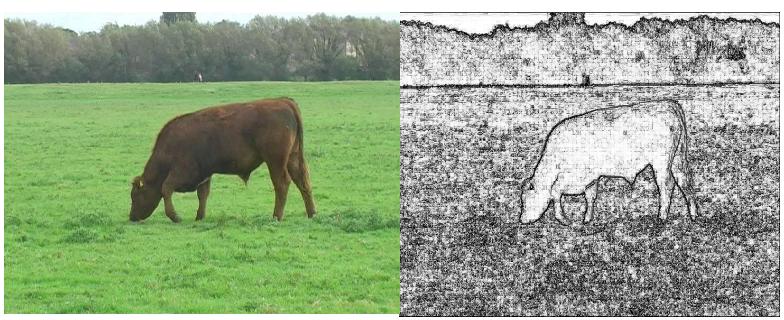
initial segmentation

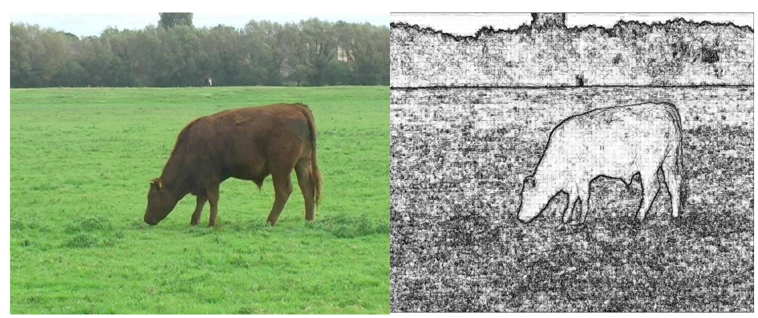
refined segmentation

Dynamic pair-wise terms

$$\text{Energy}_{\text{MRF}} = \sum_{i \in S} \left(\phi(\mathbf{D}|\mathbf{x}_i) \boxed{+ \sum_{j \in N_i} \left(\phi(\mathbf{D}|\mathbf{x}_i, \mathbf{x}_j) + \psi(\mathbf{x}_i, \mathbf{x}_j)\right)} \right)$$

- Corresponds to change in n-edge capacities
- Applications: Efficient Image Segmentation in Videos



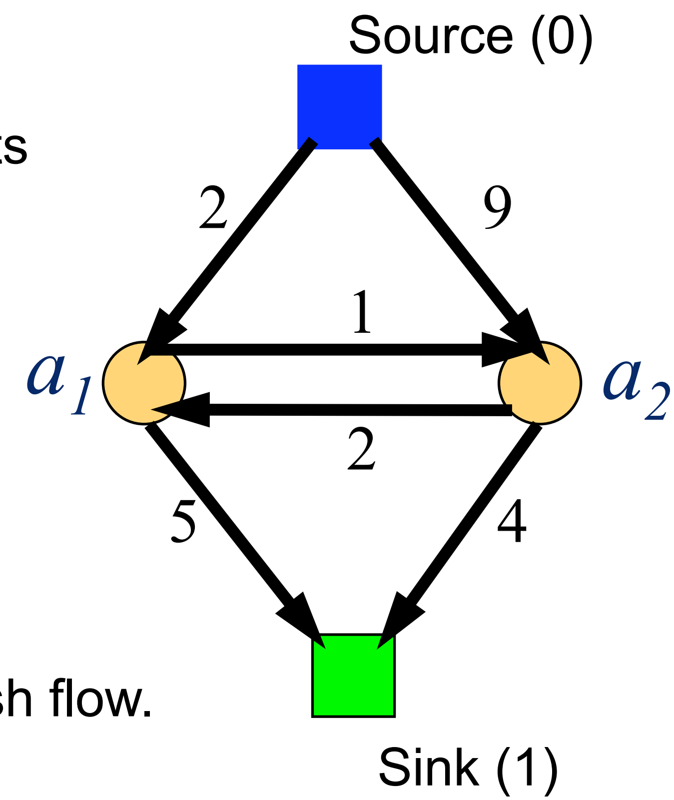


Computing the st-mincut from Max-flow algorithms

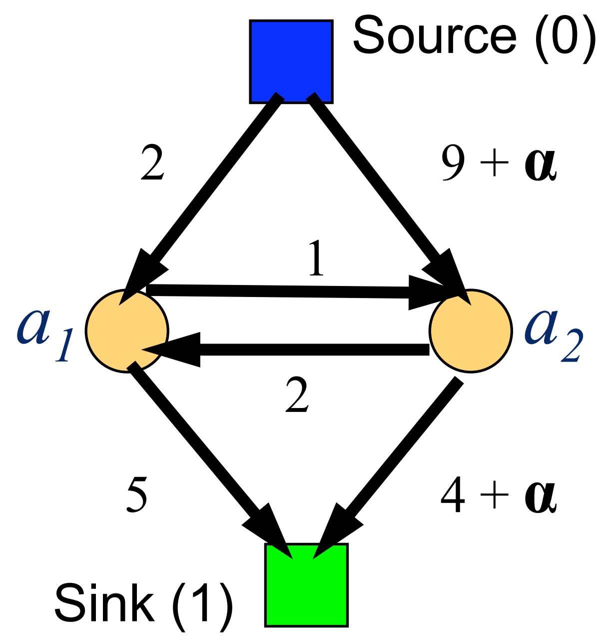
- The Max-flow Problem
  - Edge capacity and flow balance constraints

- Notation
  - Residual capacity
    (edge capacity – current flow)
  - Augmenting path

- Simple Augmenting Path based Algorithms
  - Repeatedly find augmenting paths and push flow.
  - Saturated edges constitute the st-mincut.
    [Ford-Fulkerson Theorem]



Reparametrization



Key Observation

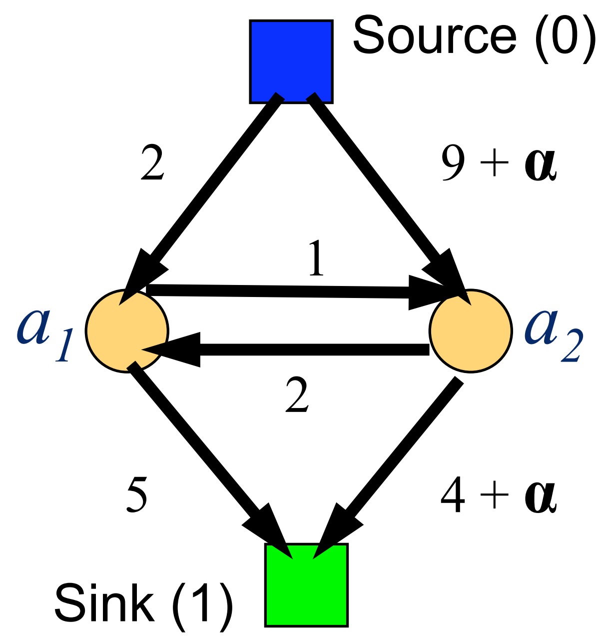
Adding a constant to both the t-edges of a node does not change the edges constituting the st-mincut.

$$E(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1 a_2$$

$$E^*(a_1,a_2) = E(a_1,a_2) + \alpha(a_2+\bar{a}_2)$$

$$= E(a_1,a_2) + \alpha \qquad [a_2+\bar{a}_2 = 1]$$

Reparametrization



Key Observation

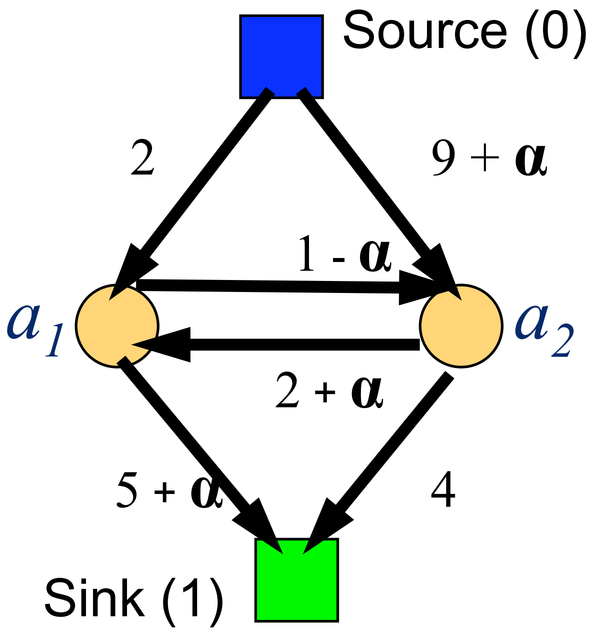
All it does is change the constant term.

$$E\,(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + 2a_1\bar{a}_2 + \bar{a}_1 a_2$$

$$E^*(a_1,a_2) = E(a_1,a_2) + \alpha(a_2+\bar{a}_2)$$

$$= E(a_1,a_2) + \alpha \qquad\qquad [a_2+\bar{a}_2 = 1]$$

Reparametrization, second type

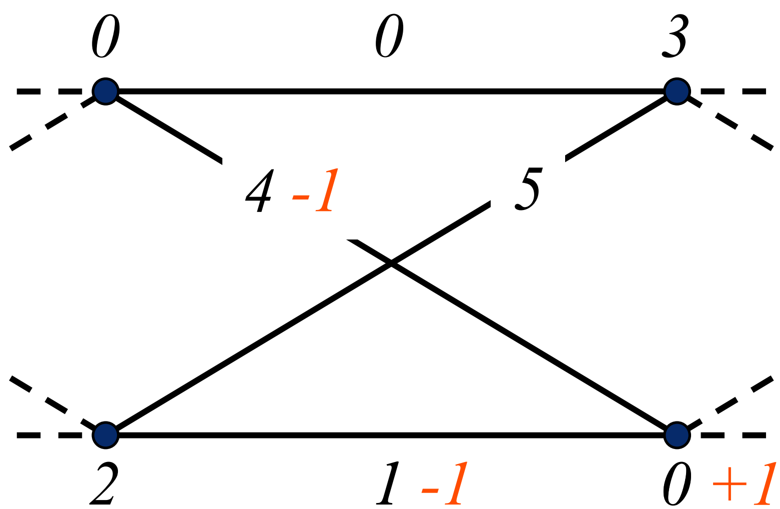


Other type of reparametrization

All reparametrizations of the graph are sums of these two types.

Both maintain the solution and add a constant **α** to the energy.

Reparameterisation



Reparametrization for shortest path 1

- **Definition.** θ' is a reparametrization of θ $(\theta' \equiv \theta)$ if they define the same energy:

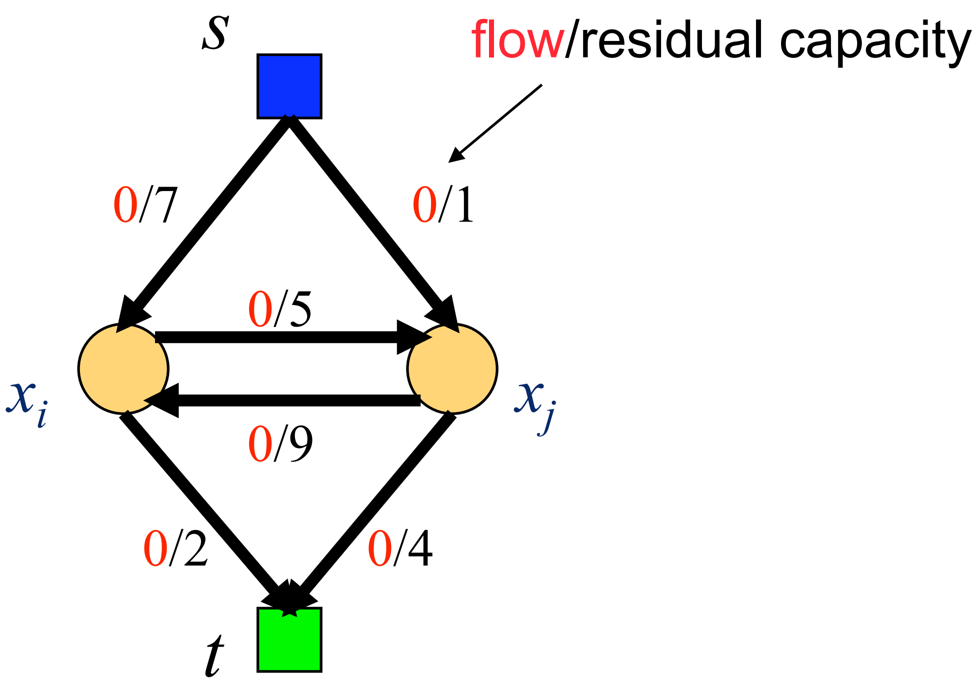
$$E(\mathbf{x} \mid \theta') = E(\mathbf{x} \mid \theta) \quad \text{for any } \mathbf{x}$$

- Maxflow, BP and TRW perform reparameterisations

Reparametrization

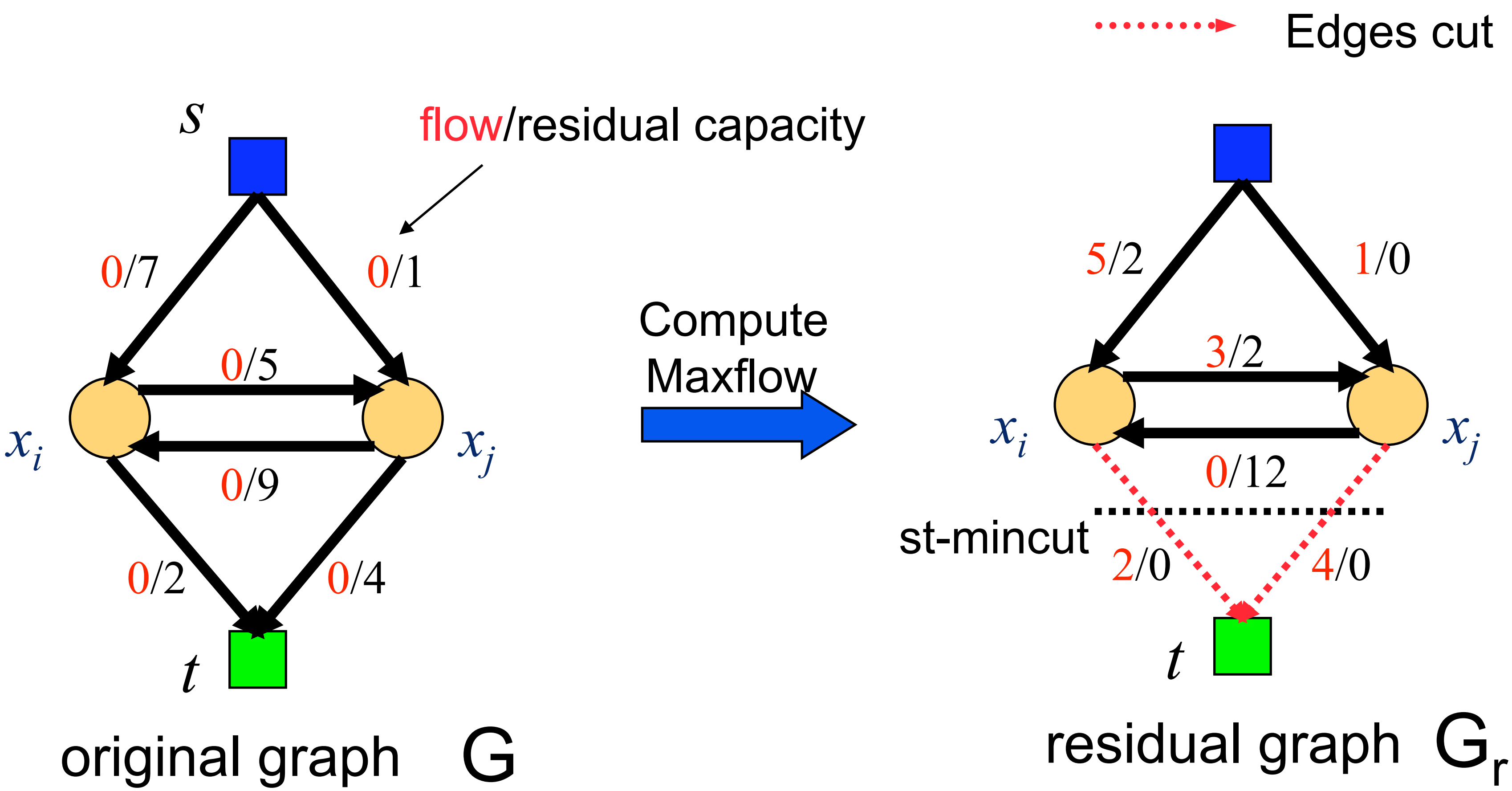
- Nice result (easy to prove)
- All other reparametrizations can be viewed in terms of these two basic operations.
- Proof in Hammer, and also in one of Vlad's recent papers.

Graph Re-parameterization

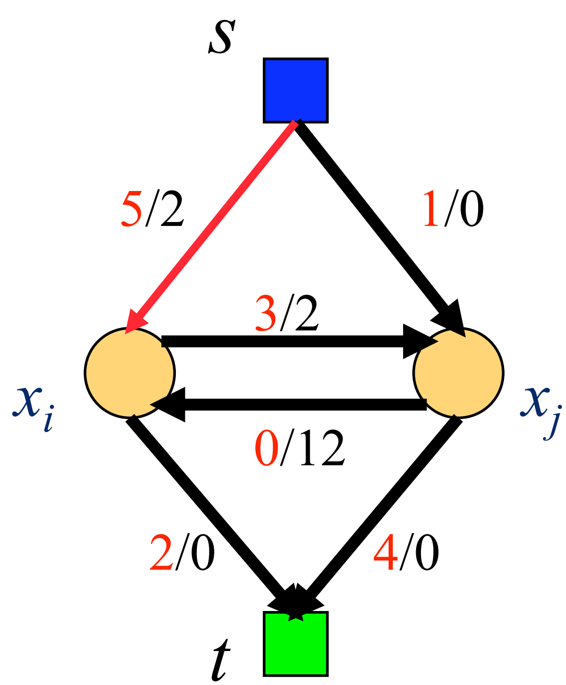


original graph G

Graph Re-parameterization



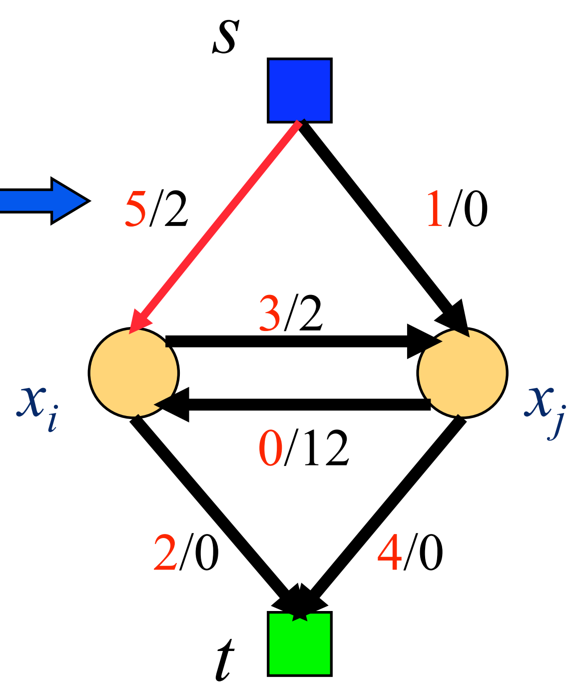
Update t-edge Capacities



residual graph G_r

Update t-edge Capacities

capacity changes from 7 to 4



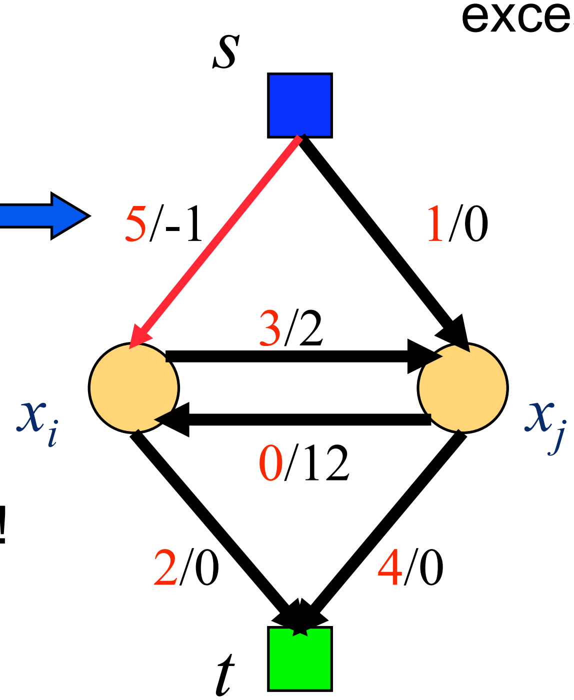
residual graph G_r

Update t-edge Capacities

excess flow (e) = flow – new capacity

= 5 – 4 = 1

capacity changes from 7 to 4



edge capacity constraint violated! (flow > capacity)

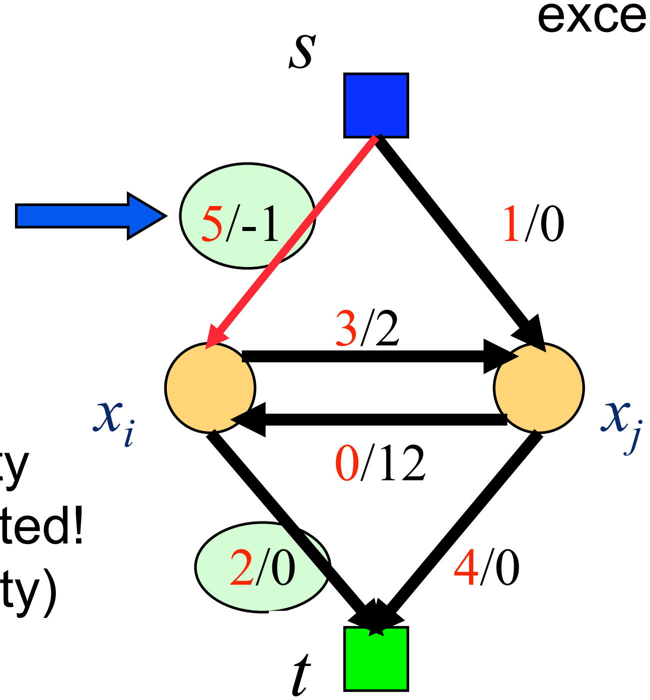
updated residual graph G`

Update t-edge Capacities

excess flow (e) = flow – new capacity

= 5 – 4 = 1

capacity changes from 7 to 4



add e to both t-edges connected to node i

edge capacity constraint violated! (flow > capacity)

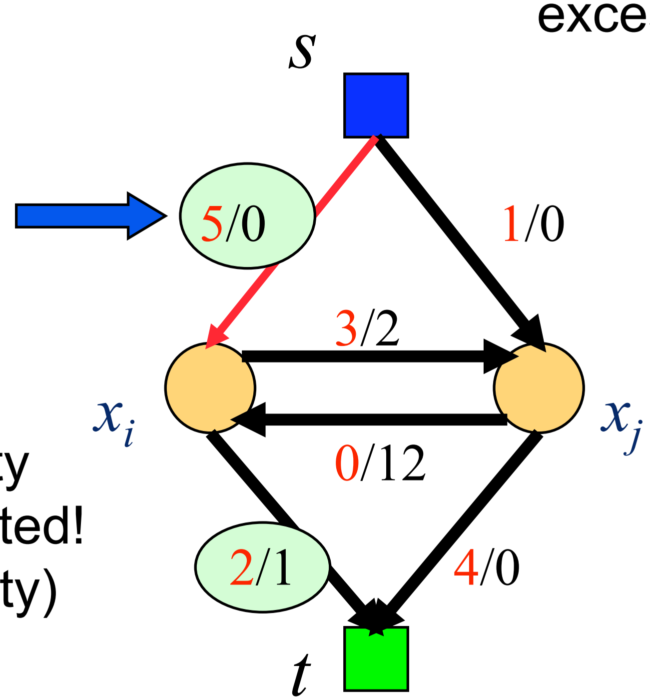
updated residual graph G`

Update t-edge Capacities

excess flow (e) = flow – new capacity

= 5 – 4 = 1

capacity
changes from
7 to 4

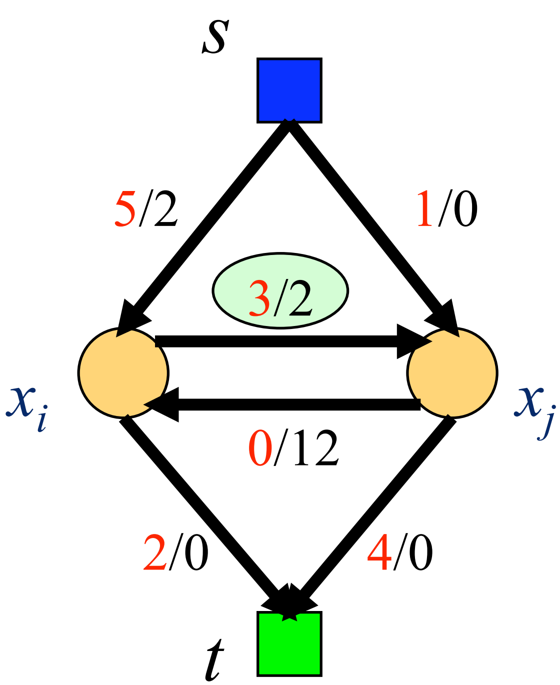


add e to both t-edges
connected to node i

edge capacity
constraint violated!
(flow > capacity)

updated residual graph G`

Update n-edge Capacities

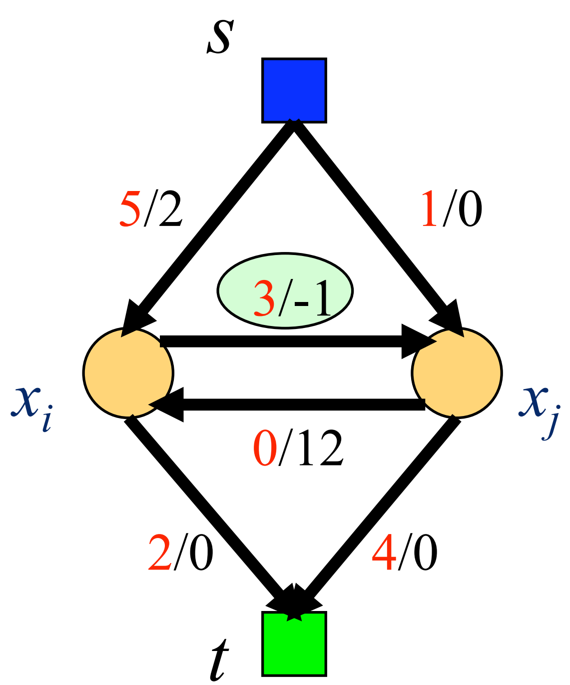


residual graph

G_r

- Capacity changes from 5 to 2

Update n-edge Capacities

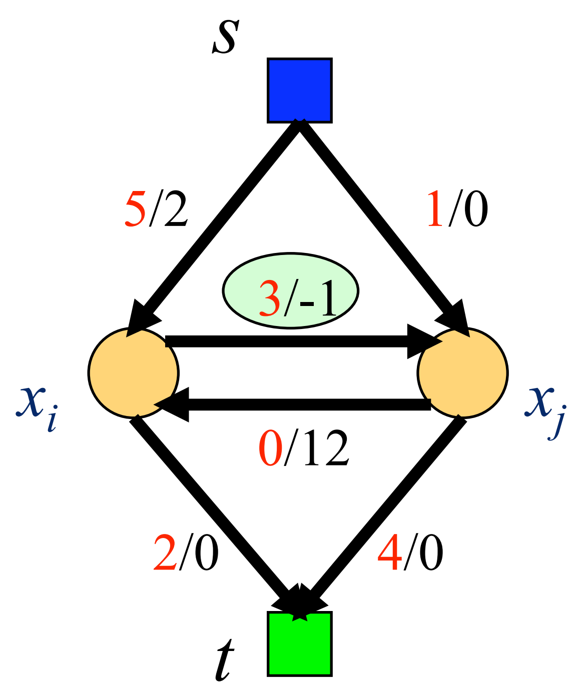


Updated residual graph

G`

- Capacity changes from 5 to 2
  - edge capacity constraint violated!

Update n-edge Capacities

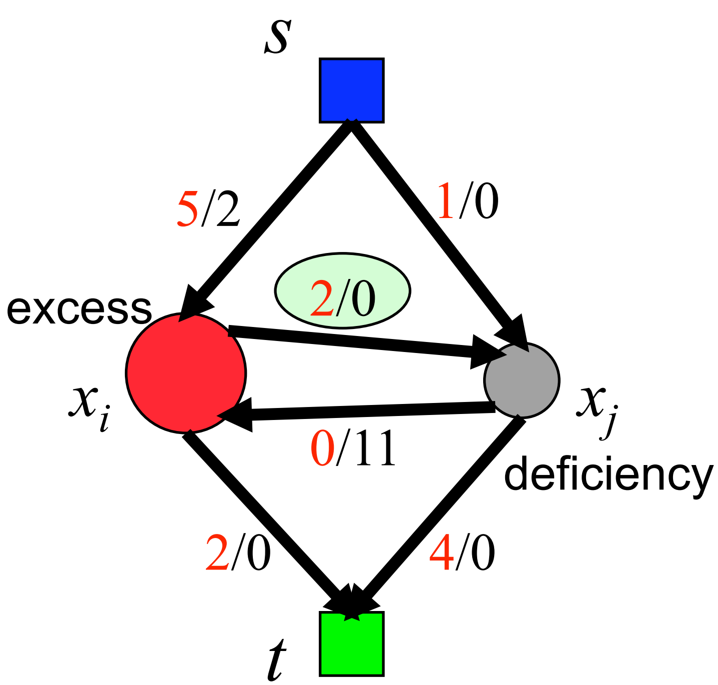


Updated residual graph

G`

- Capacity changes from 5 to 2
  - edge capacity constraint violated!
- Reduce flow to satisfy constraint

Update n-edge Capacities

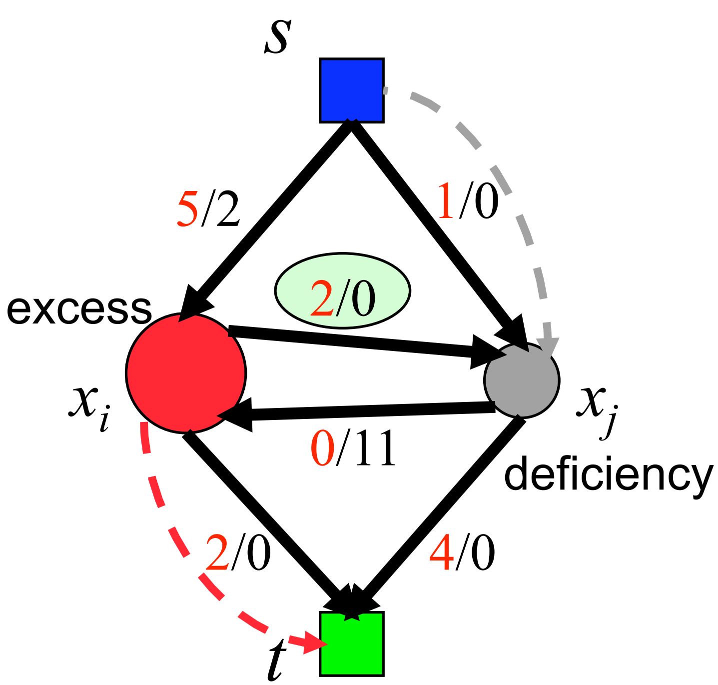


Updated residual graph

G`

- Capacity changes from 5 to 2
  - edge capacity constraint violated!
- Reduce flow to satisfy constraint
  - causes flow imbalance!

Update n-edge Capacities

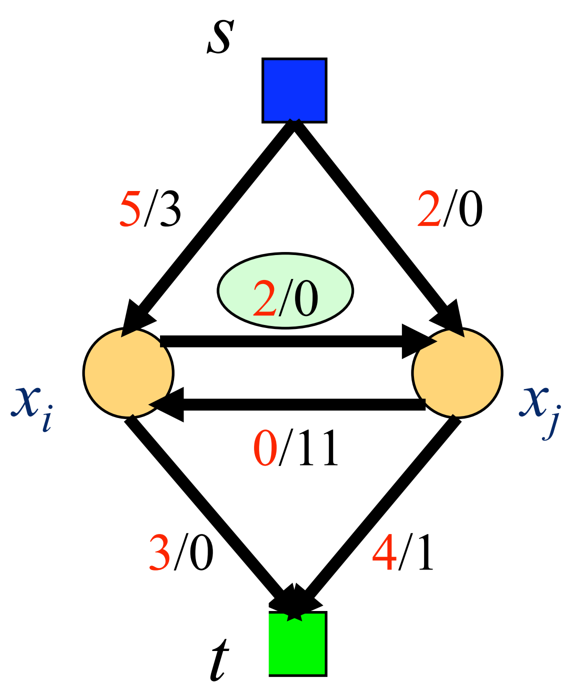


Updated residual graph

G`

- Capacity changes from 5 to 2
  - edge capacity constraint violated!
- Reduce flow to satisfy constraint
  - causes flow imbalance!
- Push excess flow to/from the terminals
- Create capacity by adding α = excess to both t-edges.

Update n-edge Capacities



Updated residual graph

G`

- Capacity changes from 5 to 2
  - edge capacity constraint violated!
- Reduce flow to satisfy constraint
  - causes flow imbalance!
- Push excess flow to the terminals
- Create capacity by adding α = excess to both t-edges.

Complexity analysis of MRF Update Operations

MRF Operation	Graph Operation	Complexity
modifying a unary term	Updating a t-edge capacity	O(1)
modifying a pair-wise term	Updating a n-edge capacity	O(1)
adding a latent variable	adding a node	O(1)
delete a latent variable	set the capacities of all edges of a node zero	O(k)*

*requires k edge update operations where k is degree of the node

Outline of the Talk

- Markov Random Fields
- Energy minimization using graph cuts
- Minimizing dynamic energy functions
- Experimental Results

Optimizing the algorithm

How to find augmenting paths?

- A trivial technique
  - Perform a breadth first search from the source to the sink.
  - Extremely computationally expensive operation
- Dual-tree maxflow algorithm [Boykov & Kolmogorov PAMI 2004]
  - Reuses search trees after each augmentation.
  - Empirically shown to be substantially faster.
- Our Idea
  - Reuse search trees from previous graph cut computation
  - Saves us search tree creation tree time [O(m)]

Reusing Search Trees

c’ = measure of change in the energy

- Algorithmic complexity:
  - Dynamic algorithm O(m + c’)
  - Optimized dynamic algorithm O(c’)
- Example:
  - Duplicate image frames (No time is needed)

Outline of the Talk

- Markov Random Fields
- Energy minimization using graph cuts
- Minimizing dynamic energy functions
- Experimental Results

Experimental Analysis

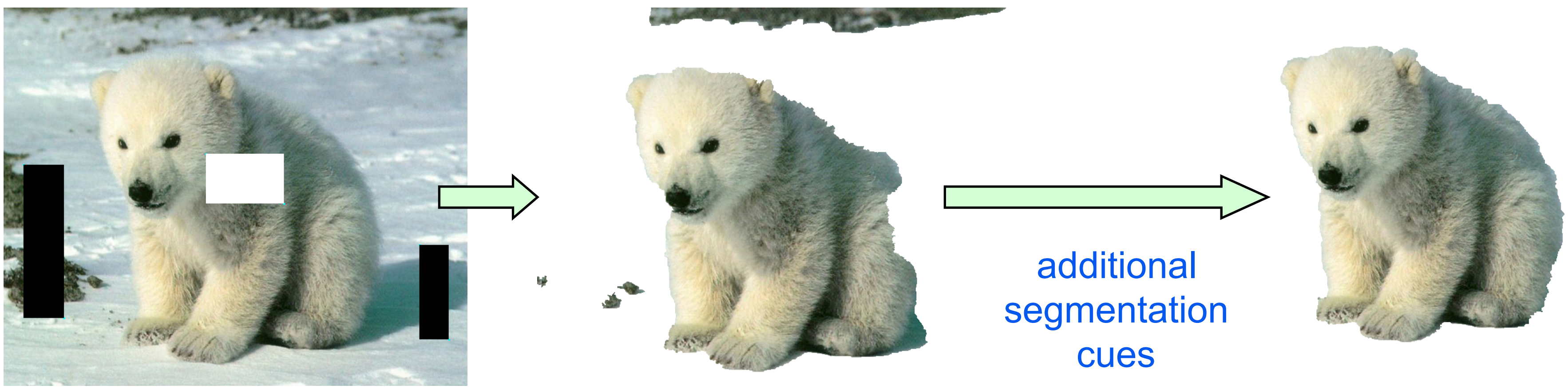
- Compared results with the best static algorithm.

- On typical video sequences a speed-up in the order of 4-5 times observed.

- Stress testing on videos with substantial jitter. (MRF changes considerably)

Experimental Analysis

Interactive Image segmentation (update unary terms)



user segmentation cues

static : 145 msec

static: 150 msec
dynamic : 60 msec
dynamic (optimized): 10 msec

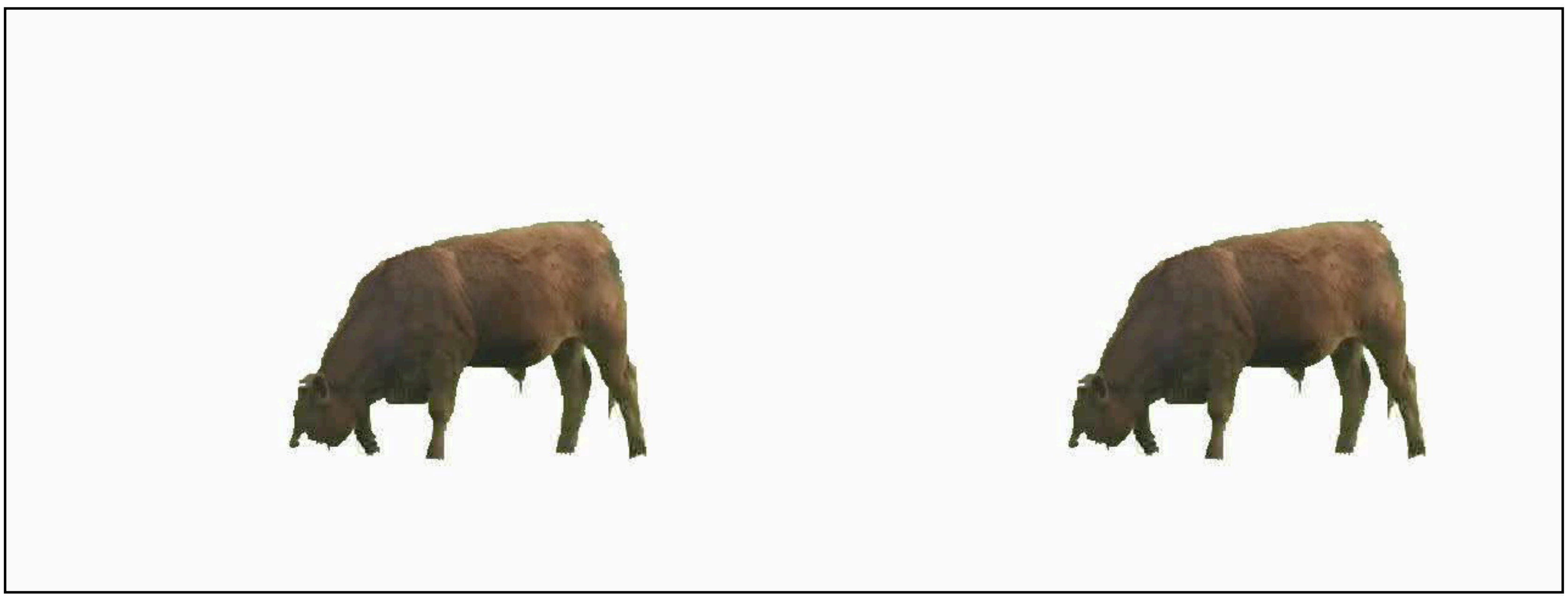
Experimental Analysis

Image segmentation in videos (unary & pairwise terms)

Image resolution: 720x576

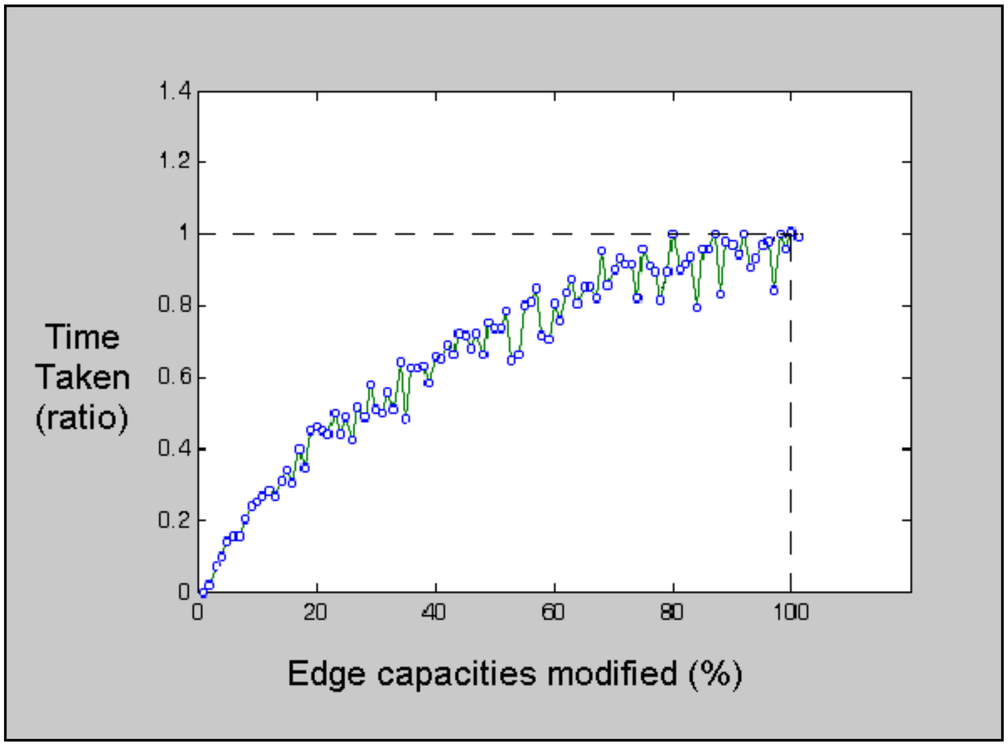
static: 190 msec
dynamic : 140 msec
dynamic (optimized): 60 msec

Dynamic Graph Cuts | Graph Cuts



Experimental Analysis

Running time of the dynamic algorithm



MRF consisting of $2x10^5$ latent variables connected in a 4-neighborhood.

Conclusions

- An exact dynamic algorithm (always gives the optimal solution)

- Sub-modular energy functions which change dynamically can be solved rapidly.

- Substantial speed-up in problems involving minimal change.

- Running time roughly proportional to the number of changes in the energy terms.

Philosophy

- Computer Vision is hard to perform fully automatically.

Philosophy

- Computer Vision is hard to perform fully automatically.
- Many things can be done with a little user input.

Philosophy

- Computer Vision is hard to perform fully automatically.
- Many things can be done with a little user input.
- But human time is a valuable resource.

Philosophy

- Computer Vision is hard to perform fully automatically.
- Many things can be done with a little user input.
- But human time is a valuable resource.
- So how can we make algorithms that give an optimal power assist to the human?

Power Assistance

- Assumption
  - We have off line time in which to do number crunching calculating useful features.
- Resulting Research Areas
  - What features do we computer offline?
  - What should the online user input be?
  - Can we develop good algorithms to combine offline information with user input?

First Example: SFM

3D model of scene

Problems

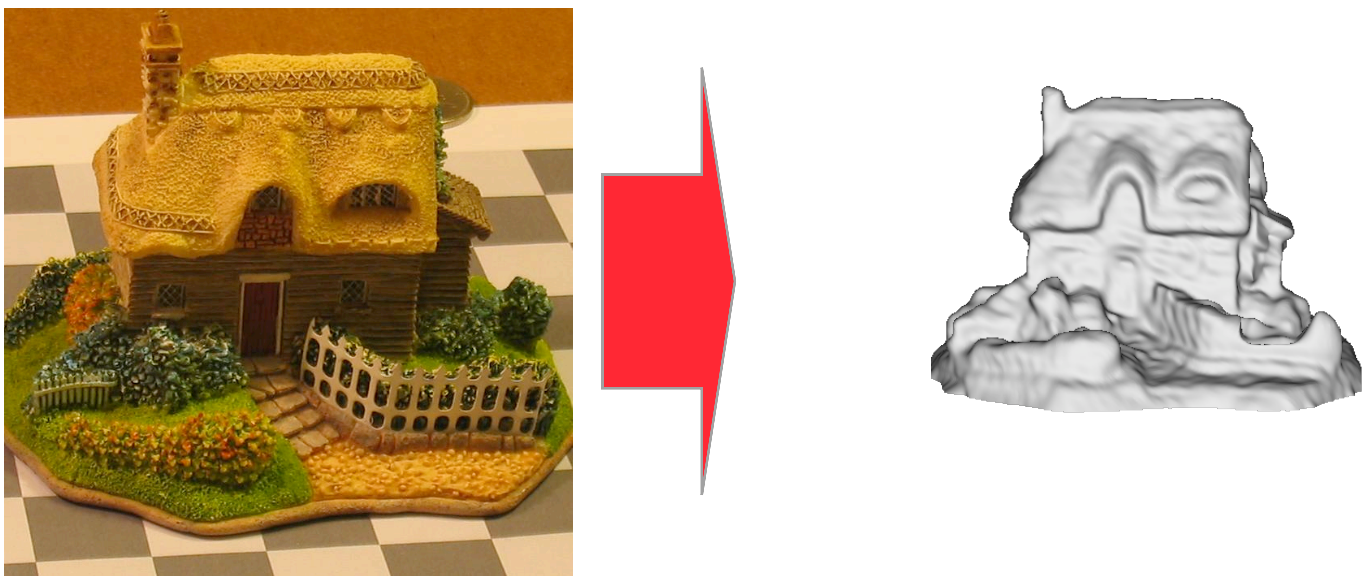
- It is not perfect apart from in ICCV papers!
  - Needs lots of views
  - Occlusions
  - Ambiguities
  - Lighting
  - Non rigidity
- Wouldn't it be great if we could easily touch up our results?

User Assist: Research Issues

- We are allowed to take the video and pre-compute all our favourite SFM primitives
  - Calibration
  - Point Tracks
  - Volumetric Stereo
- Questions
  - What are the best user edits for this?
  - What algorithms could be useful for the edits?

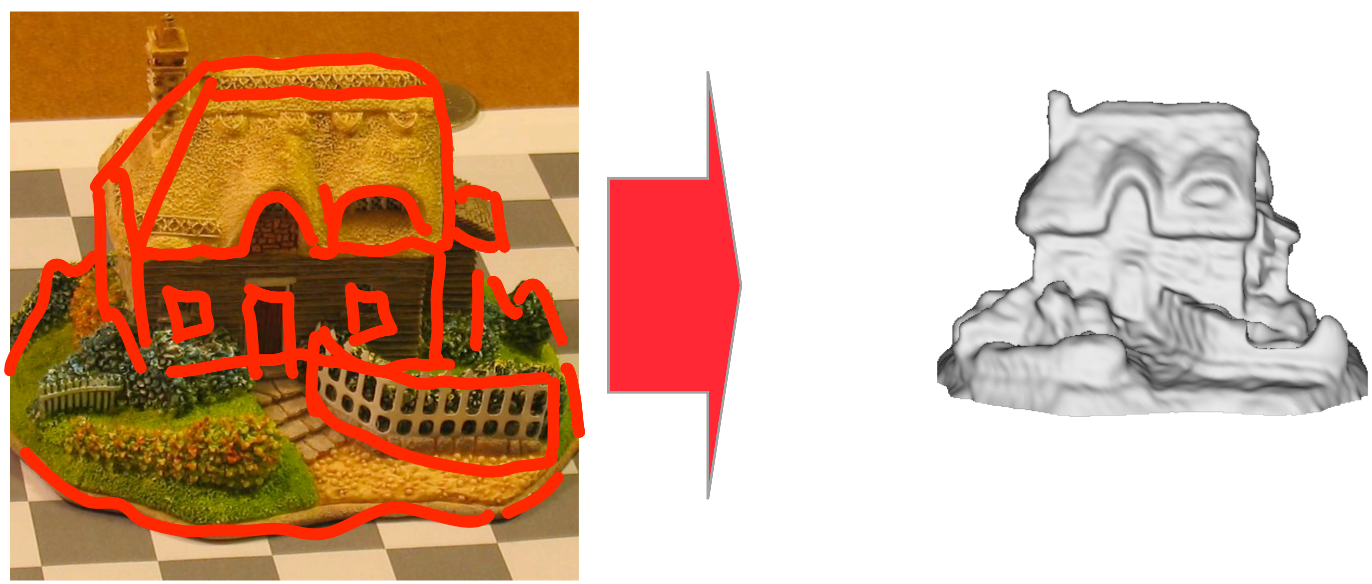
The Dream

- Suppose the user could sketch on images in the video
- and we could use his sketches to fill out and make the 3D better:



The Dream

- Suppose the user could sketch on images in the video
- and we could use his sketches to fill out and make the 3D better:



Example

Video Trace: SIGGRAPH 07

- Play Video!!!

ObjCut & PoseCut:

How to combine top down and bottom up information

How to provide power assisted segmentation

Combine theory of MRF's with Object Recognition

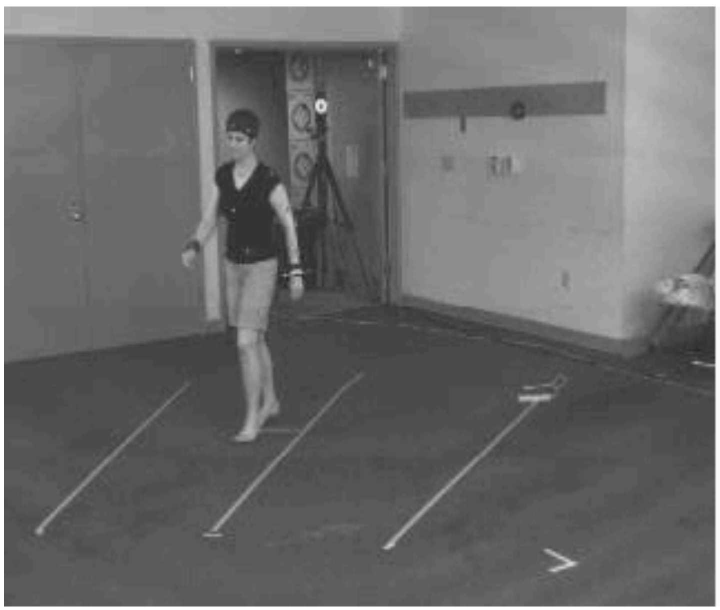
ObjCut and PoseCut:

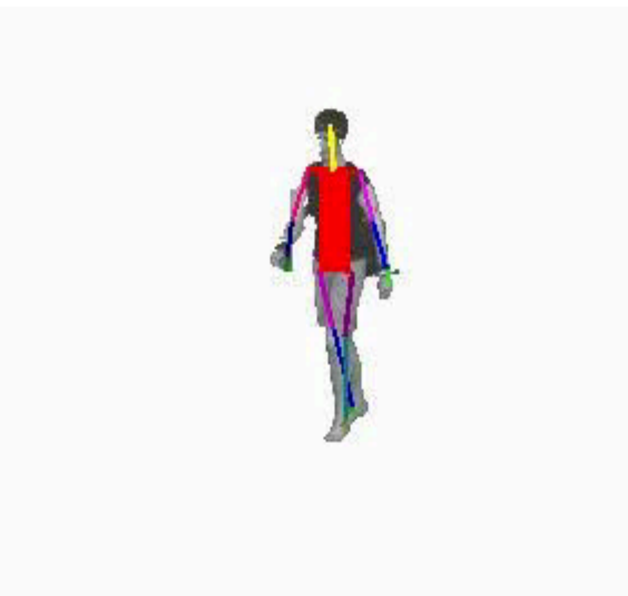
- Combination of Top Down and Bottom Up Cues..
- Classical Vision Problem.
- Combine theory of MRF’s with Object Recognition

Objective

Segmentation of Object in Video

Image | Segmentation | Pose Estimate??





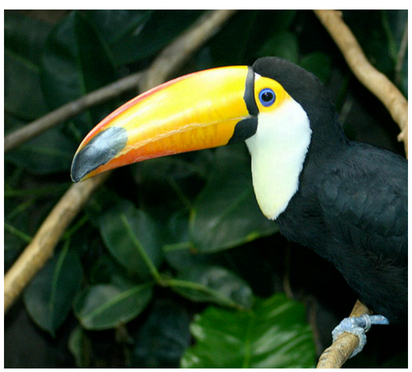
MRF for Interactive Image Segmentation,

Boykov and Jolly [ICCV 2001]

$$\text{Energy}_{\text{MRF}} = E(\mathbf{x}) = \sum_{i \in S}\left(\phi(\mathbf{D}|\mathbf{x}_i) + \sum_{j \in N_i}(\phi(\mathbf{D}|\mathbf{x}_i, \mathbf{x}_j) + \psi(\mathbf{x}_i, \mathbf{x}_j))\right) + \text{const}$$

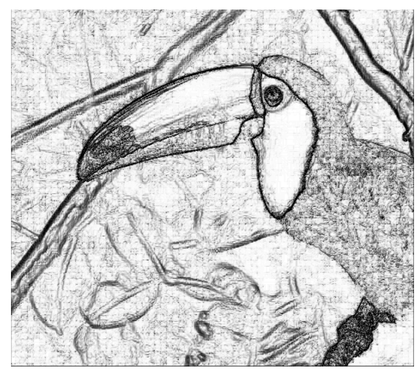
Unary likelihood — Contrast Term — Uniform Prior (Potts Model)

Maximum-a-posteriori (MAP) solution $x^* = \arg\min_{x} E(\mathbf{x})$



Data (**D**)

Unary likelihood

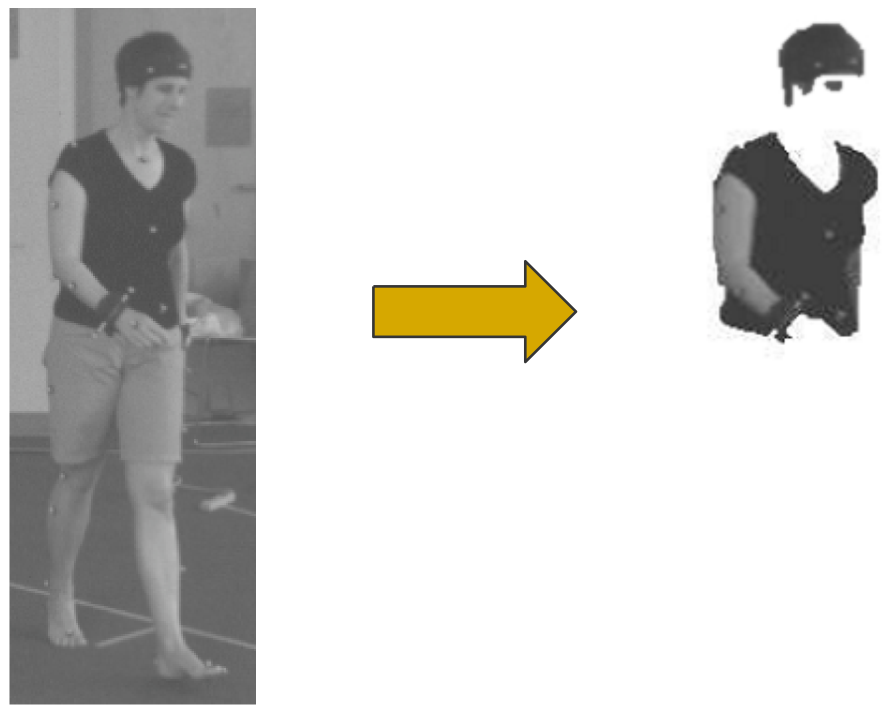


Pair-wise Terms

MAP Solution

However…

- This energy formulation rarely provides realistic (target-like) results.



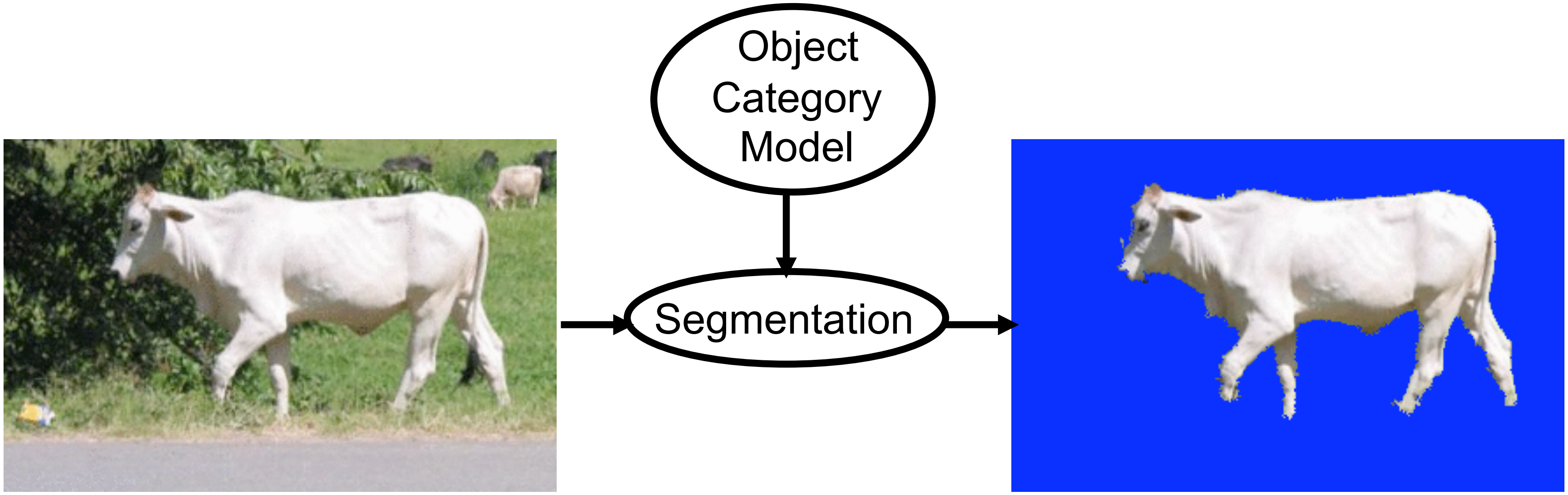
Segmentation

- To distinguish cow and horse?

First segmentation problem

Aim

■ Given an image, to segment the object



Cow Image

Segmented Cow

Segmentation should (ideally) be

- shaped like the object e.g. cow-like
- obtained efficiently in an unsupervised manner
- able to handle self-occlusion

Challenges

Intra-Class Shape Variability



Intra-Class Appearance Variability



Self Occlusion

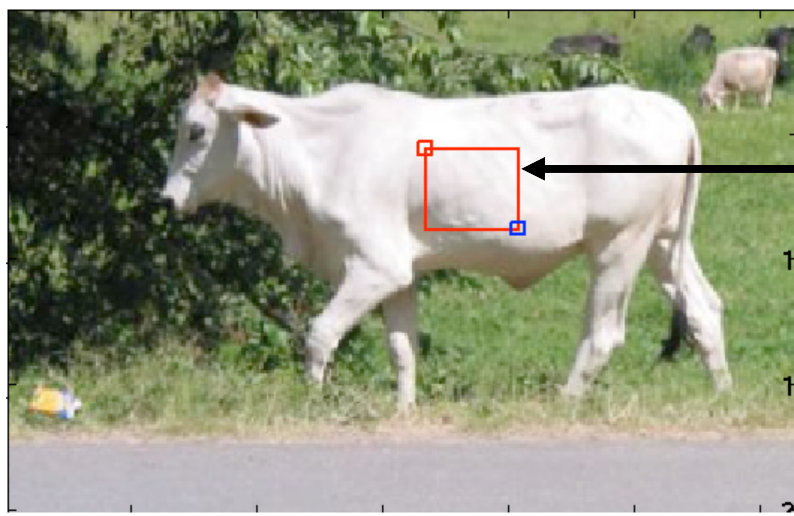


Motivation

Magic Wand

Current methods require user intervention

- Object and background seed pixels (Boykov and Jolly, ICCV 01)
- Bounding Box of object (Rother *et al.* SIGGRAPH 04)



Object Seed Pixels

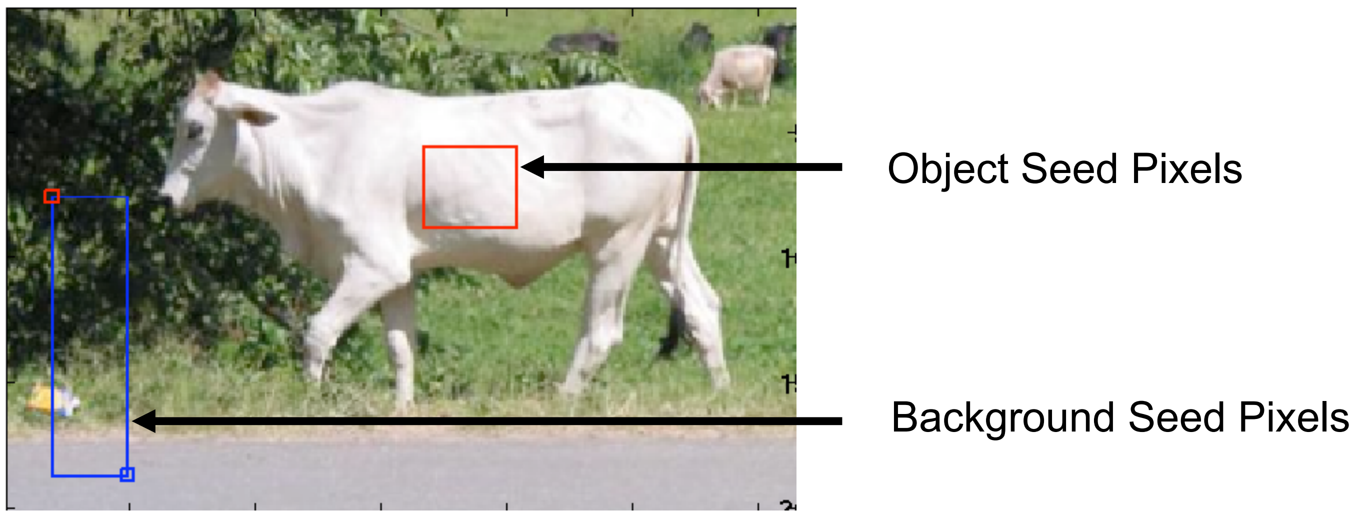
Cow Image

Magic Wand

Motivation

Current methods require user intervention

- Object and background seed pixels (Boykov and Jolly, ICCV 01)
- Bounding Box of object (Rother *et al.* SIGGRAPH 04)



Cow Image

Motivation

Magic Wand

Current methods require user intervention

- Object and background seed pixels (Boykov and Jolly, ICCV 01)
- Bounding Box of object (Rother *et al.* SIGGRAPH 04)

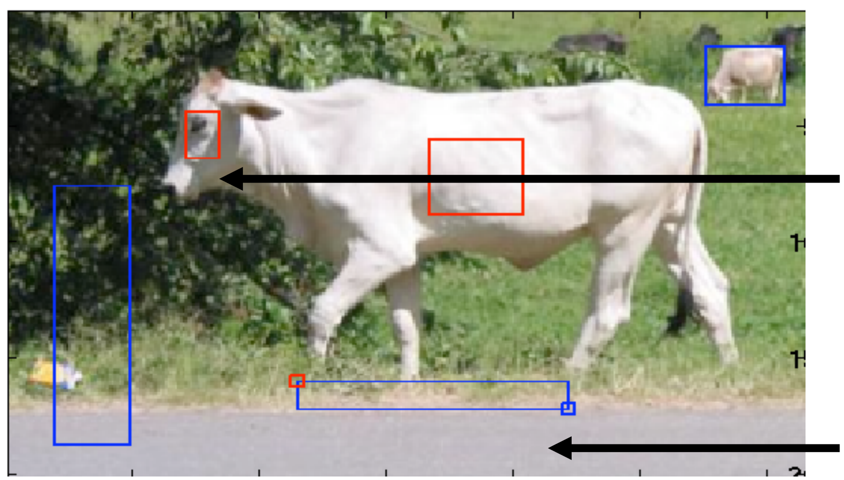
Segmented Image

Motivation

Magic Wand

Current methods require user intervention

- Object and background seed pixels (Boykov and Jolly, ICCV 01)
- Bounding Box of object (Rother *et al.* SIGGRAPH 04)



Object Seed Pixels

Background Seed Pixels

Cow Image

Motivation

Magic Wand

Current methods require user intervention

- Object and background seed pixels (Boykov and Jolly, ICCV 01)
- Bounding Box of object (Rother *et al.* SIGGRAPH 04)

Segmented Image

Motivation

Problem

- Manually intensive
- Segmentation is not guaranteed to be ‘object-like’

Non Object-like Segmentation

Our Method

- Combine object detection with segmentation
  - ***Borenstein and Ullman, ECCV '02***
  - Leibe and Schiele, BMVC '03

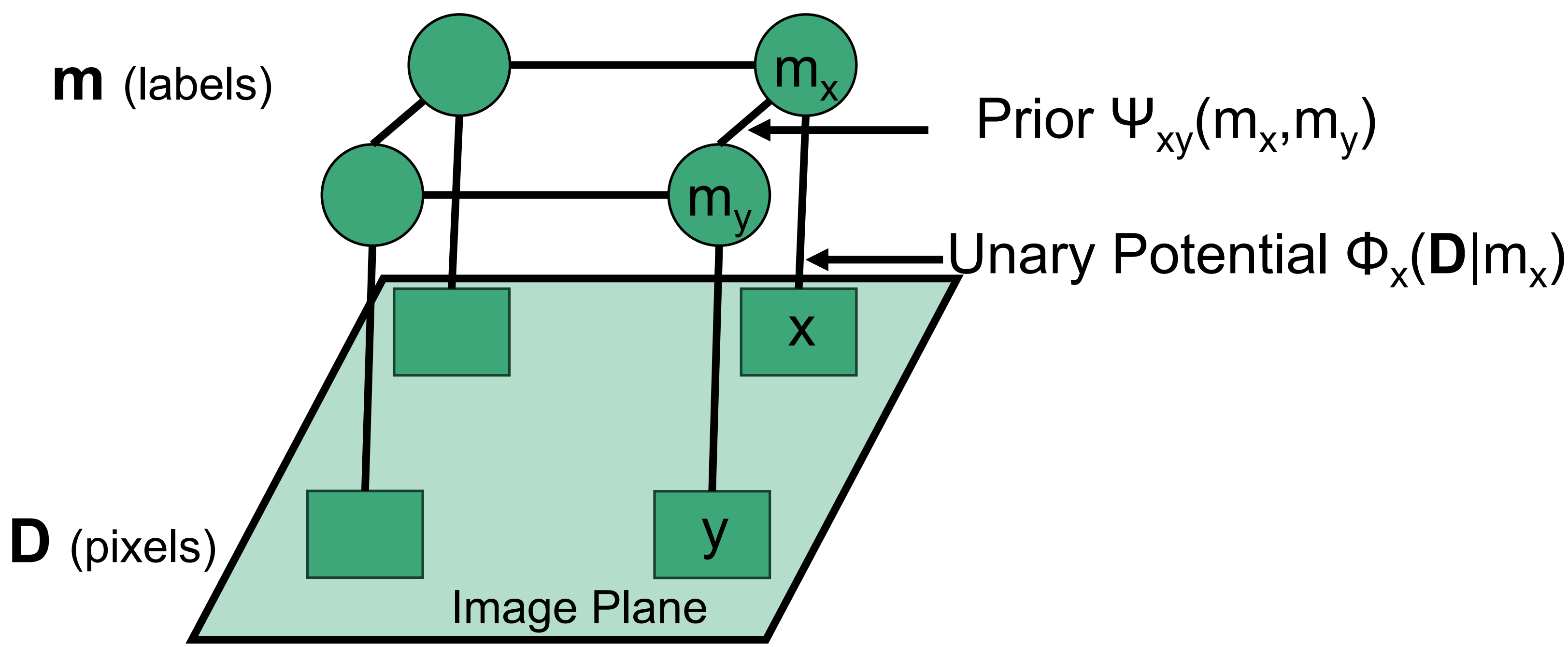
- Incorporate global shape priors in MRF

- Detection provides
  - Object Localization
  - Global shape priors

- Automatically segments the object
  - Note our method completely generic
  - Applicable to any object category model

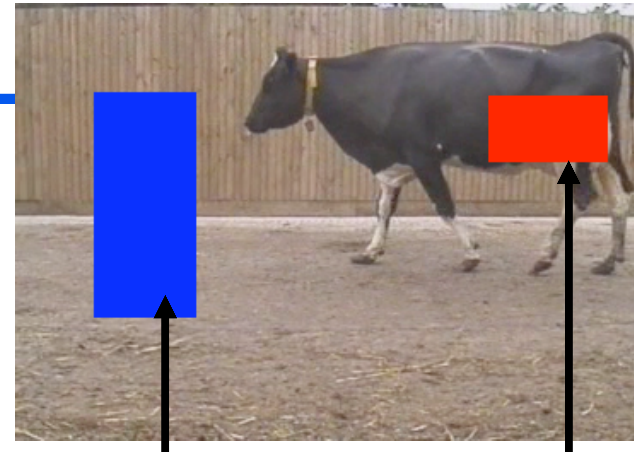
MRF

Probability for a labelling consists of
- Likelihood
  - Unary potential based on colour of pixel
- Prior which favours same labels for neighbours (pairwise potentials)



Example

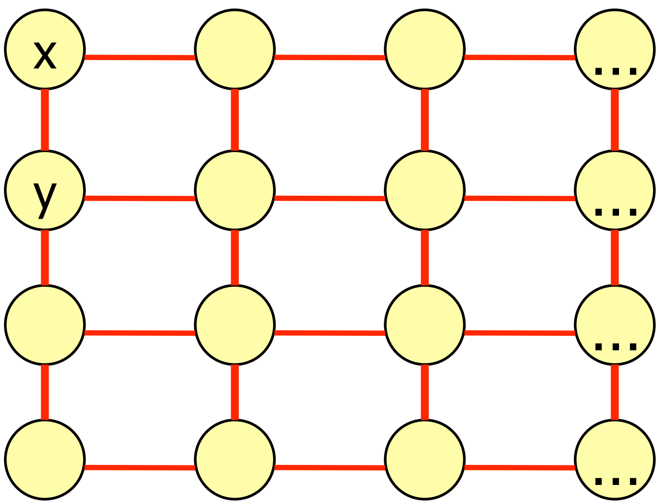
Cow Image



Background Seed Pixels

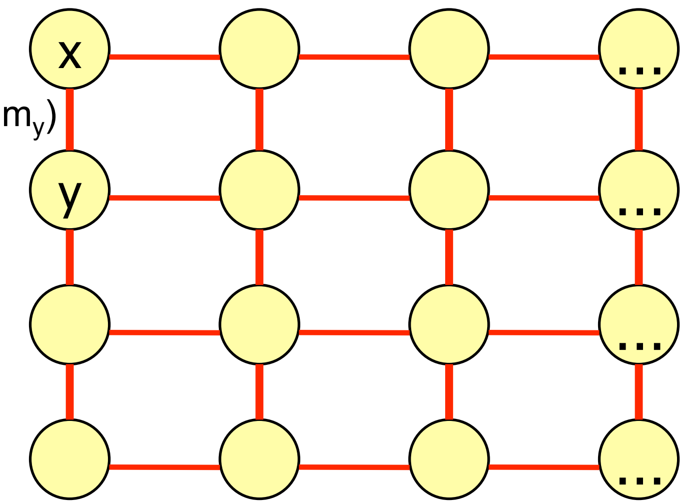
Object Seed Pixels

$$\frac{\Phi_x(\mathbf{D}|obj)}{\Phi_x(\mathbf{D}|bkg)}$$



Likelihood Ratio (Colour)

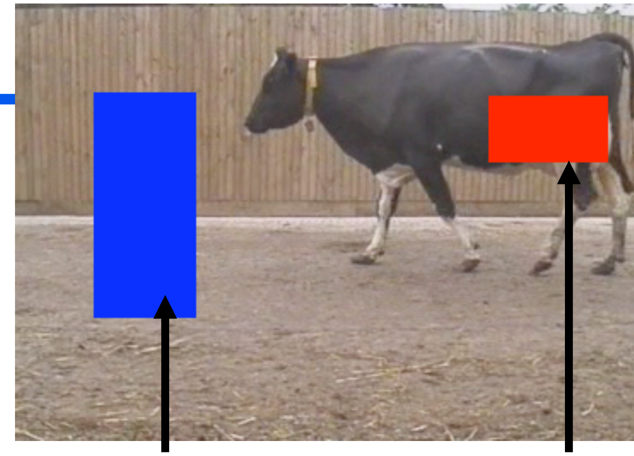
$\Psi_{xy}(m_x,m_y)$



Prior

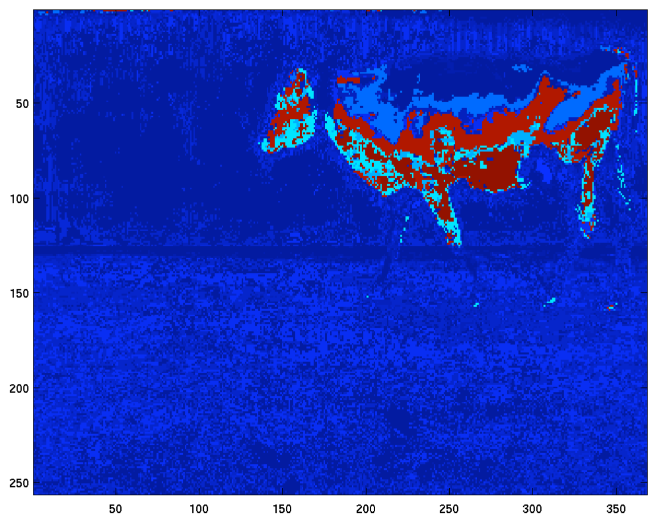
Example

Cow Image

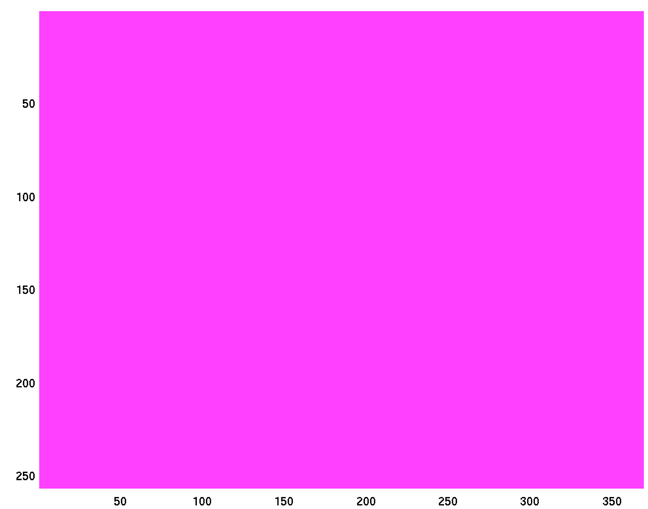


Background Seed Pixels

Object Seed Pixels



Likelihood Ratio (Colour)

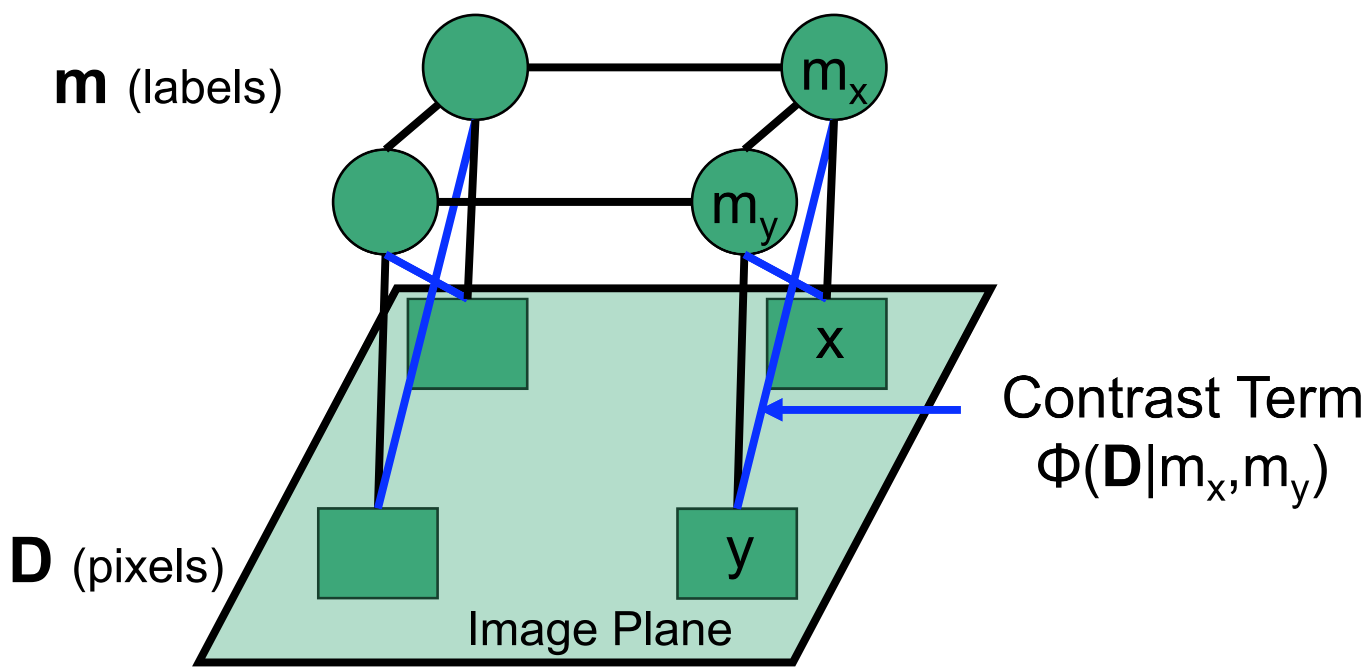


Prior

Contrast-Dependent MRF

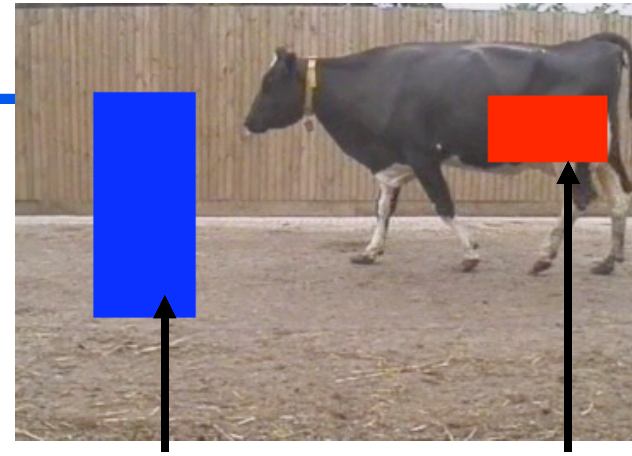
Probability of labelling in addition has

- Contrast term which favours boundaries to lie on image edges



Example

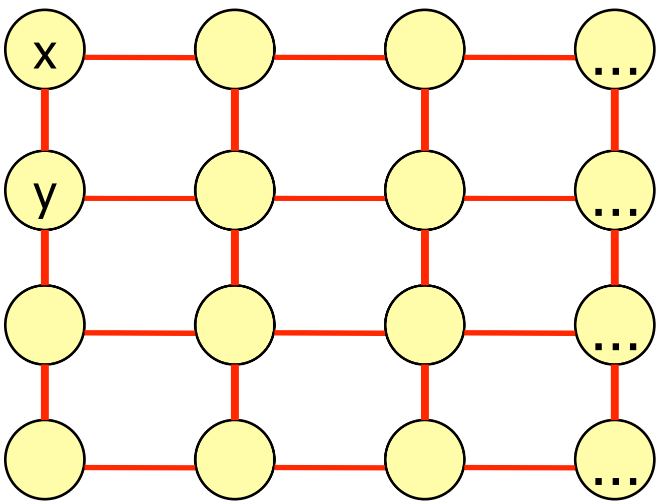
Cow Image



Background Seed Pixels

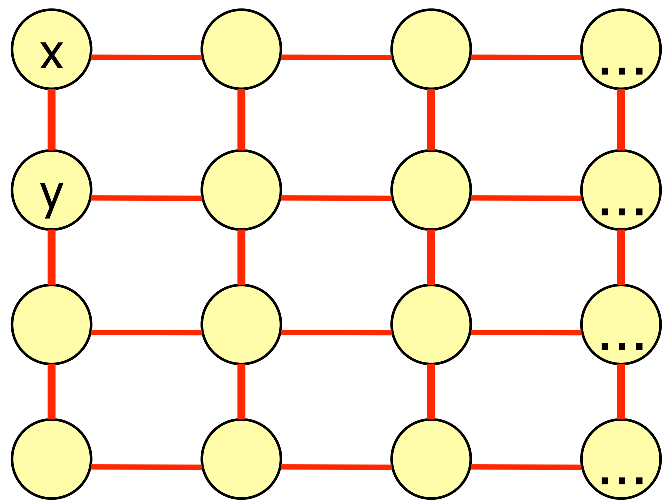
Object Seed Pixels

$$\frac{\Phi_x(\mathbf{D}|\text{obj})}{\Phi_x(\mathbf{D}|\text{bkg})}$$



Likelihood Ratio (Colour)

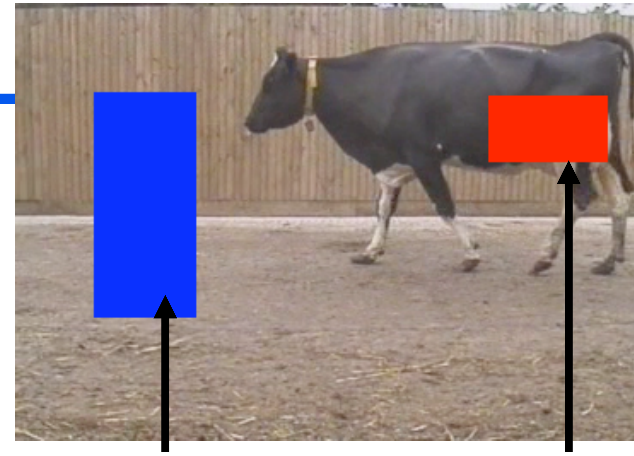
$\Psi_{xy}(m_x,m_y)$
+
$\Phi(\mathbf{D}|m_x,m_y)$



Prior + Contrast

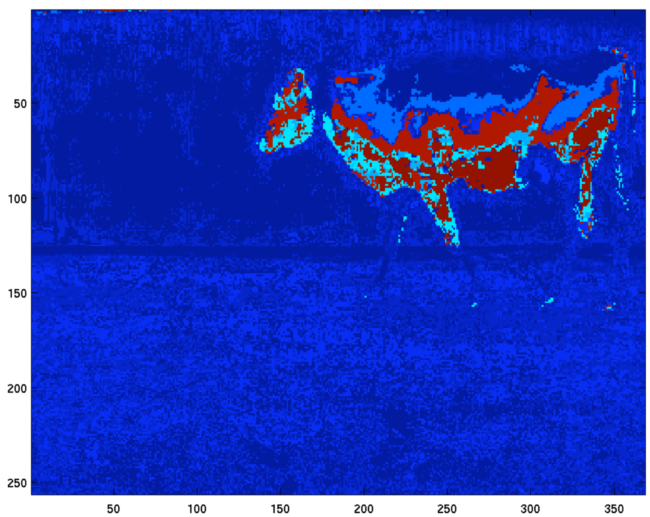
Example

Cow Image

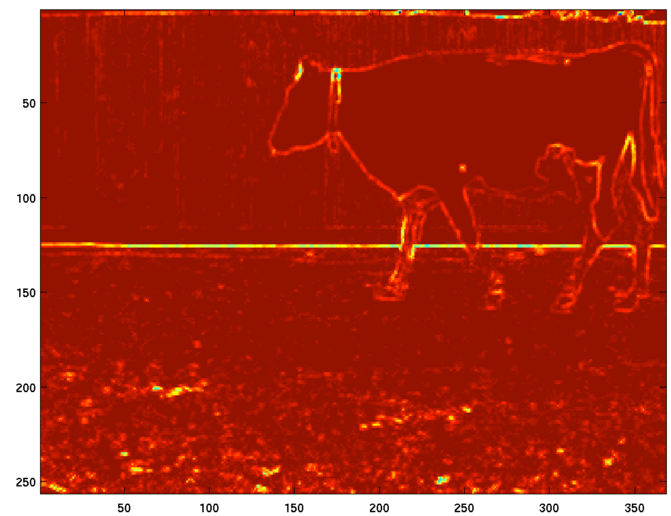


Background Seed Pixels

Object Seed Pixels

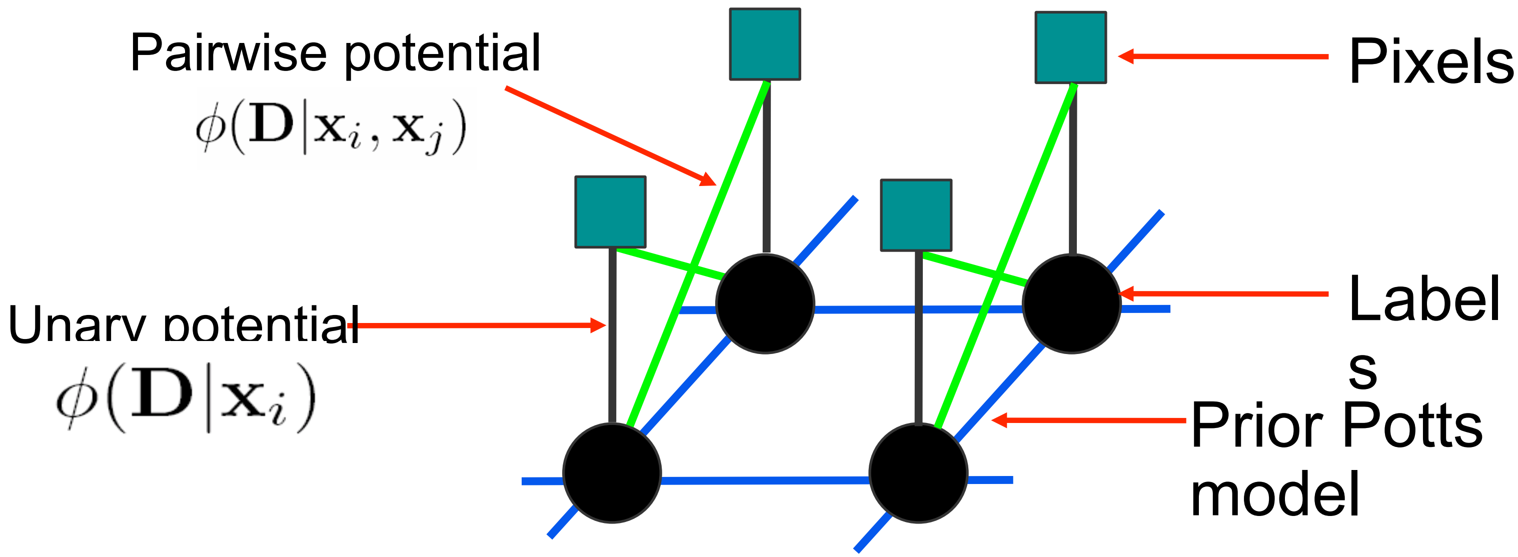


Likelihood Ratio (Colour)



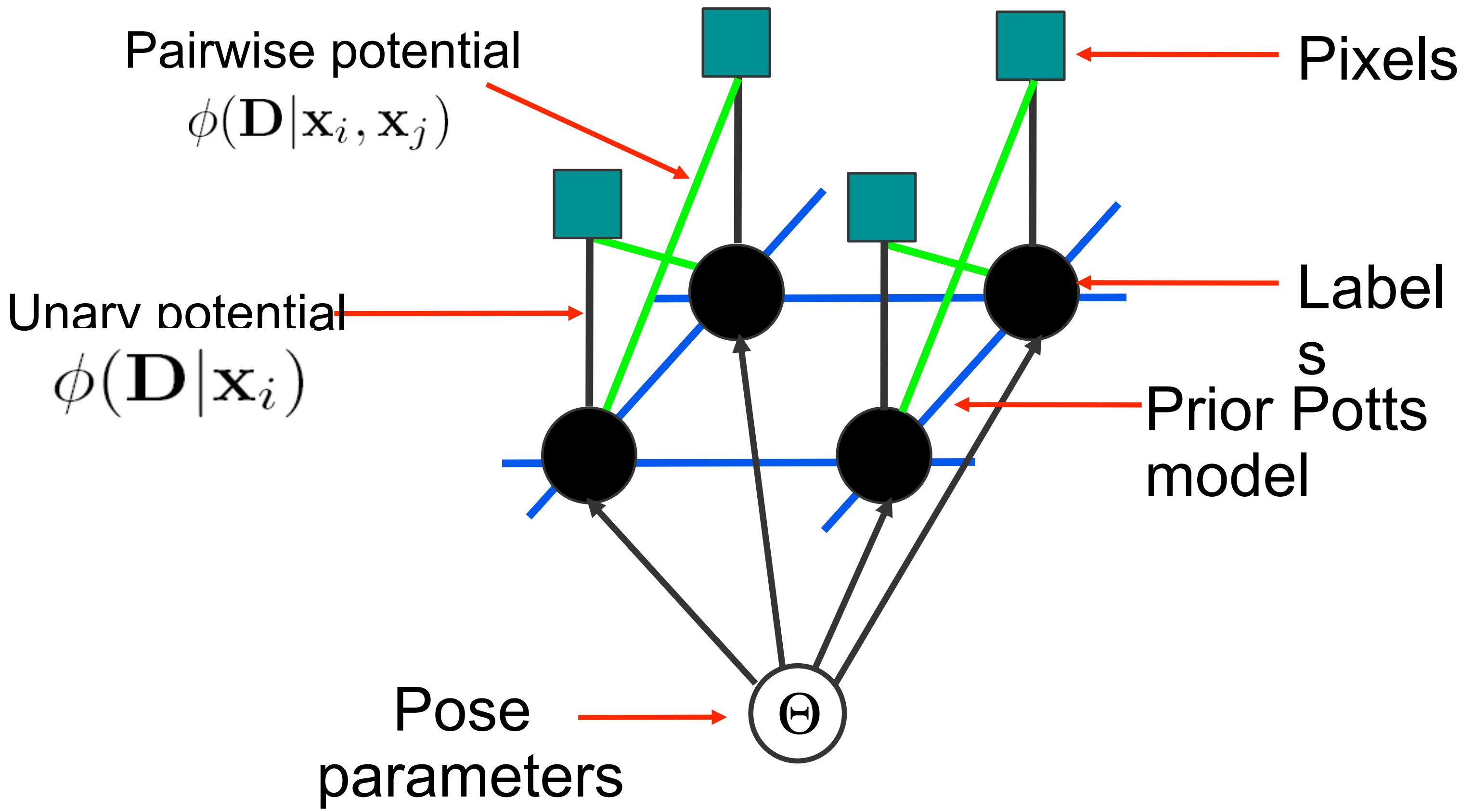
Prior + Contrast

Integrating Shape-Prior in MRFs



MRF for segmentation

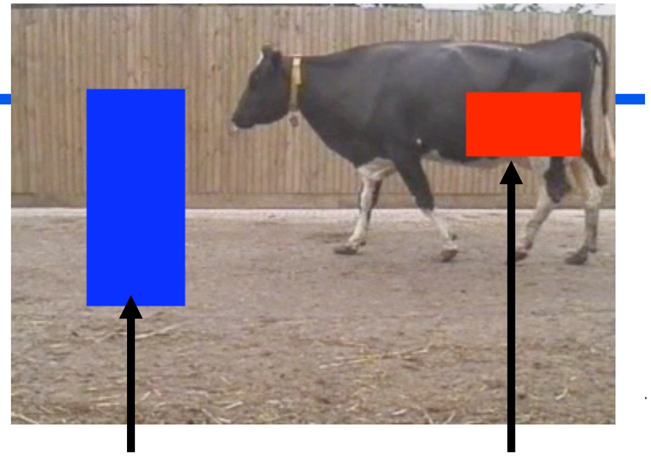
Integrating Shape-Prior in MRFs



Pose-specific MRF

Example

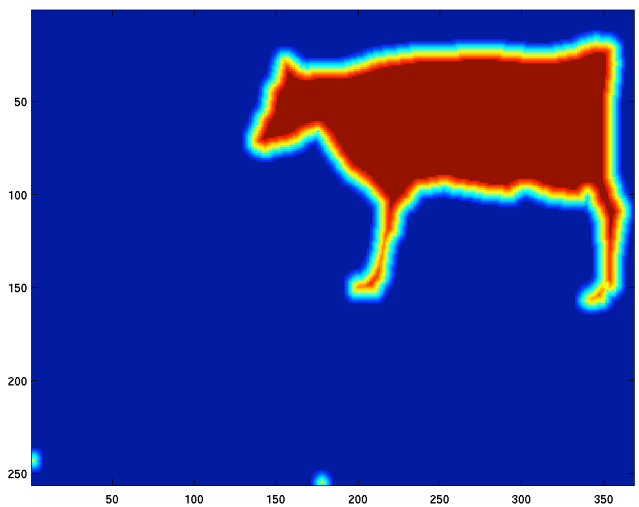
Cow Image



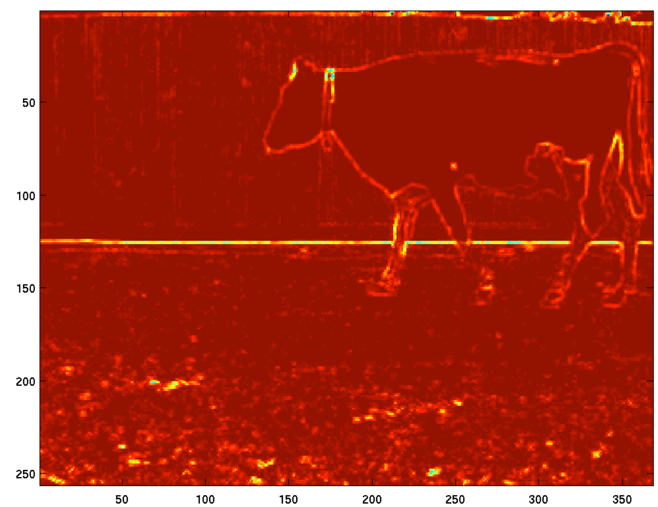
Background Seed Pixels

Object Seed Pixels

Shape Prior Θ



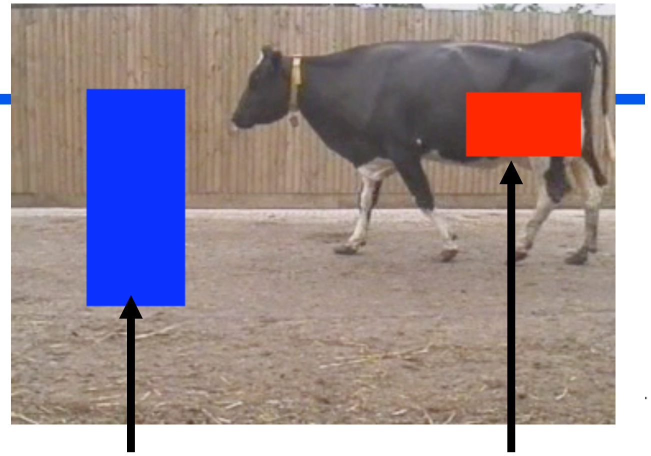
Distance from Θ



Prior + Contrast

Example

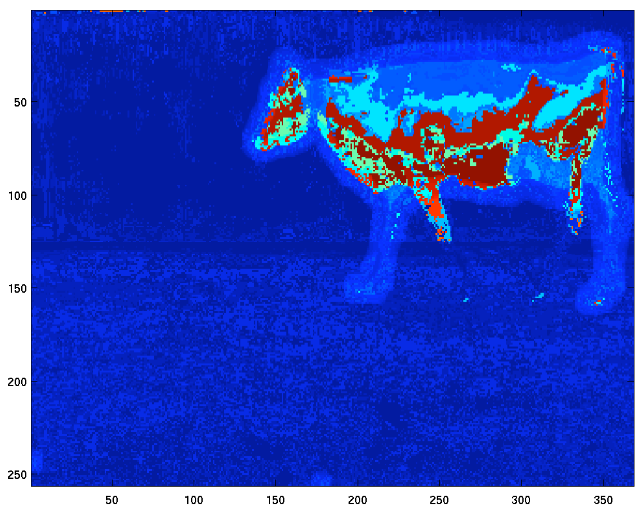
Cow Image



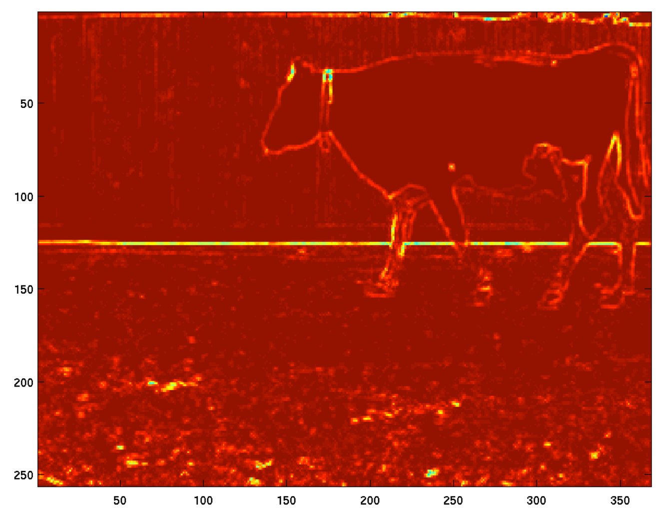
Background Seed Pixels

Object Seed Pixels

Shape Prior Θ



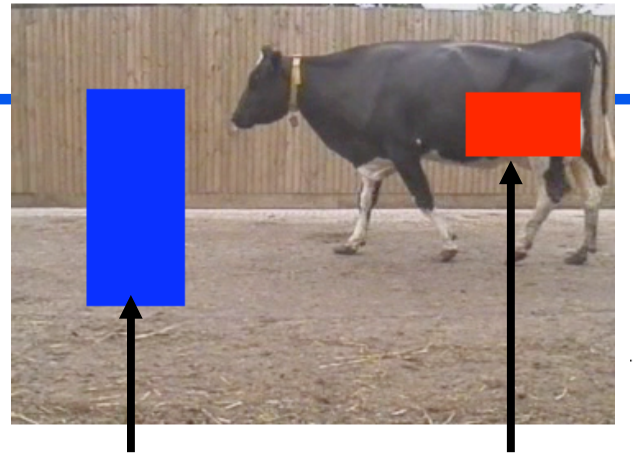
Likelihood + Distance from Θ



Prior + Contrast

Example

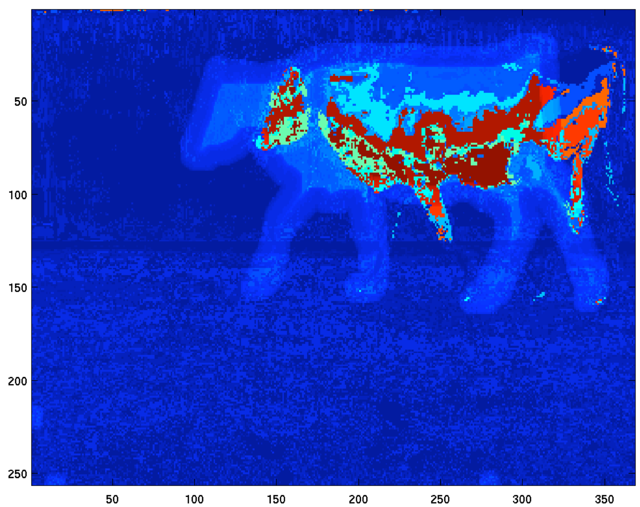
Cow Image



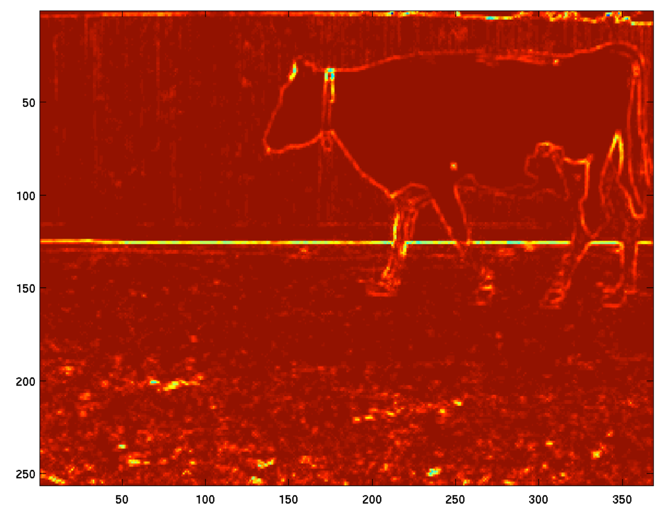
Background Seed Pixels

Object Seed Pixels

Shape Prior Θ



Likelihood + Distance from Θ



Prior + Contrast

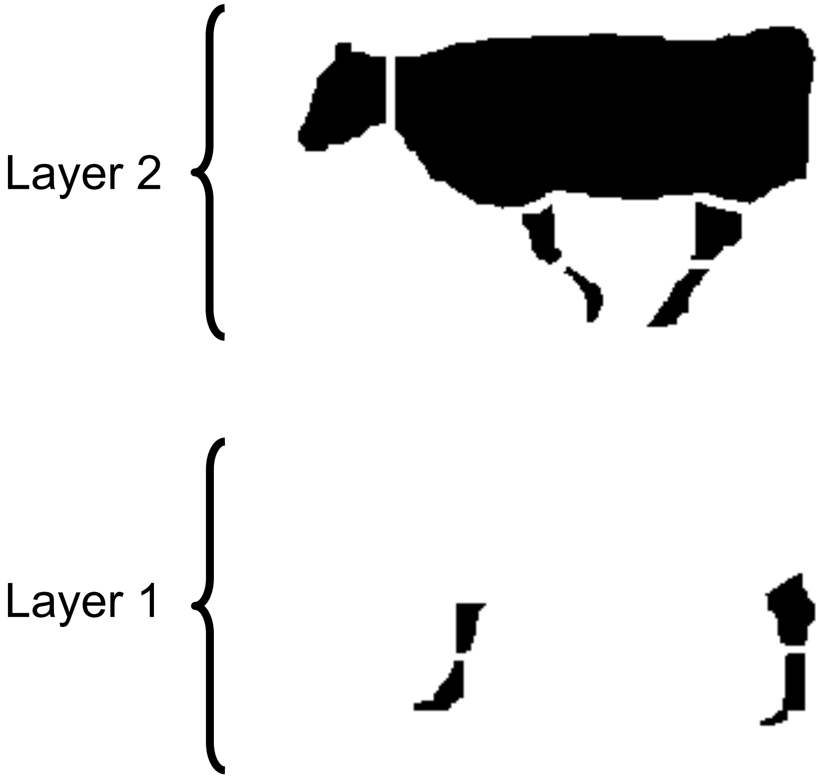
Detection

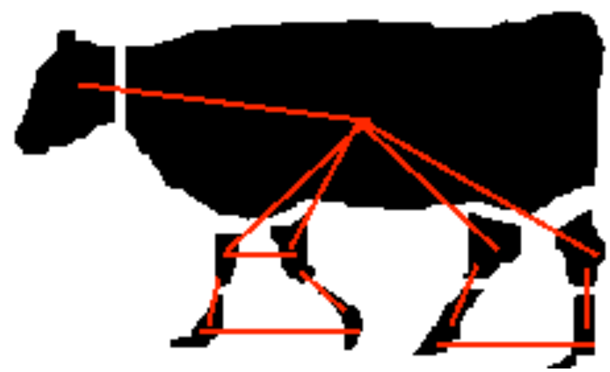
- BMVC 2004

Layered Pictorial Structures (LPS)

- Generative model

- Composition of parts + spatial layout

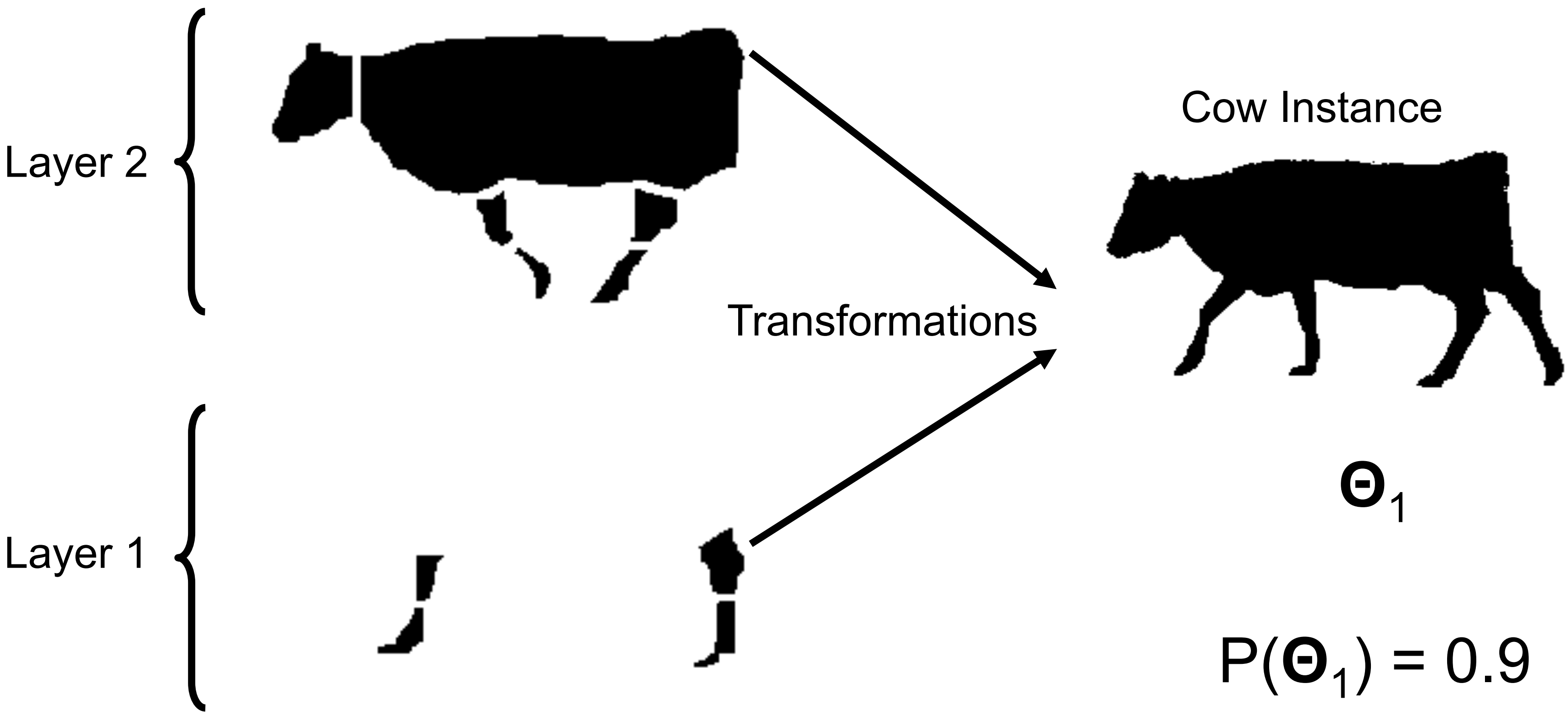




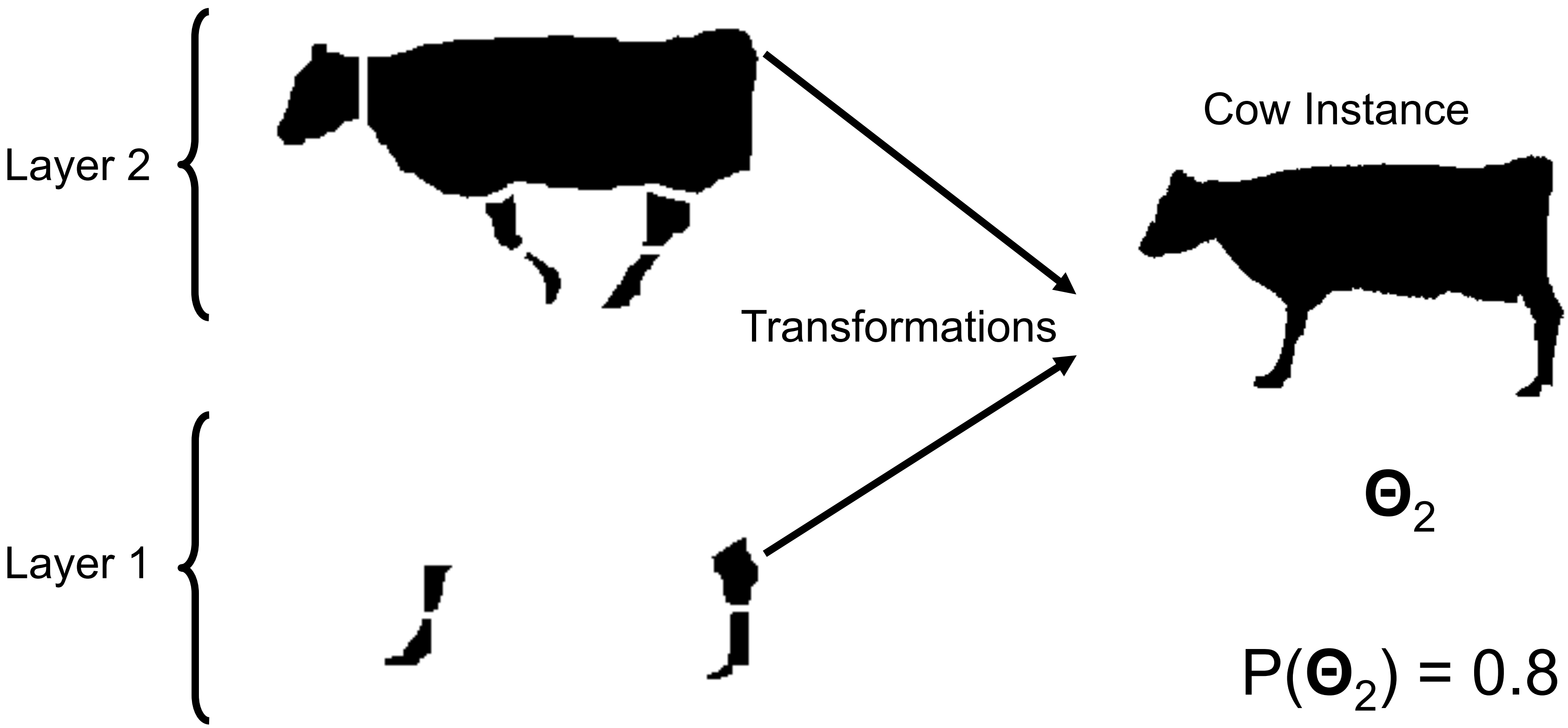
Spatial Layout
(Pairwise Configuration)

Parts in Layer 2 can occlude parts in Layer 1

Layered Pictorial Structures (LPS)



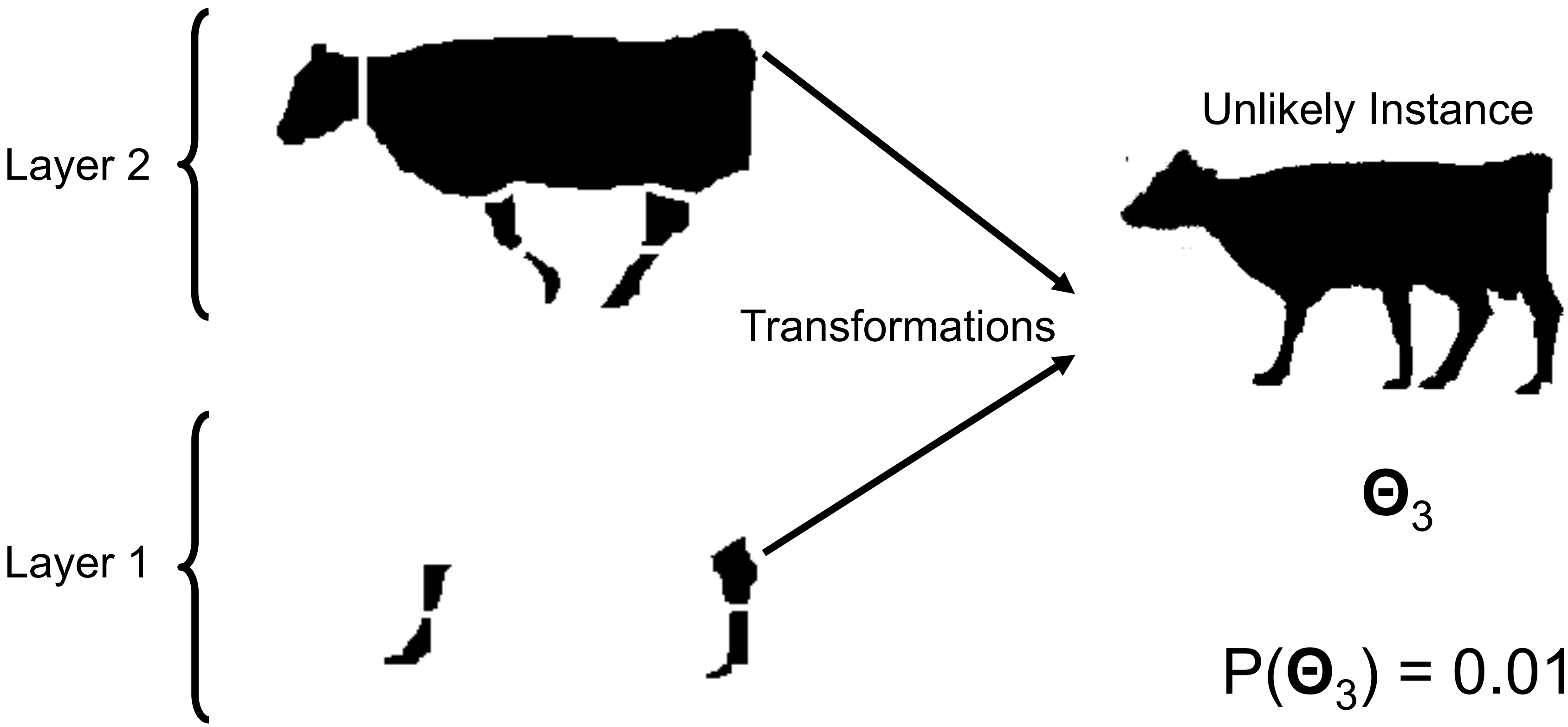
Layered Pictorial Structures (LPS)



$\mathbf{\Theta}_2$

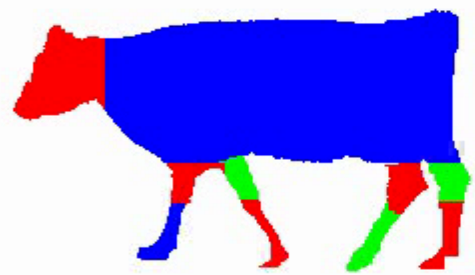
$P(\mathbf{\Theta}_2) = 0.8$

Layered Pictorial Structures (LPS)

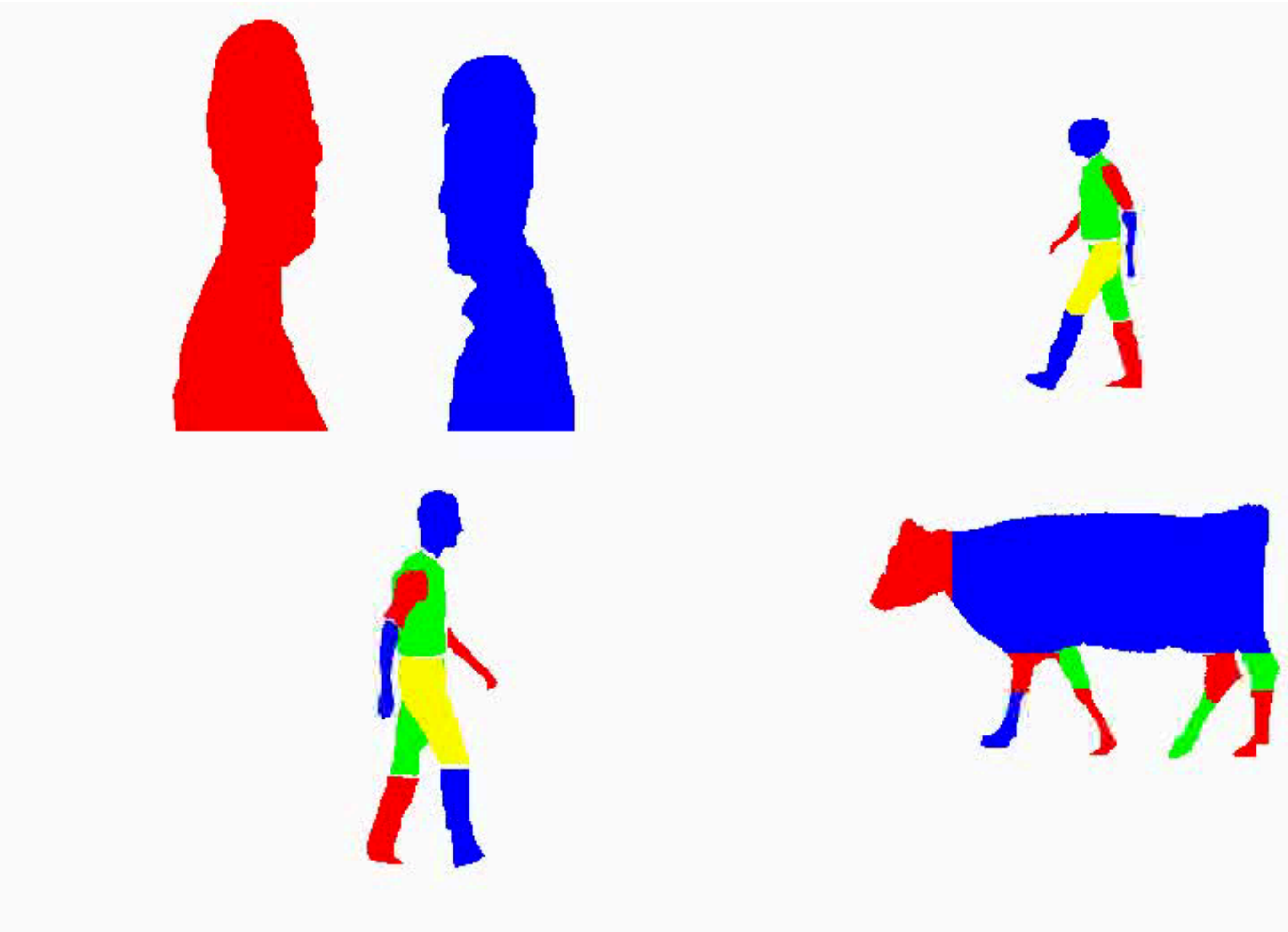


How to learn LPS

- From video via motion segmentation see Kumar Torr and Zisserman ICCV 2005.



How to learn LPS



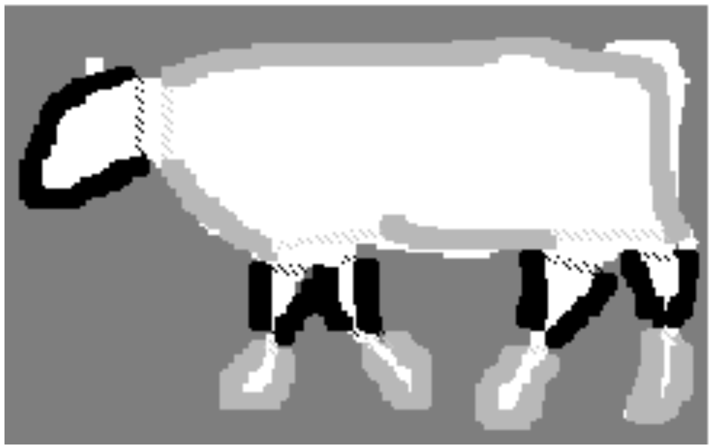
LPS for Detection

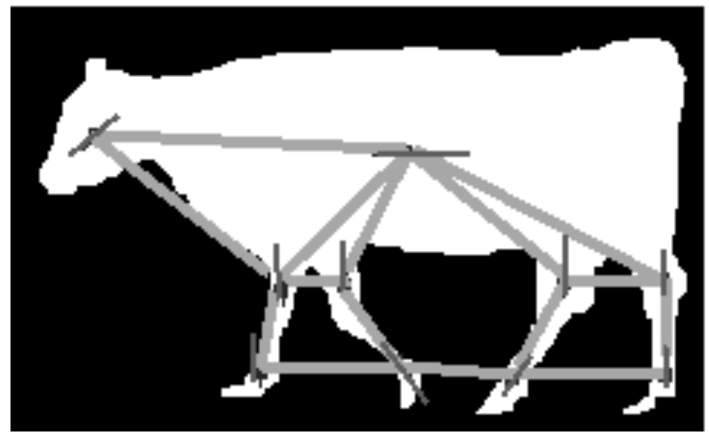
- Learning
  - Learnt automatically using a set of examples

- Detection
  - Matches LPS to image using Loopy Belief Propagation
  - Localizes object parts

Pictorial Structures (PS)

Fischler and Eschlager. 1973





PS = 2D Parts + Configuration

Aim: Learn pictorial structures in an unsupervised manner

Parts +
Configuration +
Relative depth

} Layered Pictorial Structures (LPS)

- Identify parts
- Learn configuration
- Learn relative depth of parts

Features

- Outline: z_1 Chamfer distance
- Interior: z_2 Textons

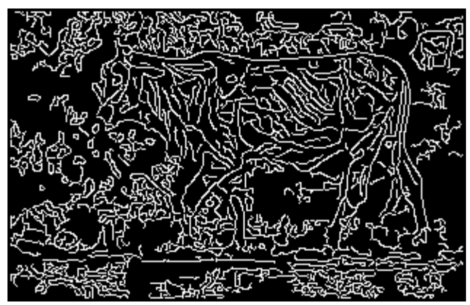
- Model joint distribution of z_1 z_2 as a 2D Gaussian.

Chamfer Match Score

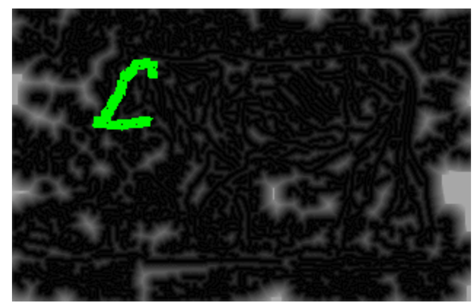
- Outline (z_1) : minimum chamfer distances over multiple outline exemplars

- $d_{cham} = 1/n \, \Sigma_i \min\{ \min_j \|u_i - v_j\|, \tau \}$

Image



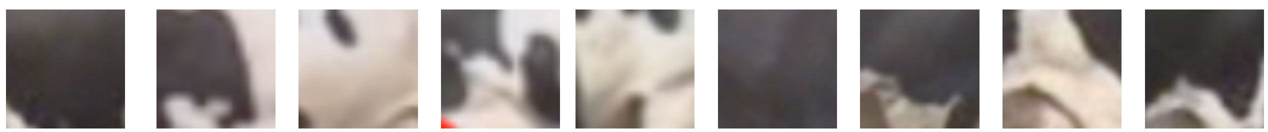
Edge Image



Distance Transform

Texton Match Score

- Texture(z_2) : MRF classifier
    - (Varma and Zisserman, CVPR '03)
- Multiple texture exemplars $\mathbf{x}$ of class t



- Textons: 3 X 3 square neighbourhood
- VQ in texton space
- Descriptor: histogram of texton labelling
- χ^2 distance

2. Fitting the Model

- Cascades of classifiers
  - Efficient likelihood evaluation
- Solving MRF
  - LBP, use fast algorithm
  - GBP if LBP doesn’t converge
  - Could use Semi Definite Programming (2003)
  - Recent work second order cone programming method best CVPR 2006.

Efficient Detection of parts

- Cascade of classifiers
- Top level use chamfer and distance transform for efficient pre filtering
- At lower level use full texture model for verification, using efficient nearest neighbour speed ups.

Cascade of Classifiers-for each part

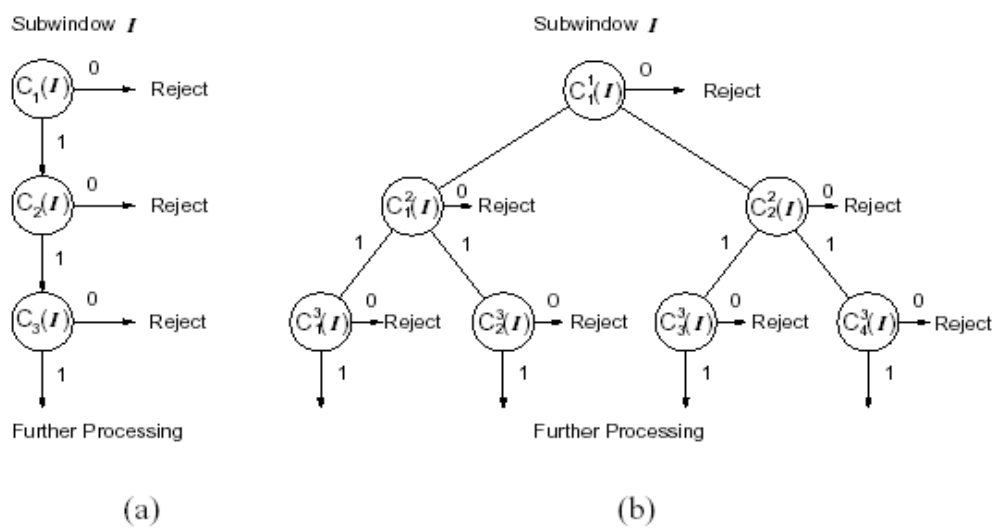
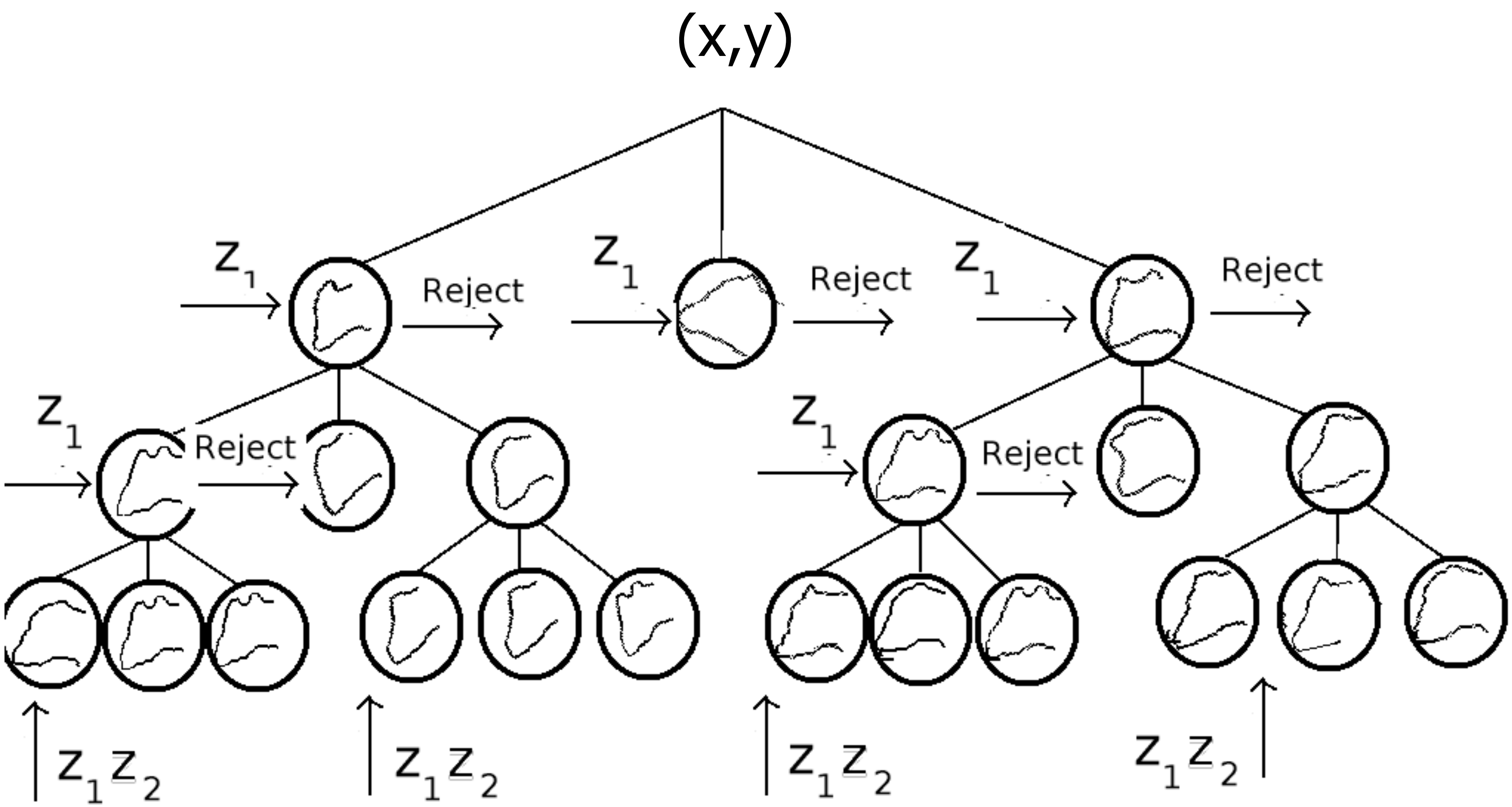


Fig. 1. Cascade of Classifiers. **(a)** *A cascade of classifiers for a single object class where each classifier has a high detection and moderate false positive rate.* **(b)** *Classifiers in a tree structure; in a tree-based object recognition scheme each leaf corresponds to a single object class. When objects in the subtrees have similar appearance, classifiers can be used to quickly prune the search. A binary tree is shown here, but the branching factor can be larger than two.*

Y. Amit, and D. Geman, 97?; S. Baker, S. Nayer 95

High Levels based on Outline

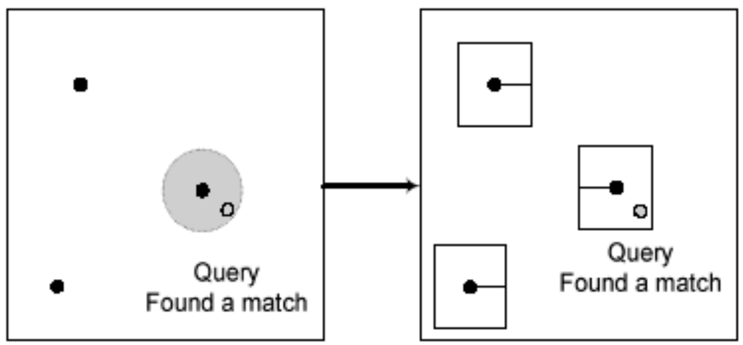


Low levels on Texture

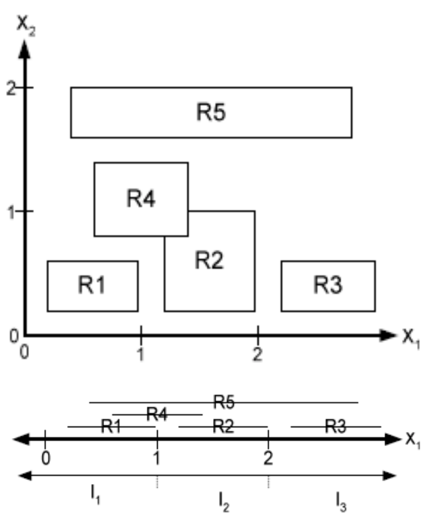
- The top levels of the tree use outline to eliminate patches of the image.
- Efficiency: Using chamfer distance and pre computed distance map.
- Remaining candidates evaluated using full texture model.

Efficient Nearest Neighbour

- Goldstein, Platt and Burges (MSR Tech Report, 2003)



Conversion from fixed distance to rectangle search

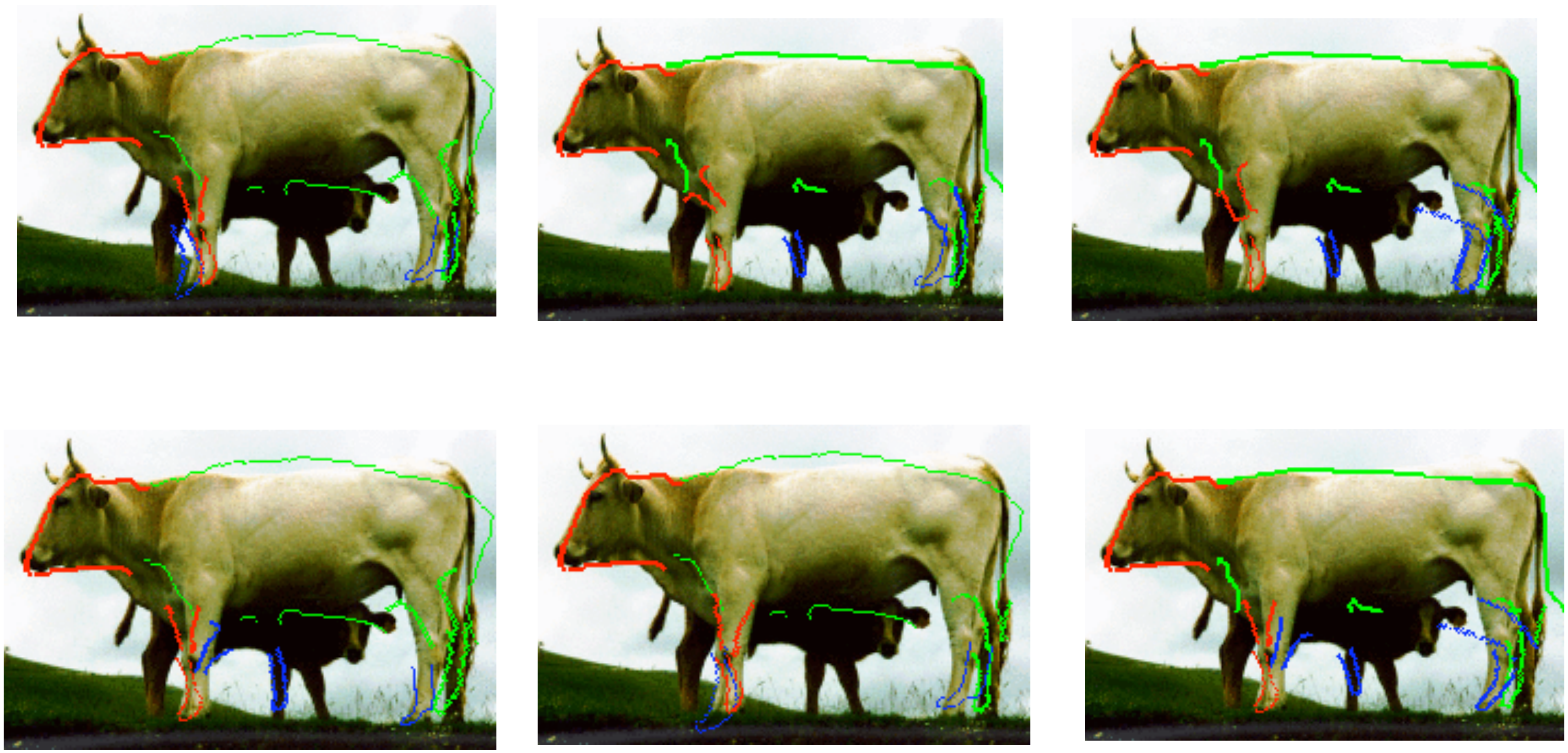


- $\text{bitvector}^i_j(R_k) \begin{rcases} = 1 \\ = 0 \end{rcases}$ $R_k \in I_i$ in dimension j
- Nearest neighbour of **x**
- Find intervals in all dimensions
- 'AND' appropriate bitvectors
- Nearest neighbour search on pruned exemplars

Recently solve via Integer Programming

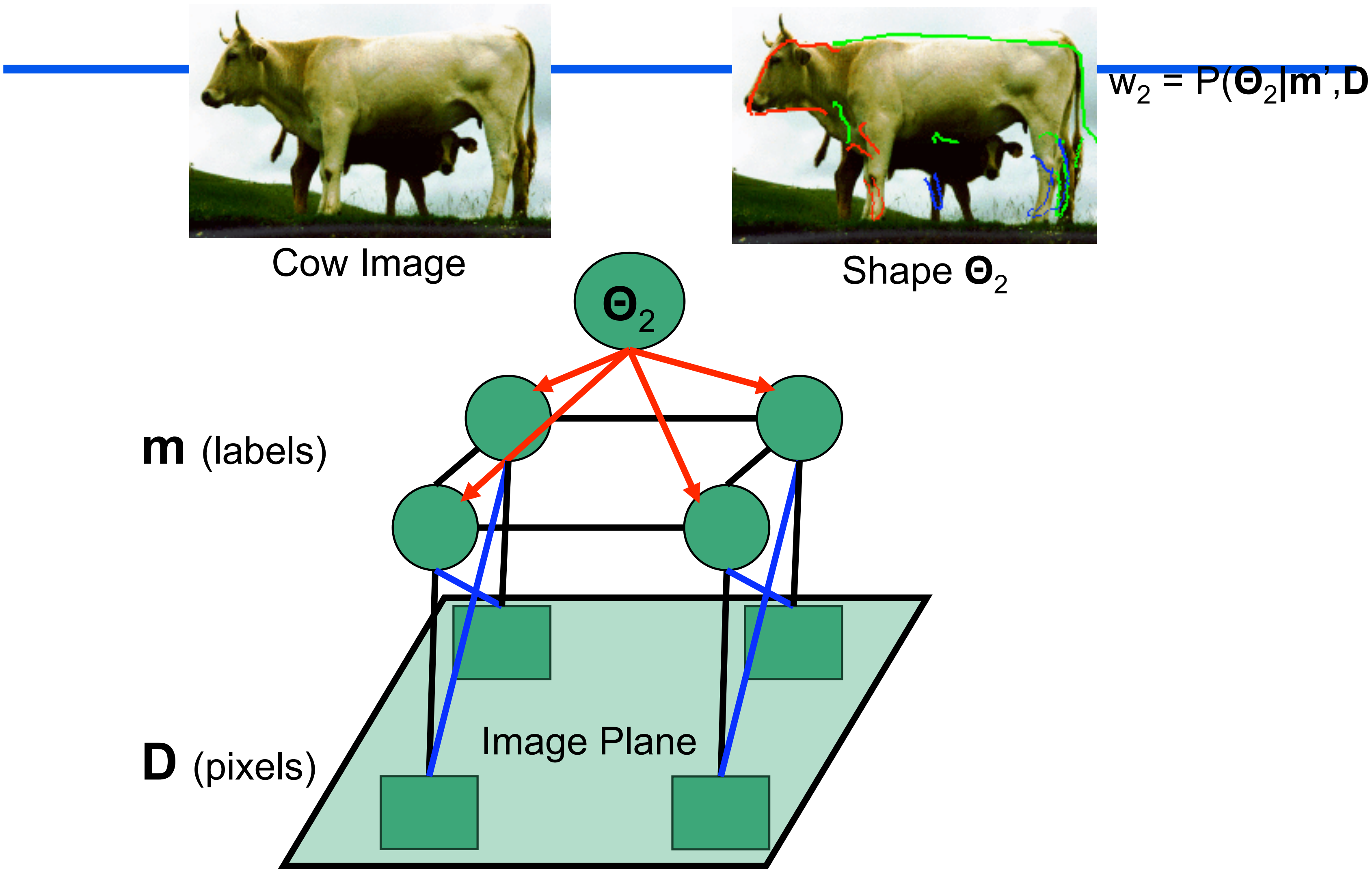
- SDP formulation (Torr 2001, AI stats)
- SOCP formulation (Kumar, Torr & Zisserman this conference)
- LBP (Huttenlocher, many)

Results



- Different samples *localize* different parts well.
- We cannot use only the MAP estimate of the LPS.

M-Step



- Best labelling found efficiently using a Single Graph Cut

Results

Using LPS Model for Cow

Image	Segmentation
	
	

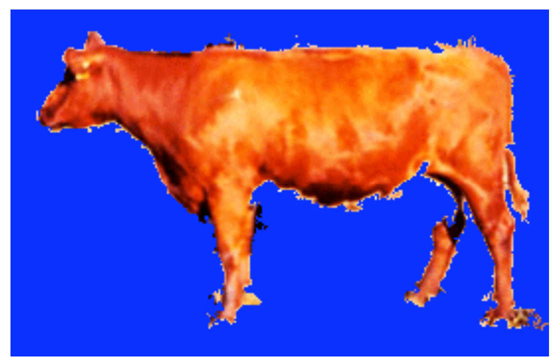
Results

Using LPS Model for Cow

In the absence of a clear boundary between object and background

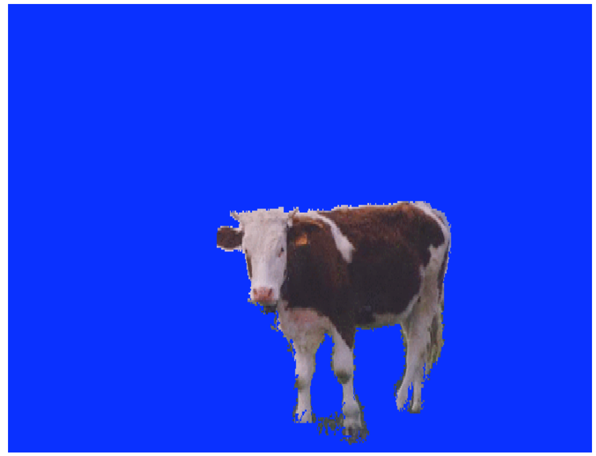
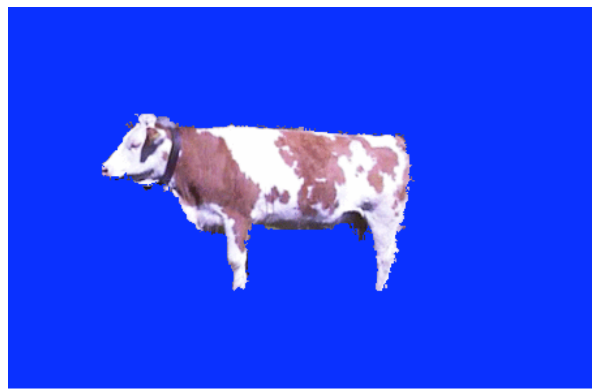
Image

Segmentation

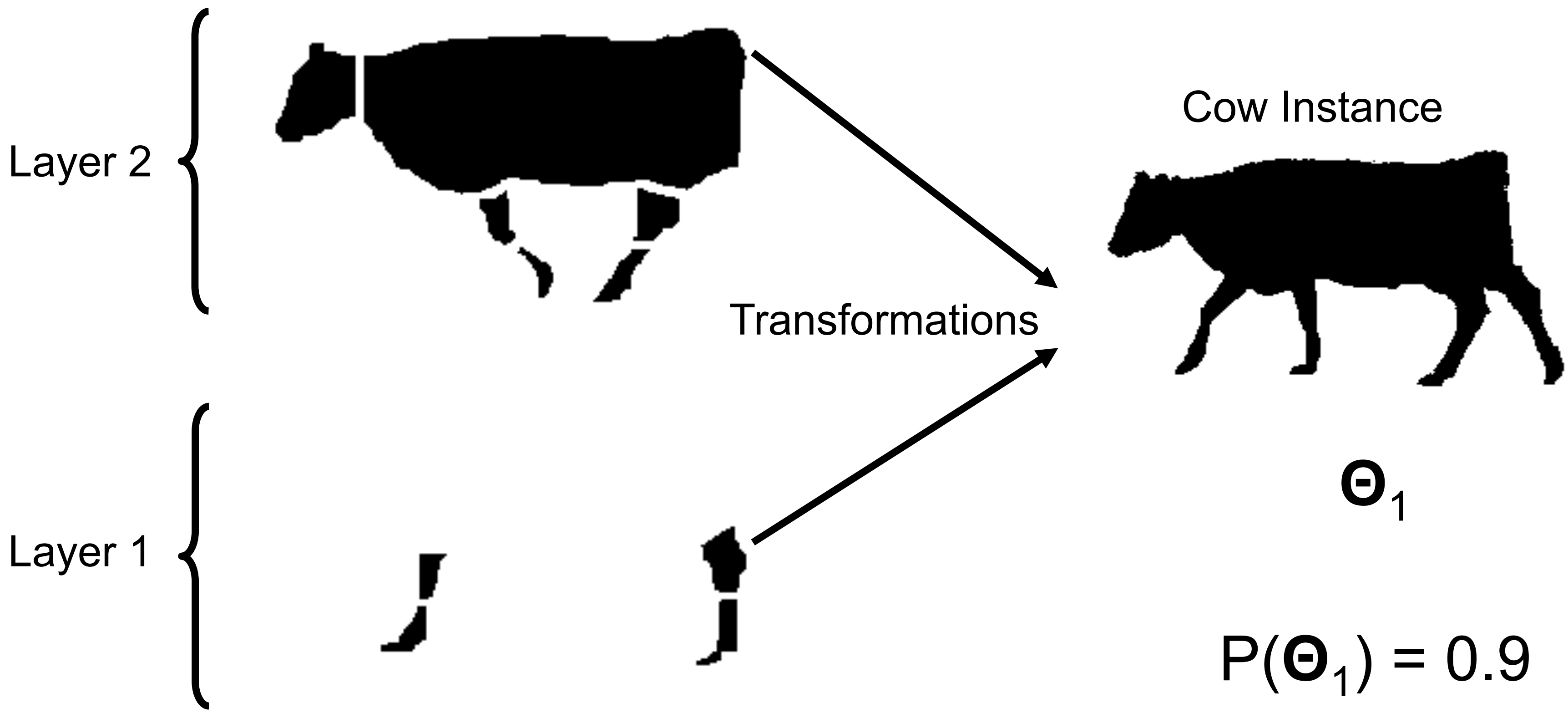


Results

Using LPS Model for Cow

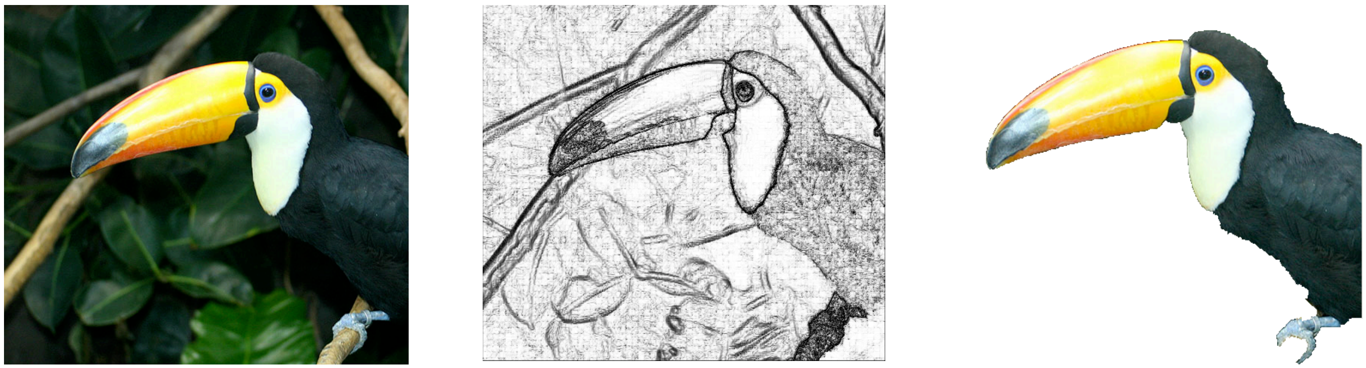
Image	Segmentation
	
	

Do we really need accurate models?

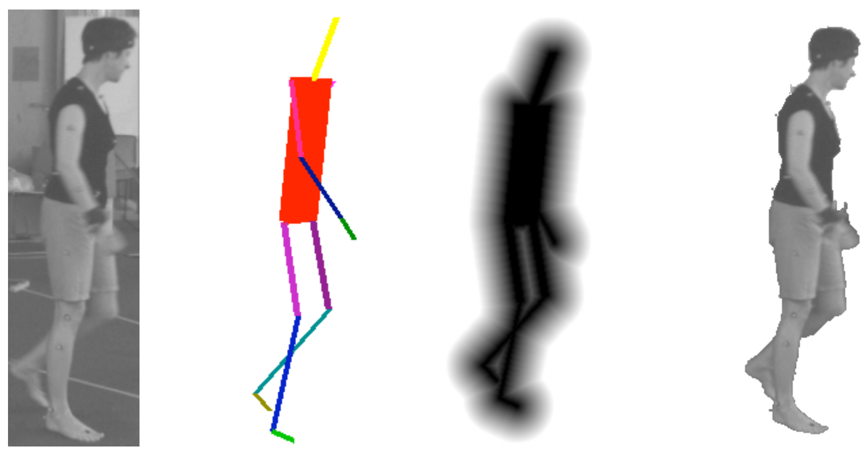


Do we really need accurate models?

- Segmentation boundary can be extracted from edges



- Rough 3D Shape-prior enough for region disambiguation



Energy of the Pose-specific MRF

Energy to be minimized

Unary term

Shape prior

Pairwise potential

Potts model

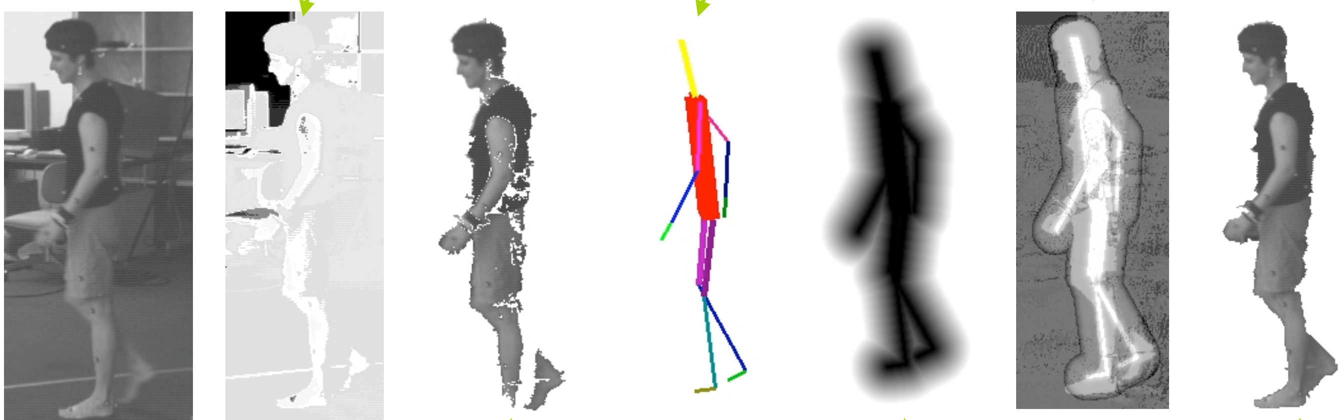
$$\Psi(\mathbf{x},\boldsymbol{\Theta}) = \sum_{views}\sum_i\Big(\phi(\mathbf{D}|x_i) \boxed{+\phi(x_i|\boldsymbol{\Theta})+} \sum_j \big(\phi(\mathbf{D}|x_i,x_j)+\psi(x_i,x_j)\big)\Big)$$

$$\frac{1}{1+\exp(\mu*(d(i,\boldsymbol{\Theta})-d_r))}$$

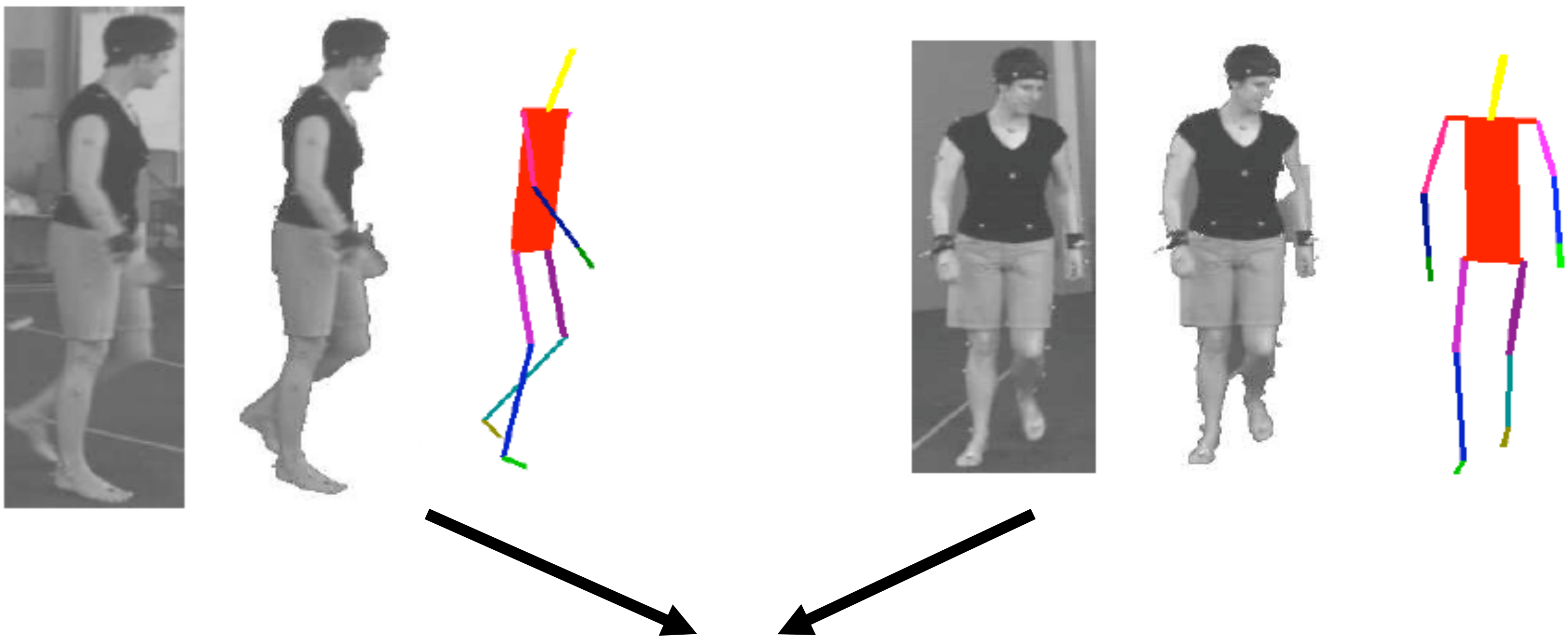
$\phi(x_i|\boldsymbol{\Theta})$

But what should be the value of **θ**?

The different terms of the MRF



Can segment multiple views simultaneously



$$\Psi(\mathbf{x}, \mathbf{\Theta})$$

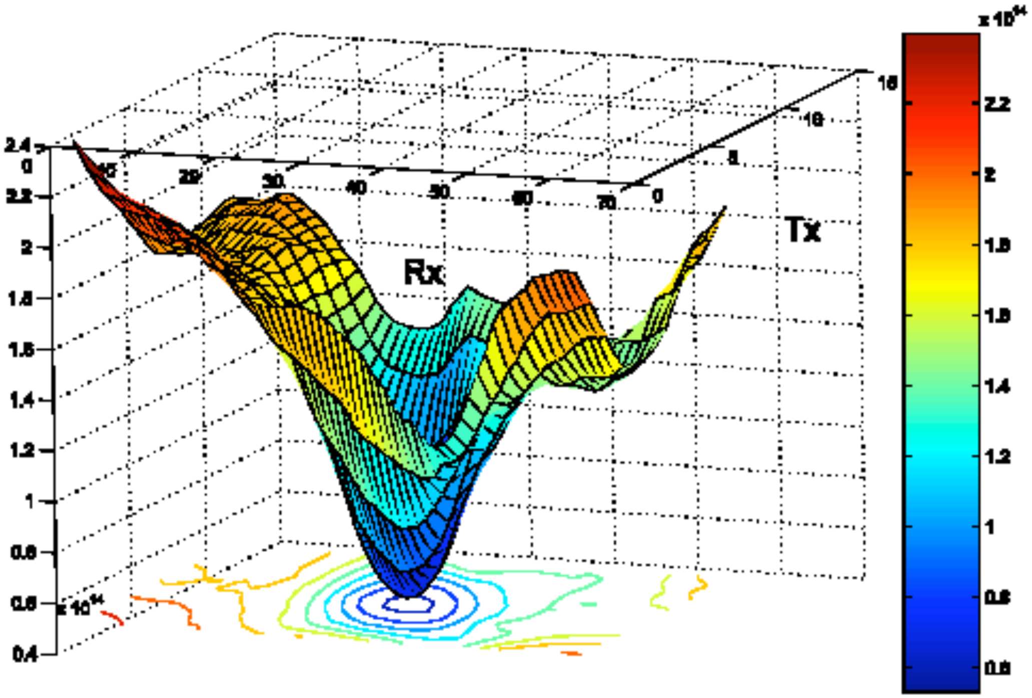
$$\mathbf{\Theta_{opt}} = \arg\min_{\mathbf{\Theta}} \Big(\min_{\mathbf{x}} \sum_{\text{views}} \big(\Psi\ (\mathbf{x}, \mathbf{\Theta})\big)\Big)$$

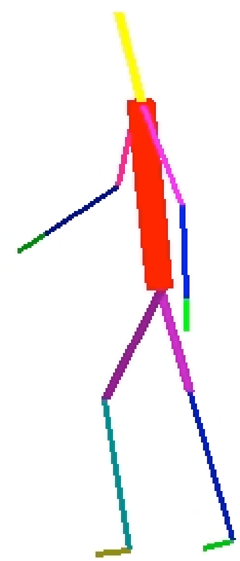
Solve via gradient descent

- Comparable to level set methods
- Could use other approaches (e.g. Objcut)
- Need a graph cut per function evaluation

Formulating the Pose Inference Problem

$$\mathbf{\Theta}_{\mathbf{opt}} = \arg\min_{\mathbf{\Theta}}(\min_{\mathbf{x}} \Psi_3(\mathbf{x}, \mathbf{\Theta}))$$





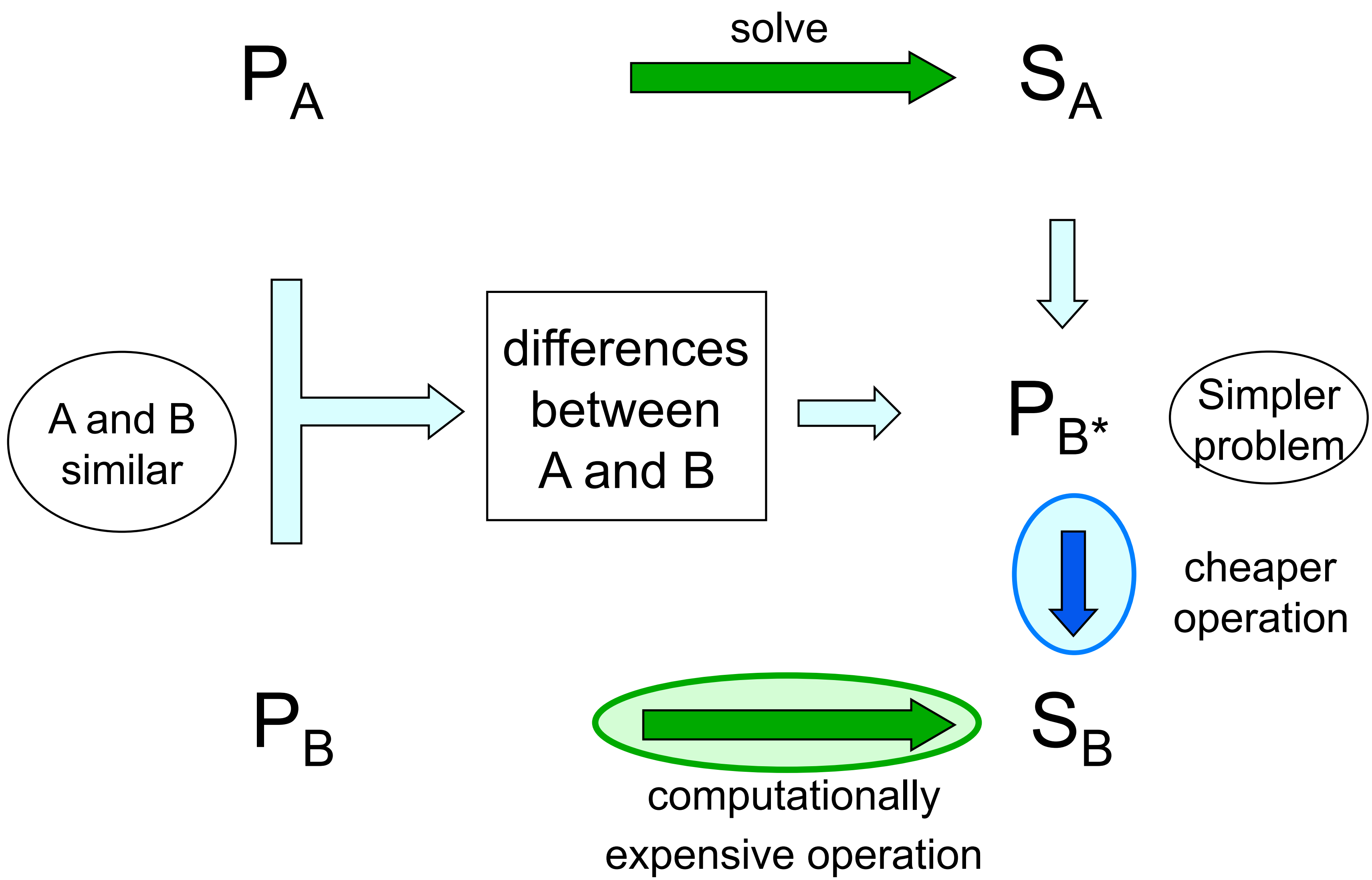
But…

… to compute the MAP of E(x) w.r.t the pose, it means that the unary terms will be changed at **EACH** iteration and the maxflow recomputed!

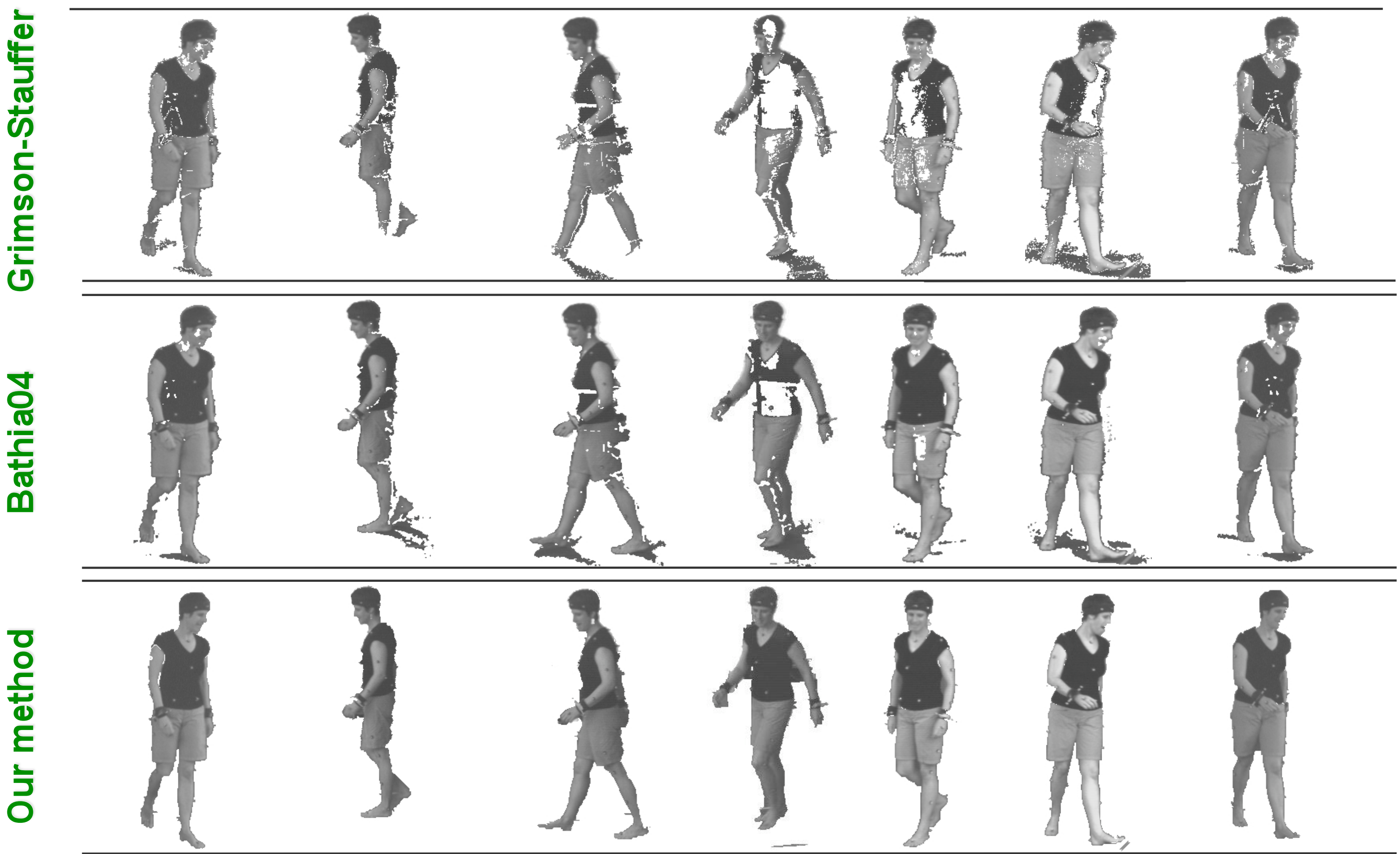
However…

- Kohli and Torr showed how dynamic graph cuts can be used to efficiently find MAP solutions for MRFs that change minimally from one time instant to the next: Dynamic Graph Cuts (ICCV05).

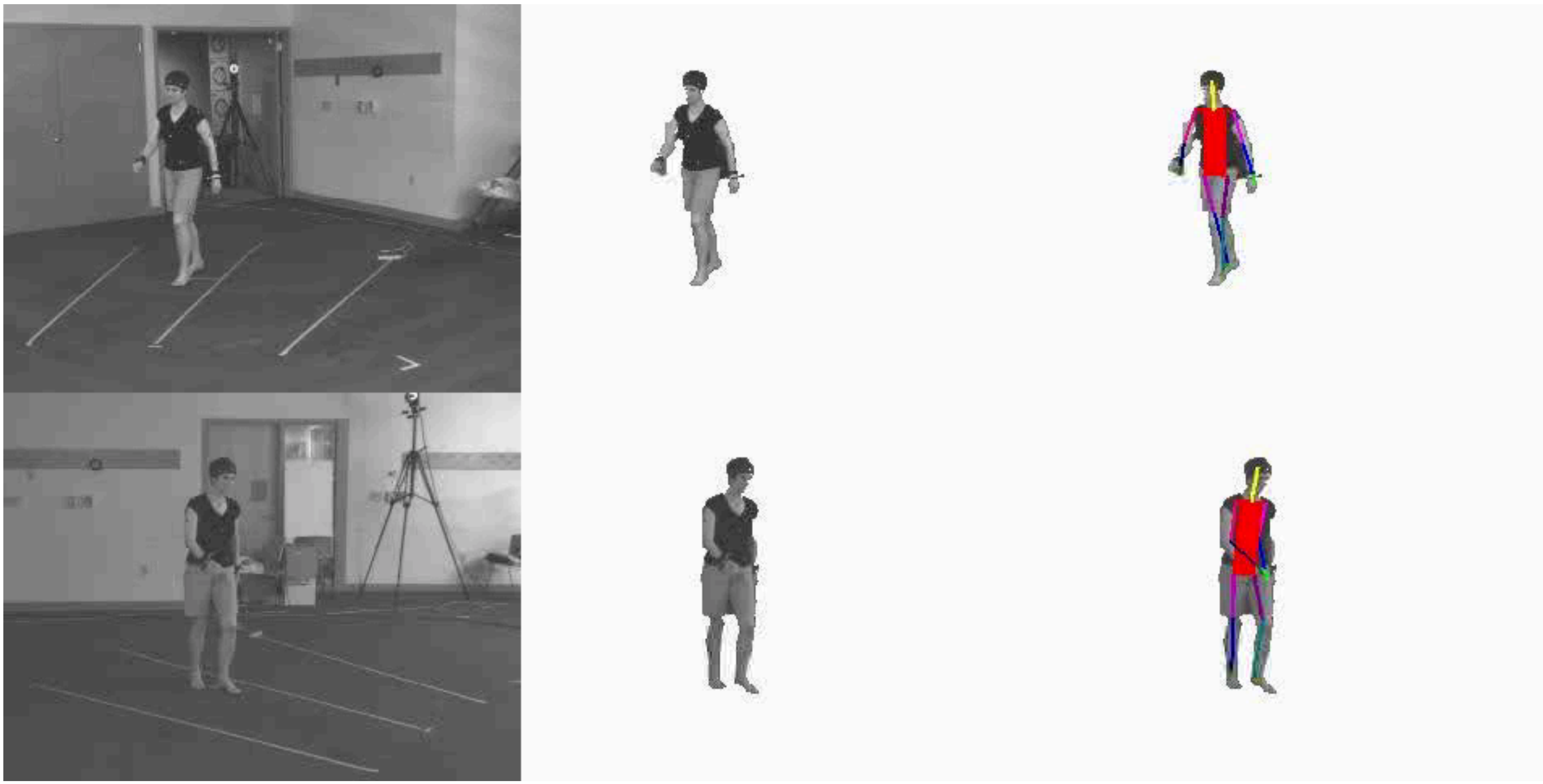
Dynamic Graph Cuts



Segmentation Comparison



Segmentation



Combine Recognition and Segmentation

- Using all the information should help…
  - E.g. segmentation result can be used to eliminate false positives.
- Face/Head and shoulders segmentation particularly useful for applications such as Skype or Windows Messenger.

ObjCut Face Segmentation

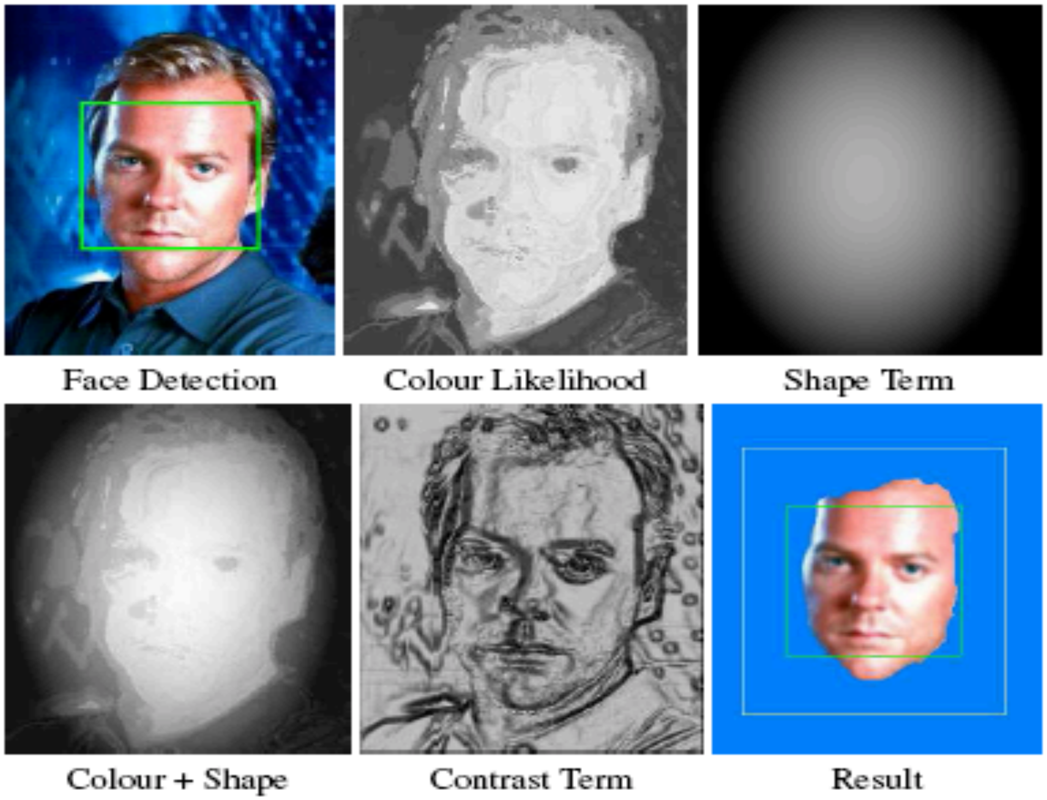
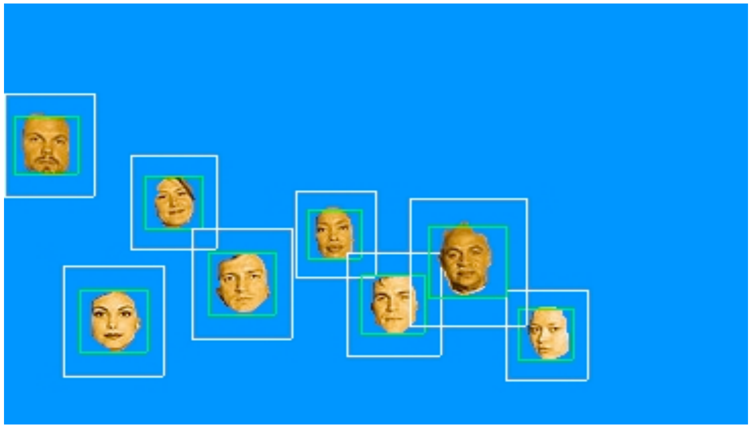


Fig. 3. *Different terms of the shape-prior + MRF energy function. The figure shows the different terms of the energy function for a particular face detection and the corresponding image segmentation obtained.*

Face Detector and ObjCut



Segmentation + Recognition

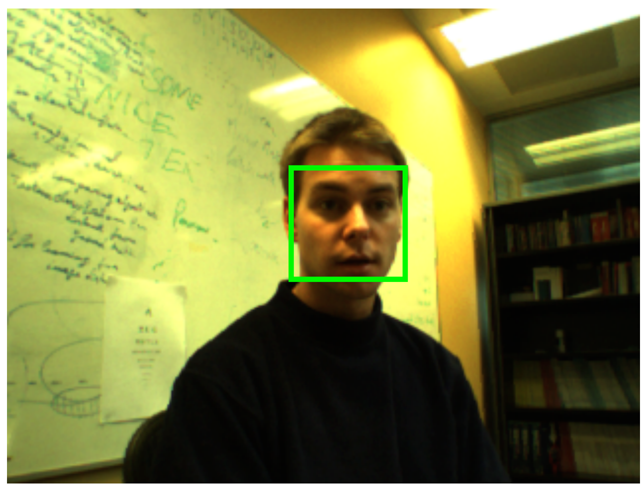
to prune false positives

(see also recent work D Ramanan CVPR 07)

Fig. 4. *The figure shows an image from the INRIA pedestrian data set. After running our algorithm, we obtain four face segmentations, one of which (the one bounded by a black square) is a false detection. The energy-per-pixel values obtained for the true detections were 74, 82 and 83 while that for the false detection was 87. As you can see the energy of false detection is significantly higher than that of the true detections, and can be used to detect and remove it.*

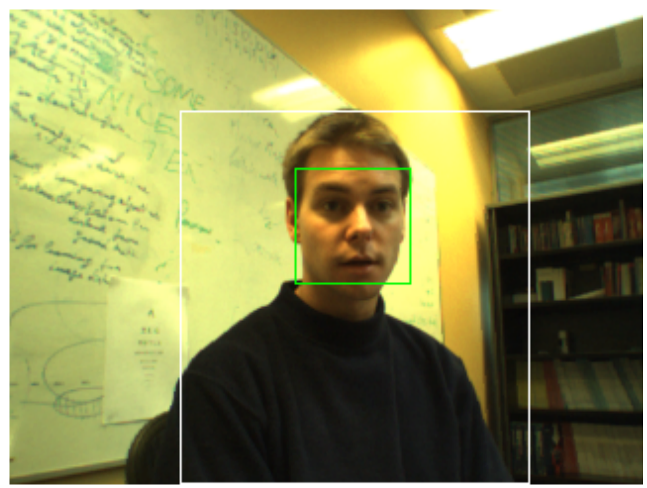
Initialisation

- Use off-the-shelf face detector to find location of face in an image



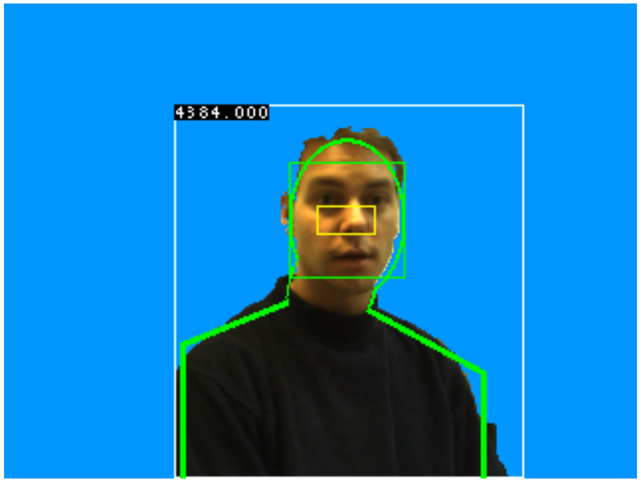
Initialisation

- Place shape prior relative to face detection
- Define region over which to perform segmentation



Adjustment

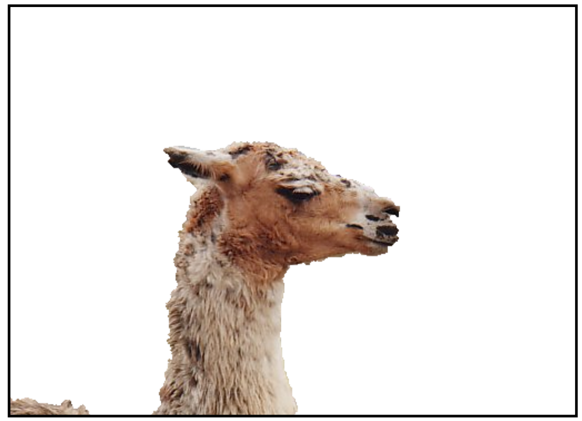
- Vary parameters of shape prior to find lowest segmentation energy

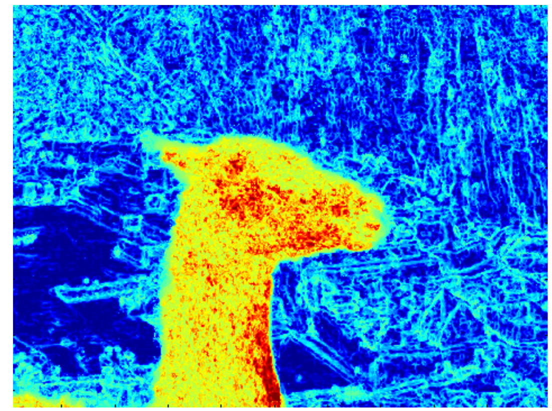


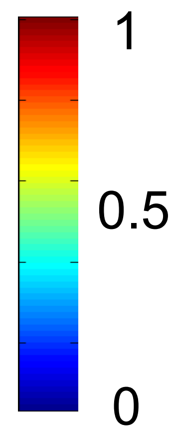
Other Applications of Dynamic Graph Cuts

Computing Max-marginals

Image (MSRC) | **Segmentation** | **Max-Marginal (Fg)**







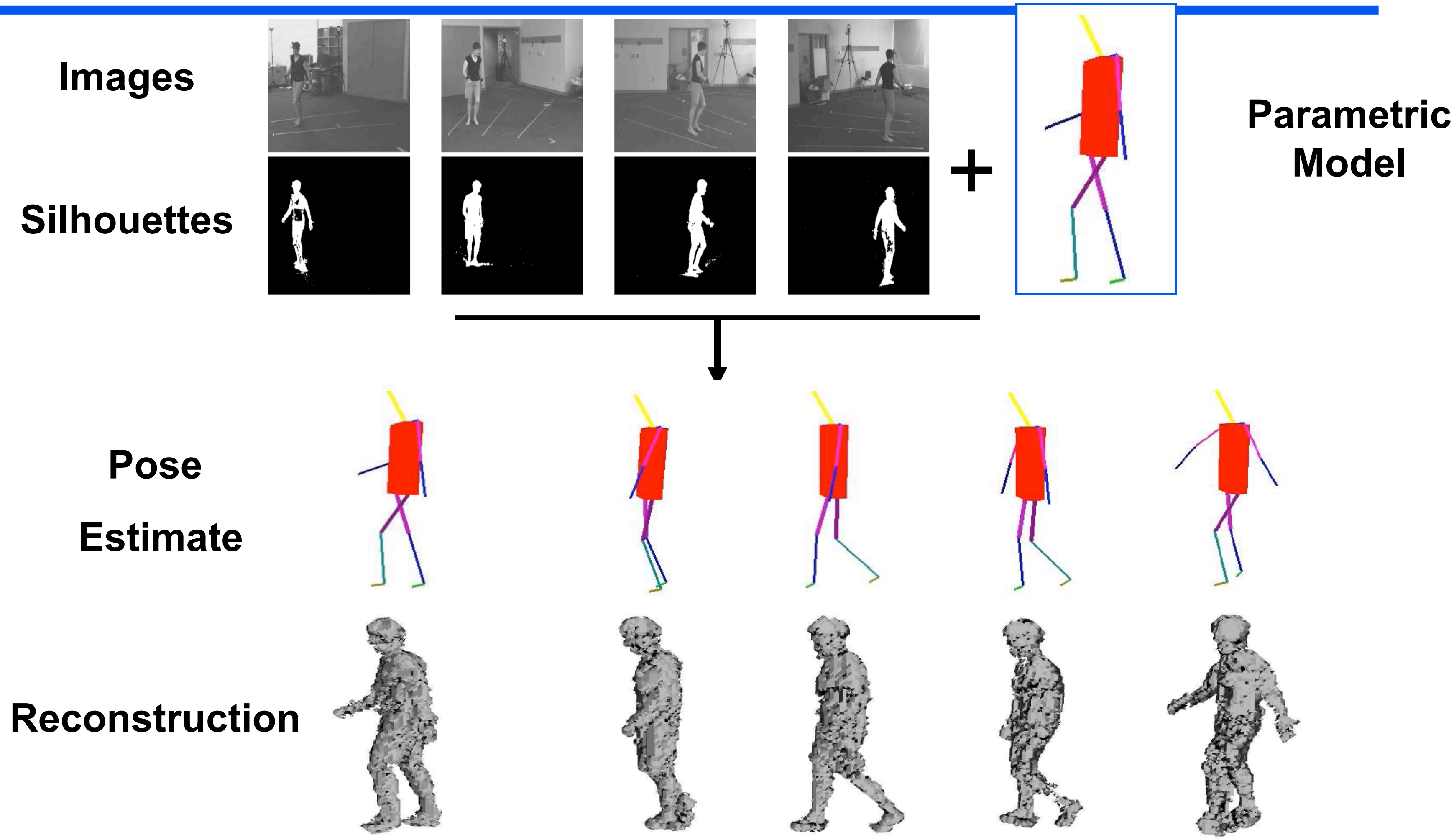
Graph Cuts : 300 msec per MM (51.2 days)

Dynamic Graph Cuts : 0.002 msec per MM (1.2 secs)

Parameter Learning in CRFs

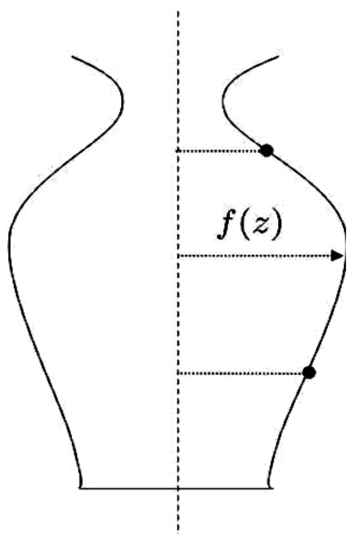
- Maximum Likelihood Parameter Learning
  - Kumar and Herbert
  - NIPS 2004, EMMCVPR 2005

Shape Priors For Reconstruction



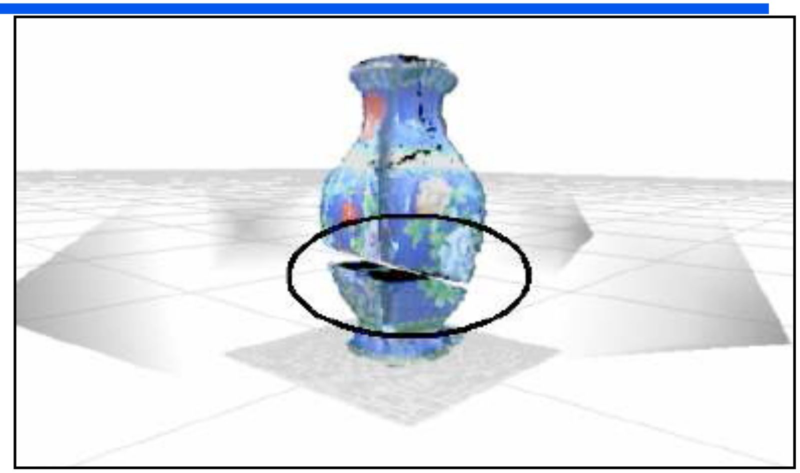
OXFORD BROOKES UNIVERSITY

Shape Priors For Reconstruction

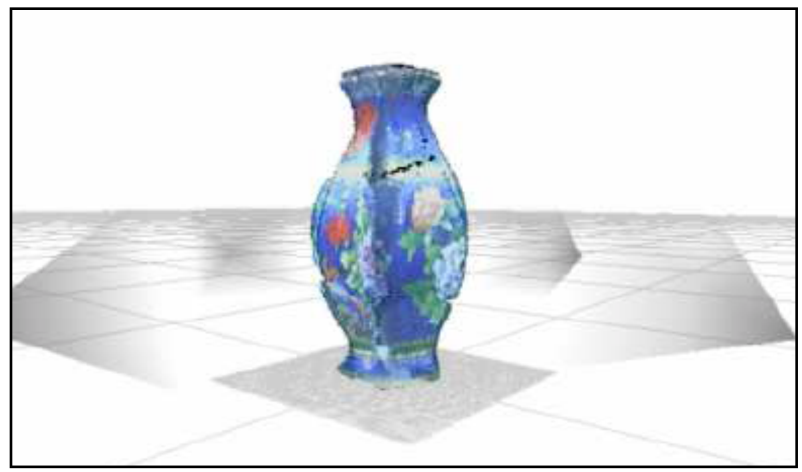


Shape Model

Visual Hull



Our Reconstruction



Multi Label Problems

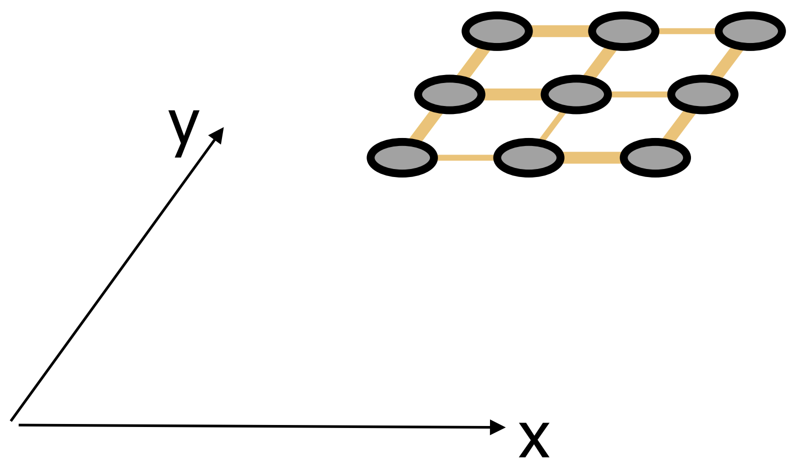
- So far we have considered generic cuts or 2 label problems

- Now we consider multi label problems

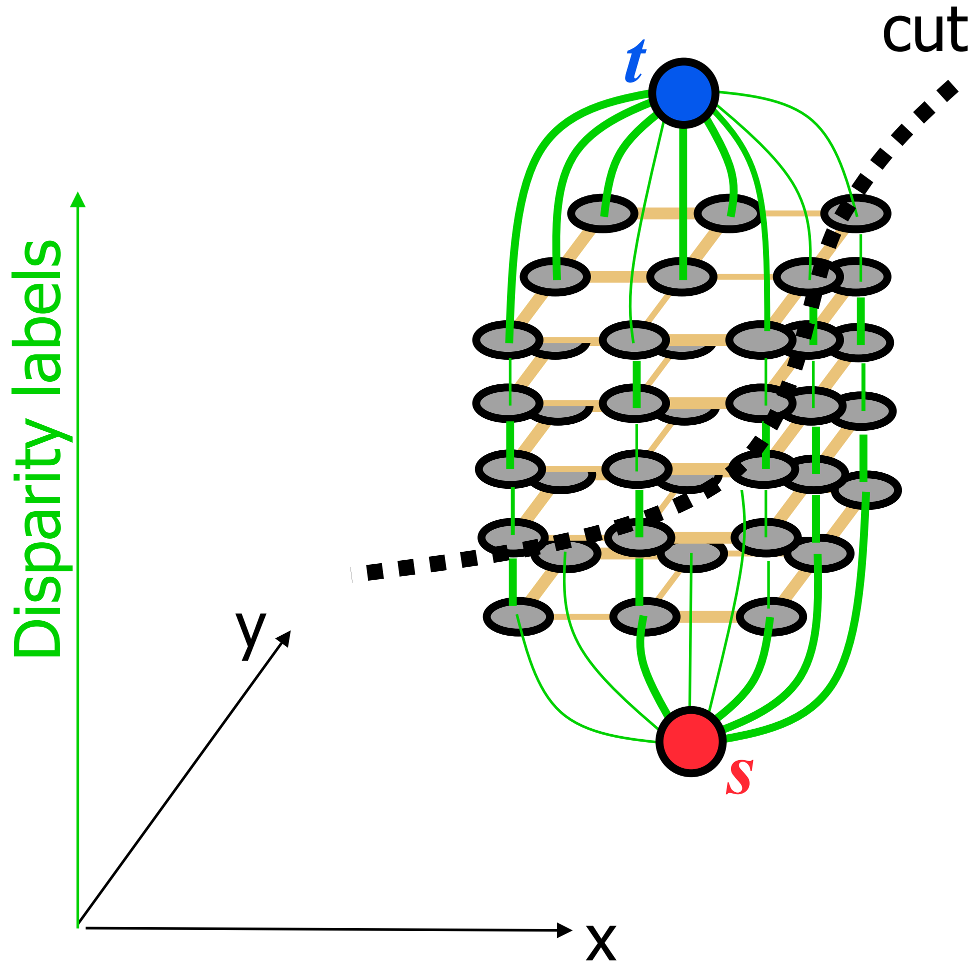
s-t graph-cuts for multi-label problems

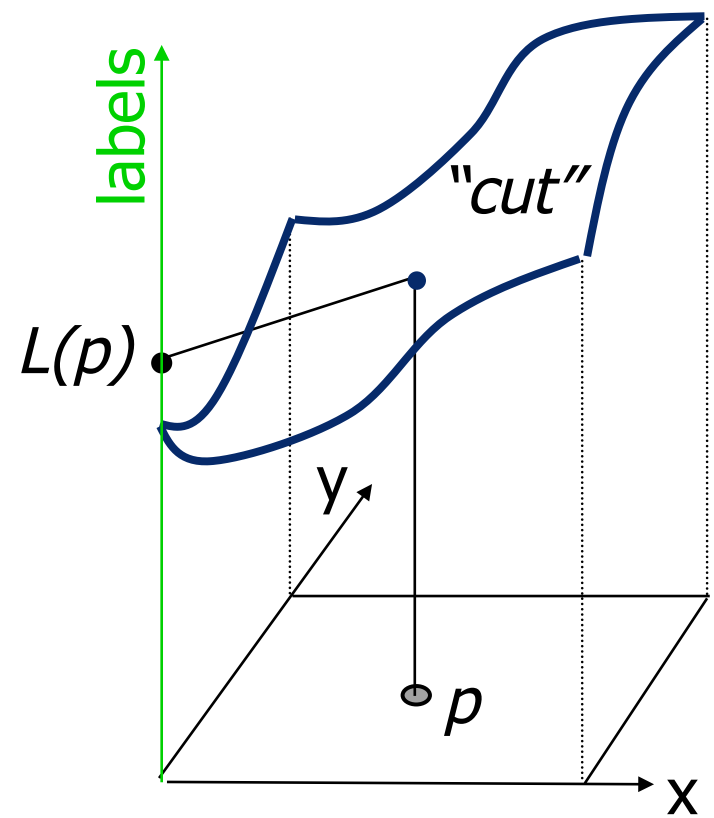
- Multi-scan-line stereo
  - Roy & Cox 1998, 1999
  - Ishikawa & Geiger 1999 (occlusion handling)
- "Linear" interaction energy
  - Ishikawa & Geiger 1998
  - BVZ 1998
- Convex interaction energy
  - Ishikawa 2000, 2003

Multi-scan-line stereo with *s-t* graph cuts (Roy&Cox'98)

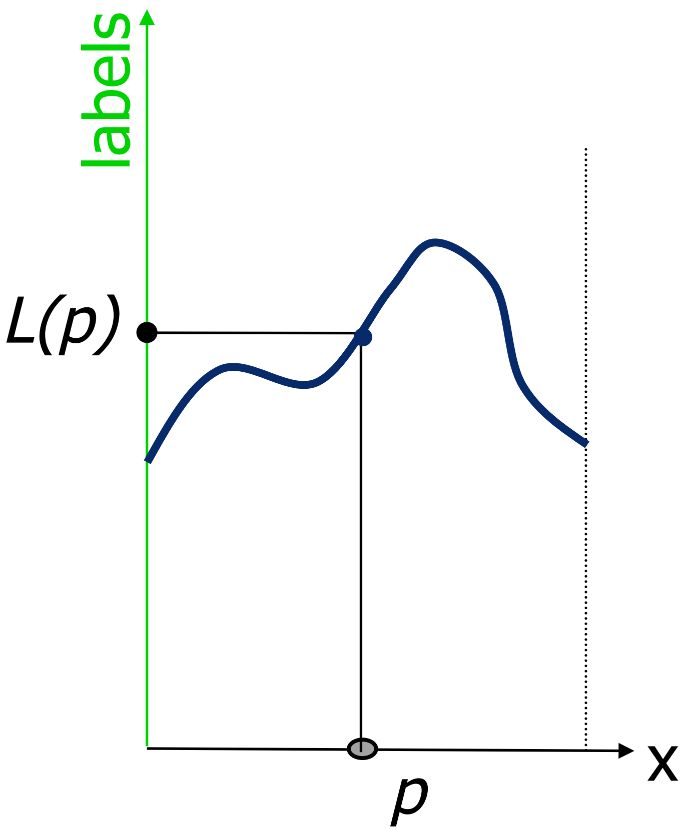


Multi-scan-line stereo with *s-t* graph cuts (Roy&Cox'98)

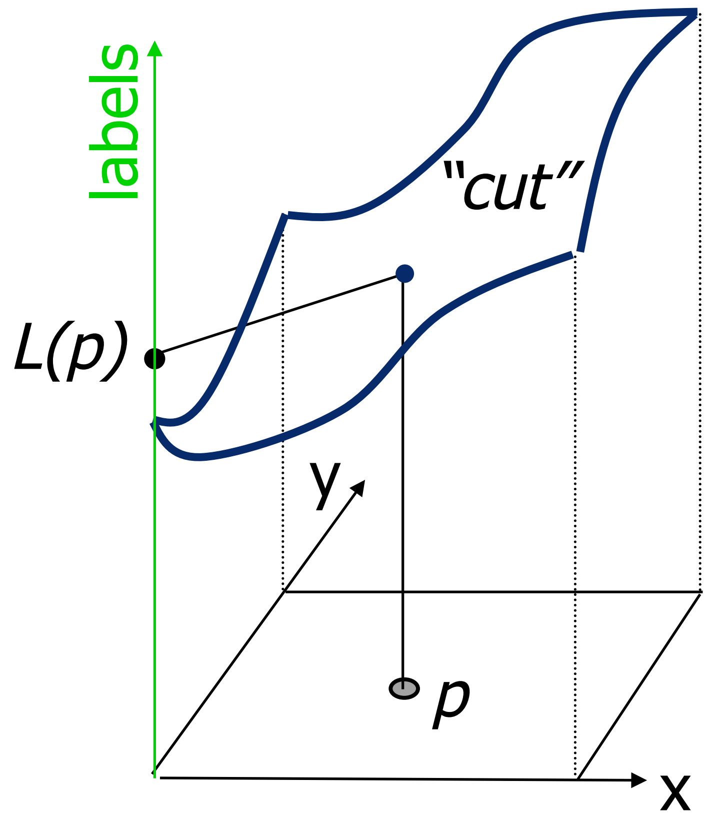




Scan-line stereo vs. Multi-scan-line stereo



Dynamic Programming
(single scan line optimization)



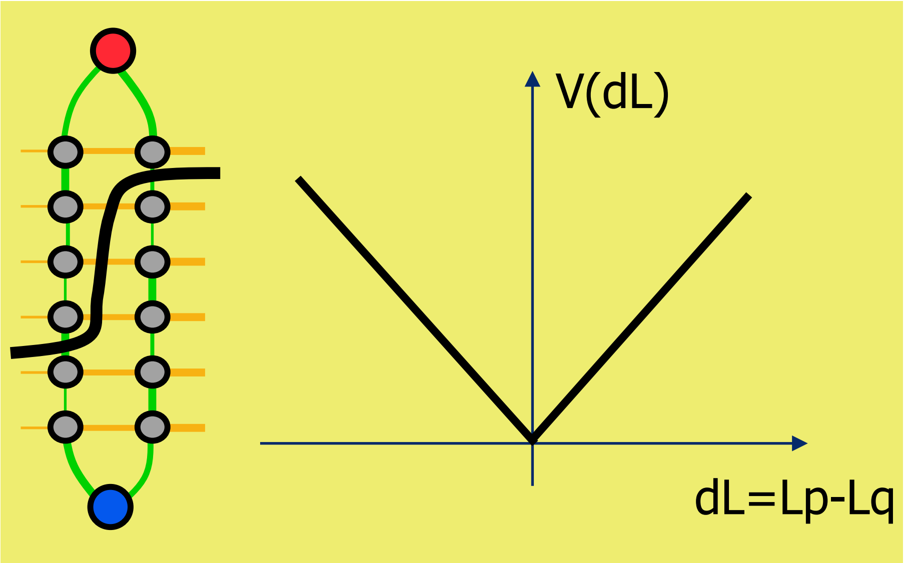
s-t Graph Cuts
(multi-scan-line optimization)

s-t graph-cuts for multi-label energy minimization

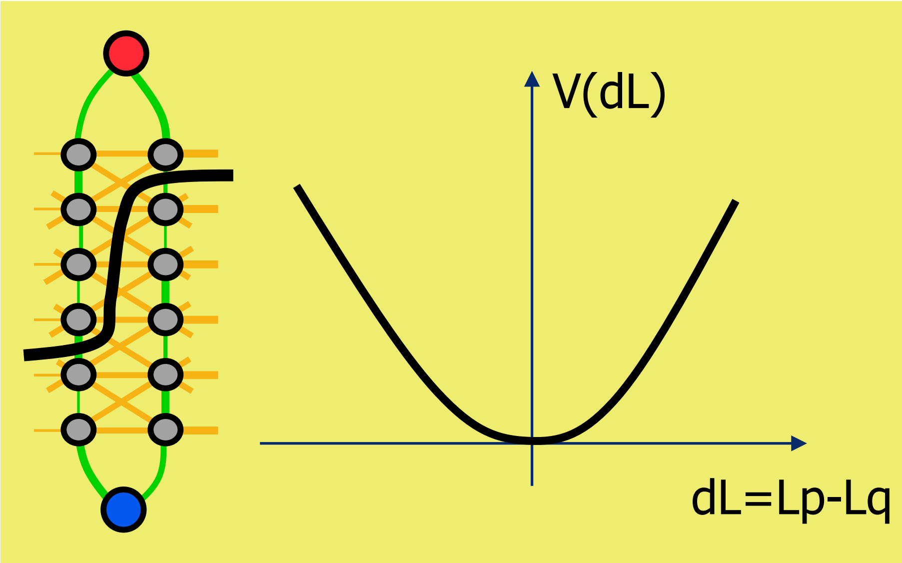
- Ishikawa 1998, 2000, 2003
- Modification of construction by Roy&Cox 1998

$$\boxed{E(L) \quad = \quad \sum_{p} -D_p(L_p) \; + \; \sum_{pq \in N} V(L_p, L_q)} \qquad L_p \in R^1$$

Linear interactions



"Convex" interactions



Extensions

- Ishikawa work generalizes nicely by Schlesinger and Flach
- We have an ICML case that deals with partial optimality non sub modular energies

---

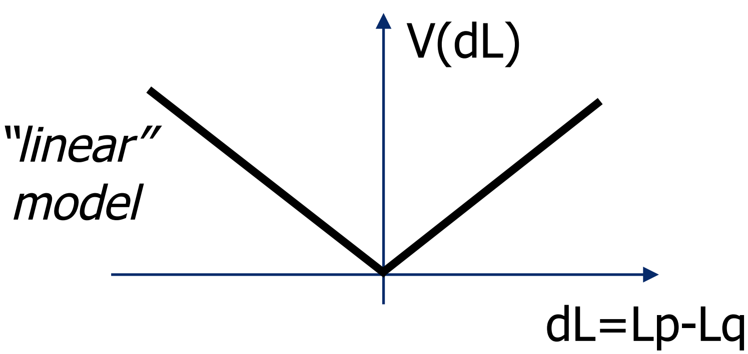
On Partial Optimality in Multi-label MRFs

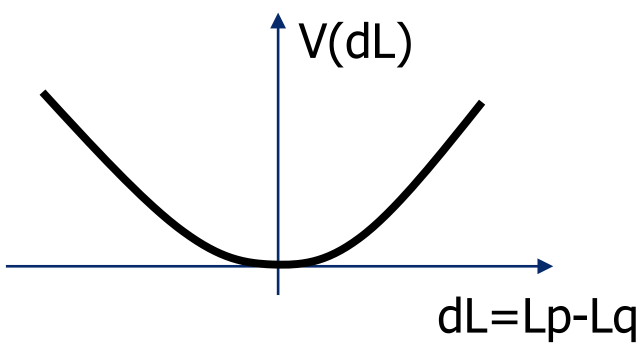
Pushmeet Kohli PKOHLI@MICROSOFT.COM
Alexander Shekhovtsov SHEKHOVT@CMP.FELK.CVUT.CZ
Carsten Rother CARROT@MICROSOFT.COM
Vladimir Kolmogorov VNK@ADASTRAL.UCL.AC.UK
Philip Torr PHILIPTORR@BROOKES.AC.UK

More details from Sasha who is here

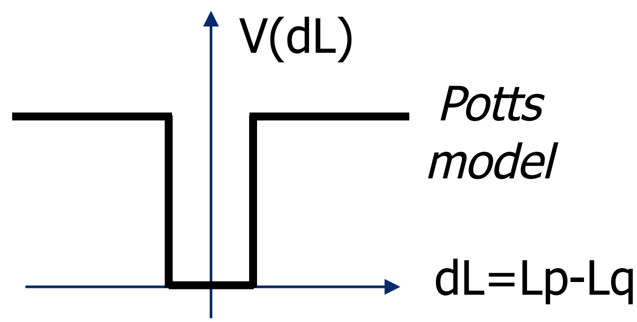
Pixel interactions:
"*convex*" vs. "*discontinuity-preserving*"

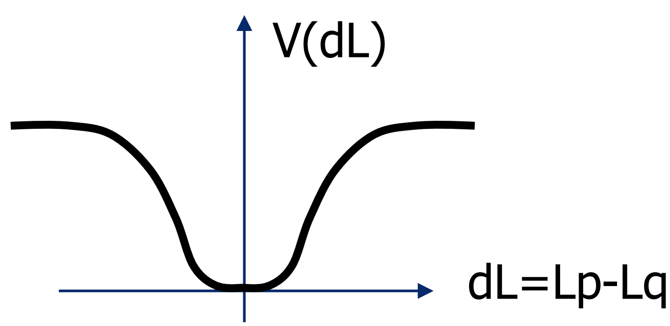
"Convex"
interactions



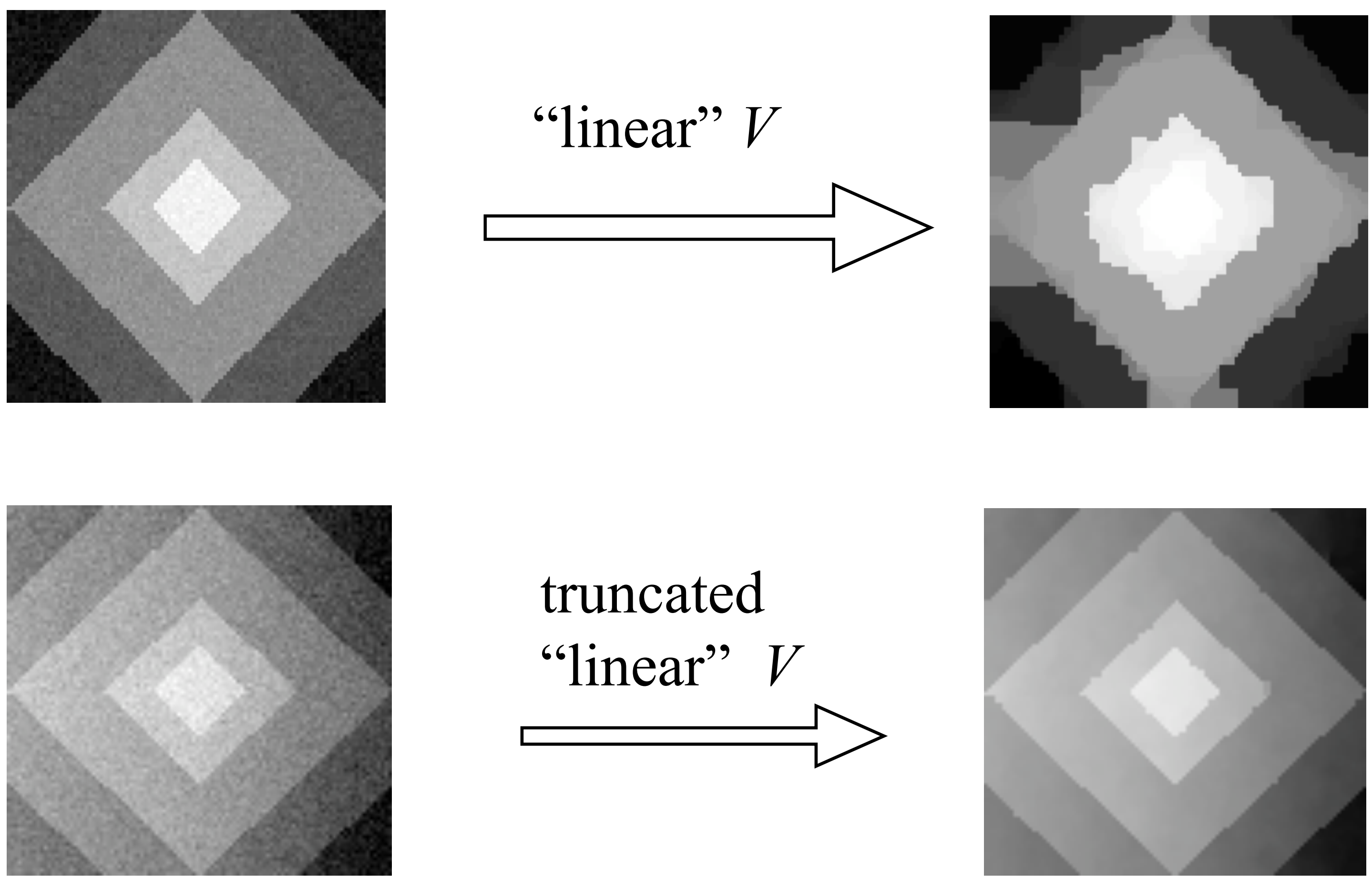


Robust
"discontinuity preserving"
interactions





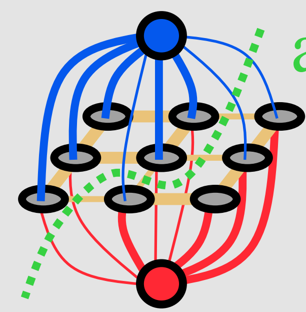
Pixel interactions:
"*convex*" vs. "*discontinuity-preserving*"



Graph cuts and Potts energy minimization

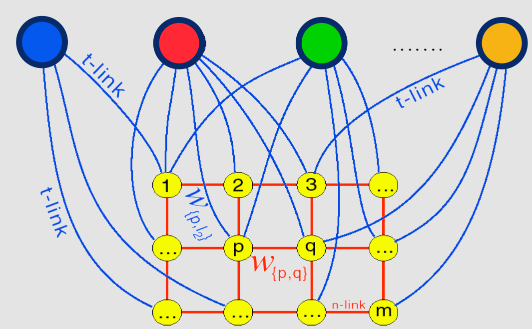
Binary Potts energy (Ising model)

_s-t_ graph cuts
(Greig at.al. 1989)



Multi-label Potts energy

Multi-way graph cuts
(BVZ 1998,2001)



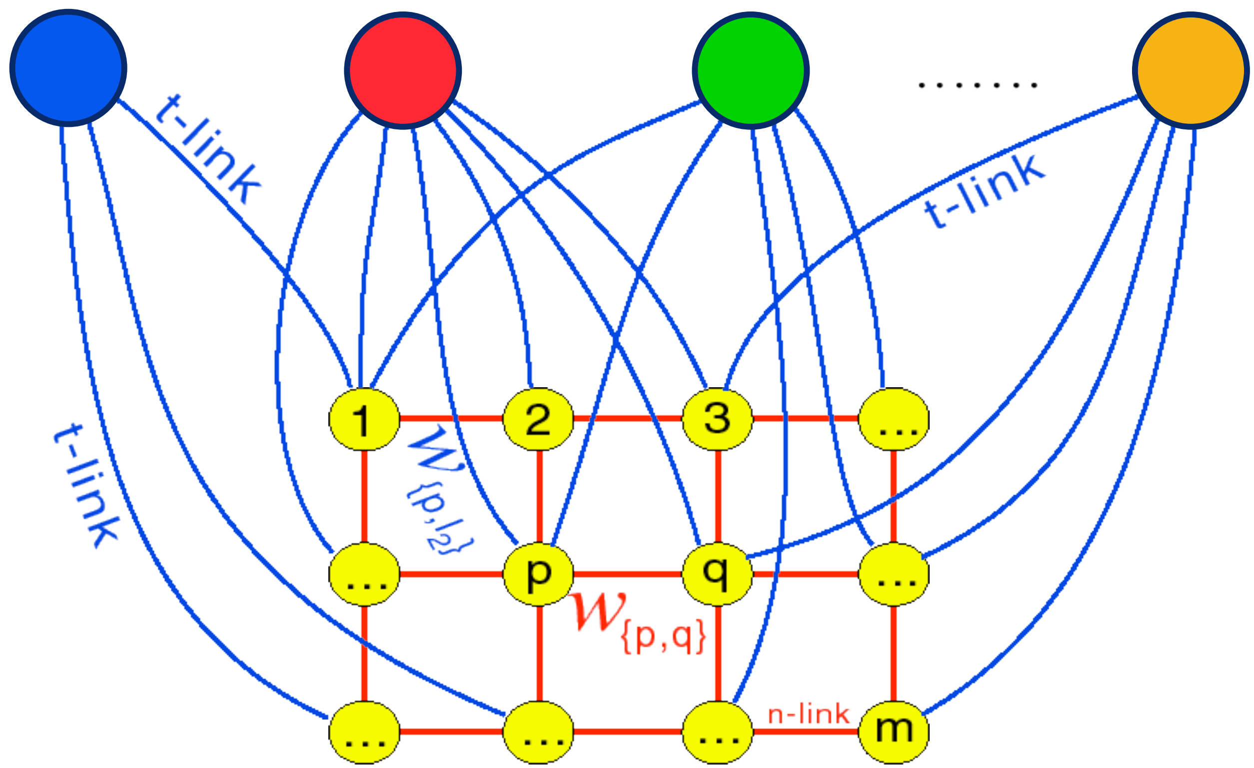
Graph cuts in computer vision:

Part I: *s-t* graph cuts

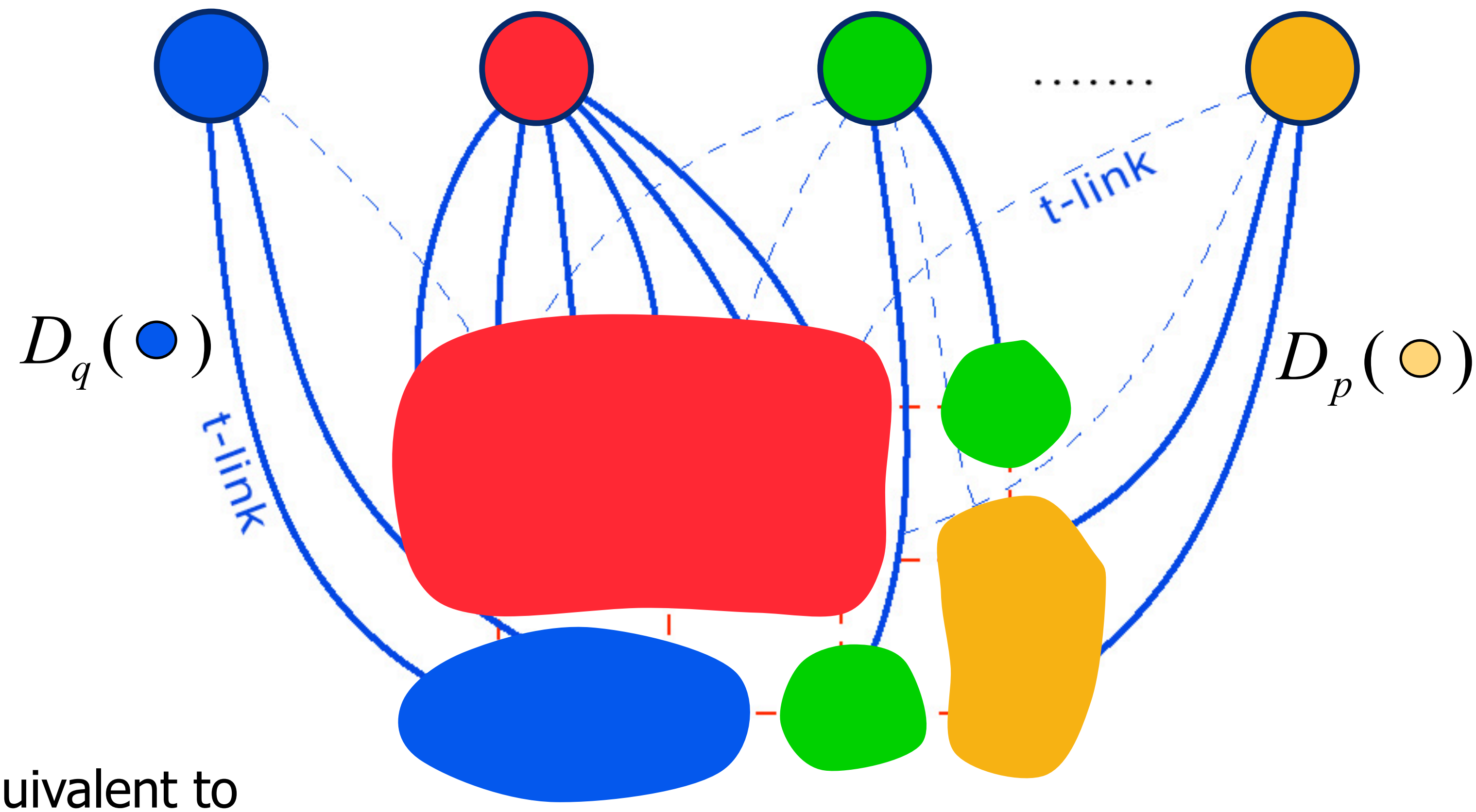
Part II: multi-way graph cuts

- applications
- general energy minimization tool

Multi-way graph cuts



Multi-way graph cuts

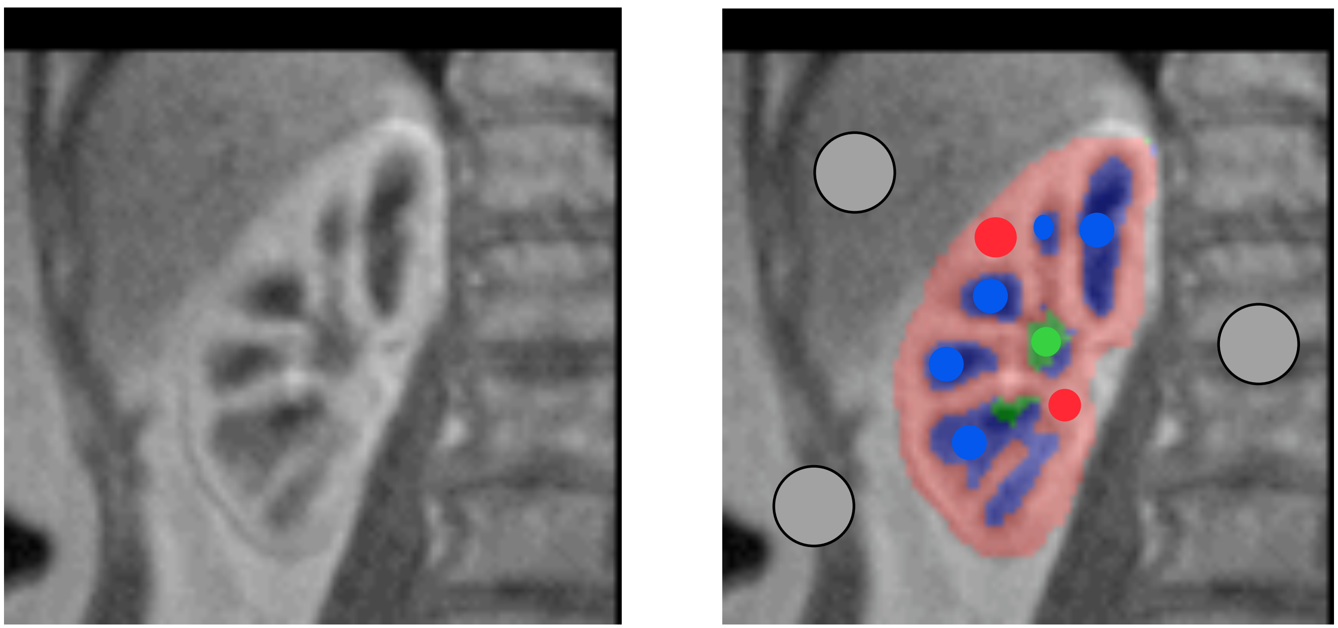


Equivalent to minimization of **the Potts energy** of labeling *L*

$$E(L) = \sum_{p} \underbrace{-D_p(L_p)}_{\text{t-links}} + \sum_{pq \in N} \underbrace{w_{pq} \cdot \delta_{Lp \neq Lq}}_{\text{n-links}}$$

Multi-way graph cuts

Multi-object Extraction



Obvious generalization of binary object extraction technique
(Boykov, Jolly, Funkalea 2004)

Multi-way graph cuts

stereo vision

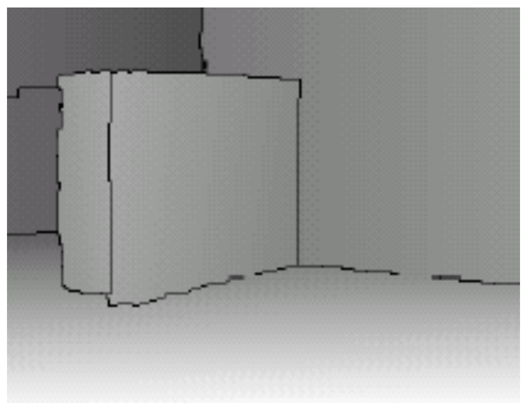
original pair of "stereo" images

depth map

Multi-way graph cuts

Stereo/Motion with slanted surfaces

(Birchfield &Tomasi 1999)



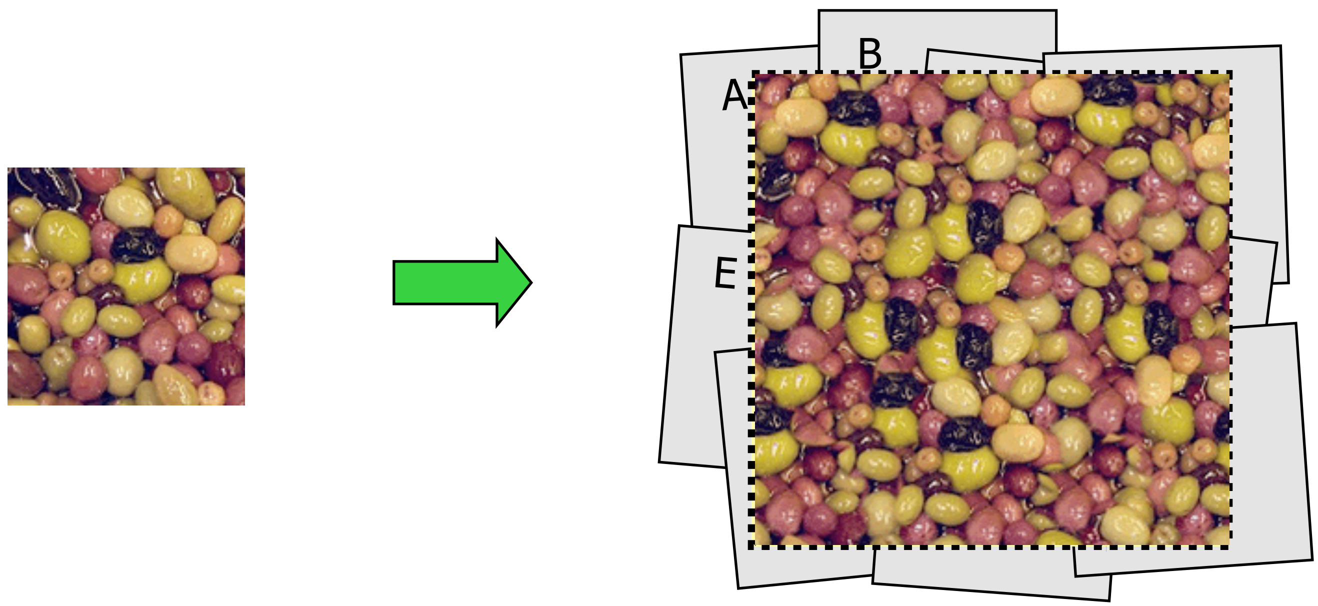
Labels = parameterized surfaces

EM based: E step = compute surface boundaries
M step = re-estimate surface parameters

Multi-way graph cuts

Graph-cut textures

(Kwatra, Schodl, Essa, Bobick 2003)



similar to "**image-quilting**" (Efros & Freeman, 2001)

Graph Cut Textures

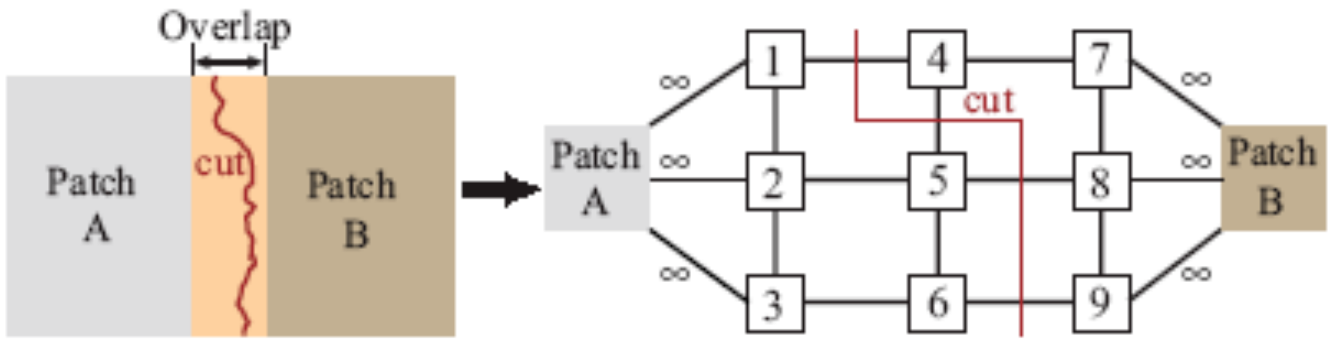


Figure 2: (Left) Schematic showing the overlapping region between two patches. (Right) Graph formulation of the seam finding problem, with the red line showing the minimum cost cut.

$$M(s,t,\mathbf{A},\mathbf{B}) = \|\mathbf{A}(s)-\mathbf{B}(s)\| + \|\mathbf{A}(t)-\mathbf{B}(t)\|$$

Graph Cut Textures

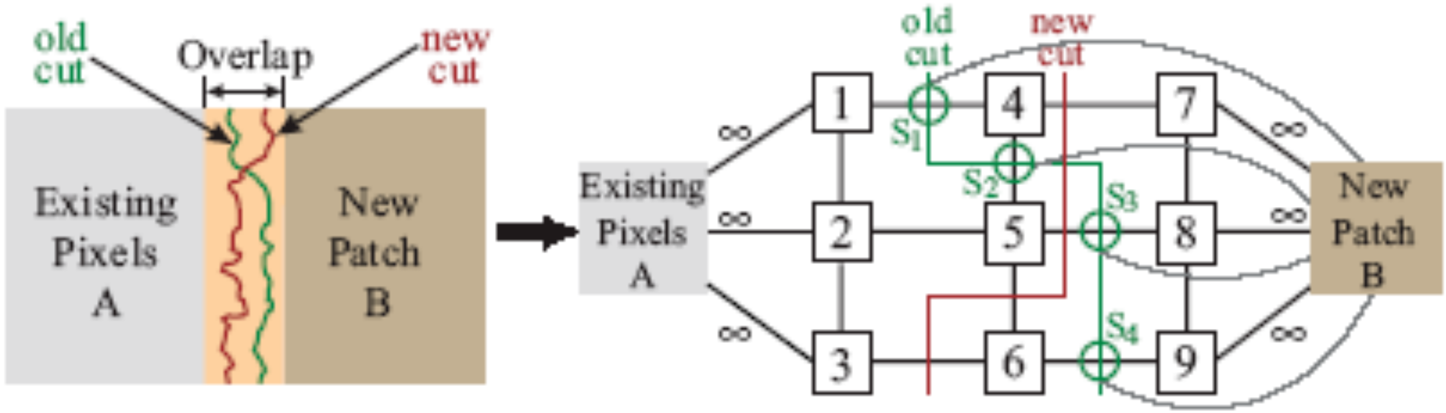
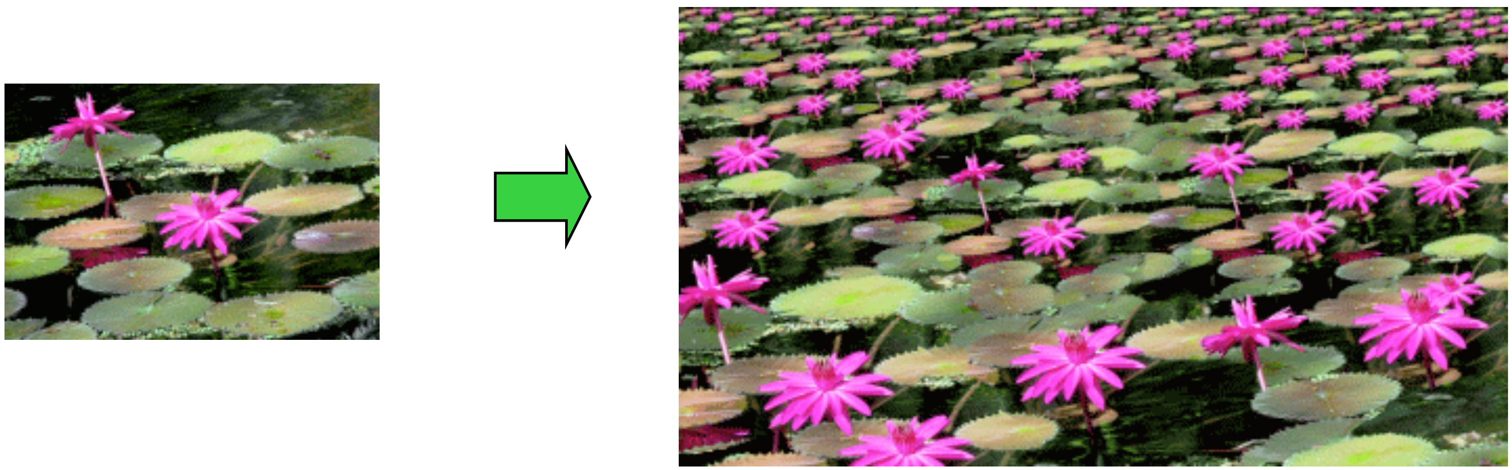


Figure 3: (Left) Finding the best new cut (red) with an old seam (green) already present. (Right) Graph formulation with old seams present. Nodes s_1 to s_4 and their arcs to **B** encode the cost of the old seam.

Multi-way graph cuts

Graph-cut textures

(Kwatra, Schodl, Essa, Bobick 2003)



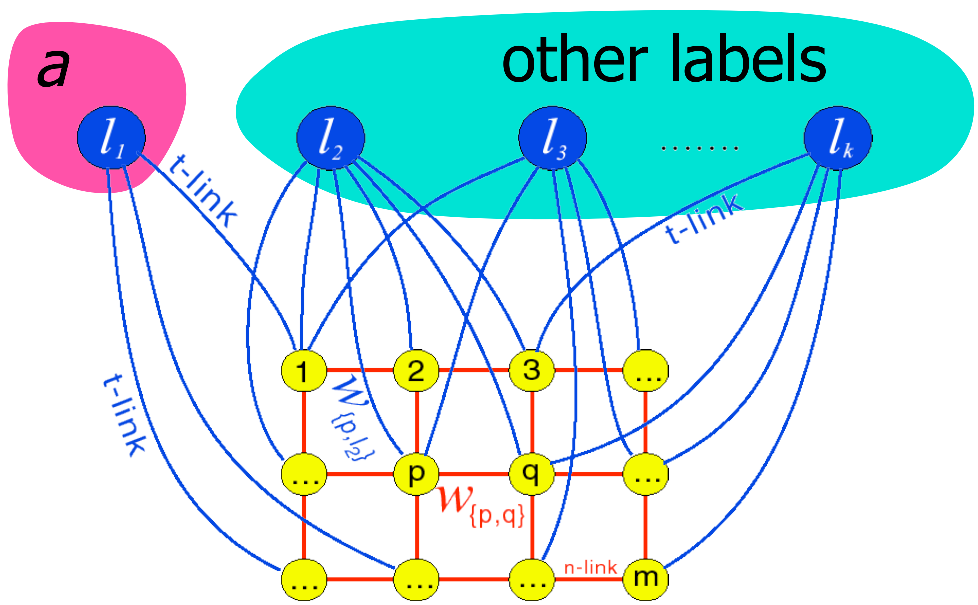
Multi-way graph cuts

- Equivalent to Potts energy minimization
- NP-hard problem (3 or more labels)
  - two labels can be solved via *s-t* cuts (Greig at. al., 1989)

- *a-expansion* approximation algorithm (BVZ 1998,2001)
  - guaranteed approximation quality
    - within a factor of 2 from the global minima (Potts model)
  - applies to a wide class of energies with robust interactions
    - Potts model (BVZ 1989)
    - "Metric" interactions (BVZ 2001)
    - "Submodular" interactions (KZ 2002,2004)

a-expansion move

Basic idea: break multi-way cut computation into a sequence of binary *s-t* cuts



Iteratively, each label competes with the other labels for space in the image

a-expansion moves

In each a-expansion a given label "a" grabs space from other labels

initial solution

● -expansion

● -expansion

● -expansion

● -expansion

● -expansion

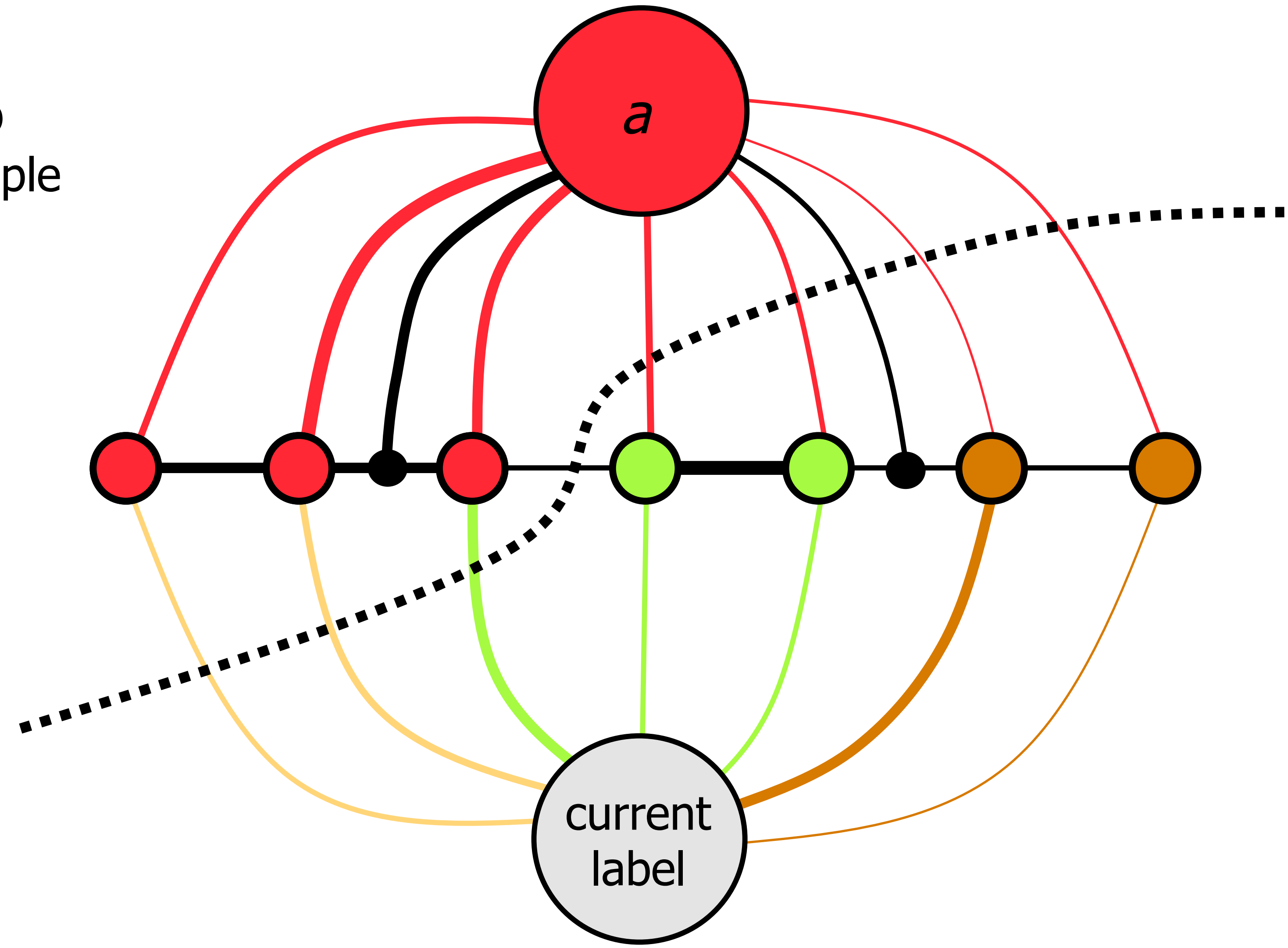
● -expansion

● -expansion

For each move we choose expansion that gives the largest decrease in the energy: **binary optimization problem**

Optimal *a*-expansion move via *s*-*t* graph cuts

1D example



a-expansion algorithm

1. Start with any initial solution
2. For each label "*a*" in any (e.g. random) order

    1. Compute optimal a-expansion move (s-t graph cuts)
    2. Decline the move if there is no energy decrease

3. Stop when no expansion move would decrease energy

a-expansions for energies with *metric* interactions

- *a*-expansion algorithm applies to pair-wise interactions that are *metrics* on the space of labels (BVZ, PAMI'01)

$$V(a,a) = 0$$

$$V(a,b) \geq 0$$

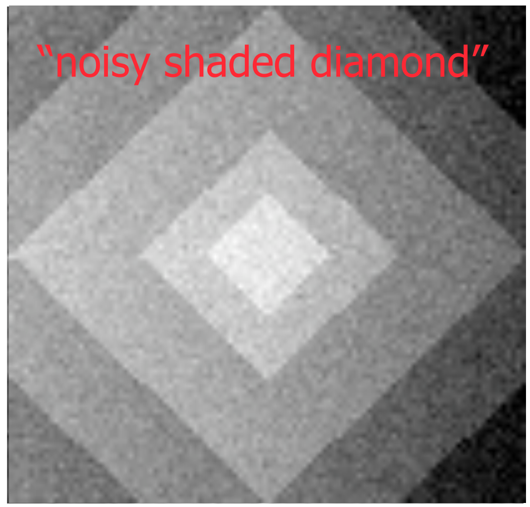
$$V(a,b) \leq V(a,c) + V(c,b)$$

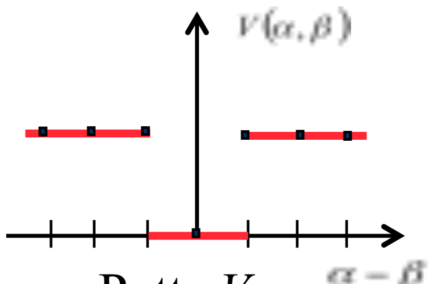
Triangular inequality

- *Example*: any truncated metric is also a metric

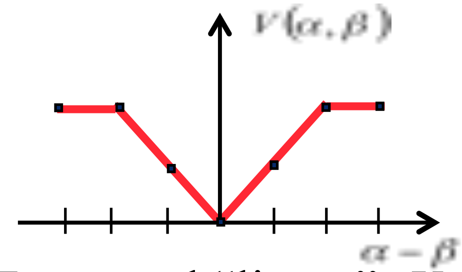
- => *Metric* case includes many **robust interactions**

a-expansions:
examples of *metric* interactions

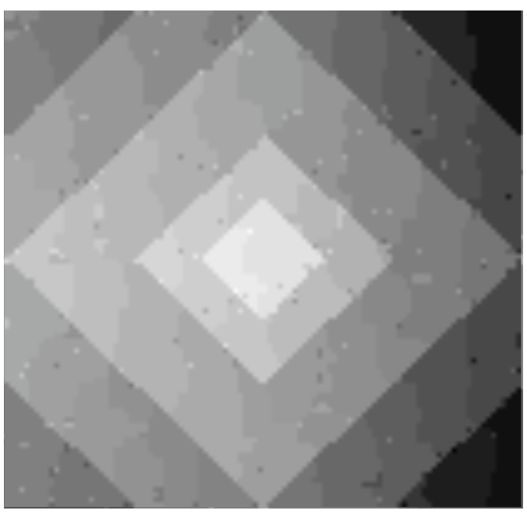


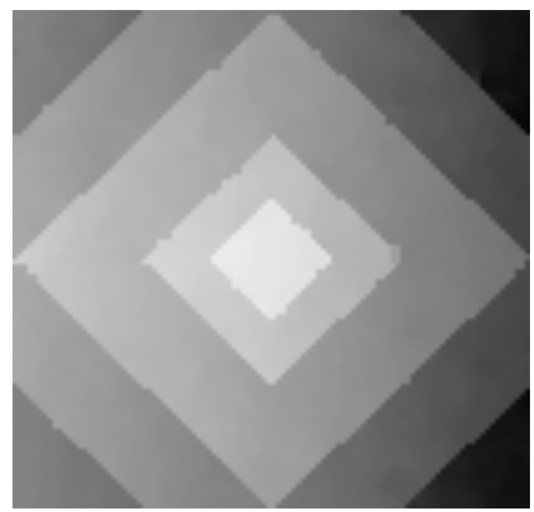


Potts V



Truncated "linear" V





a-expansions for energies with *submodular* interactions

- *a*-expansions algorithm further generalizes to *submodular* pair-wise interactions

$$V(c,c) + V(a,b) \leq V(a,c) + V(c,b)$$

- follows from complete characterization of binary energies that can be minimized via *s-t* graph cuts (KZ 2002, 2004)

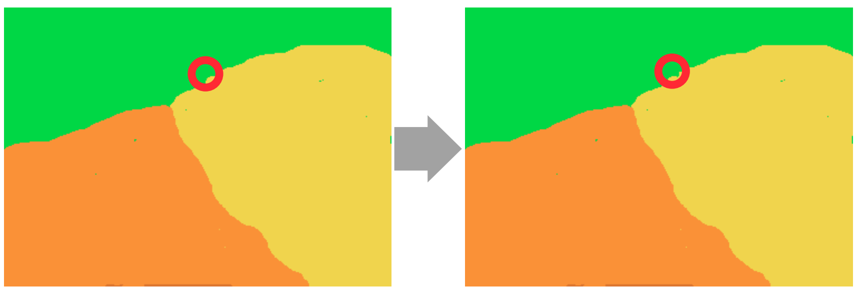
a-expansion algorithm

… can be compared to
general discrete energy minimization algorithms

(simulated annealing, ICM,…)

a-expansion algorithm vs. standard discrete energy minimization techniques

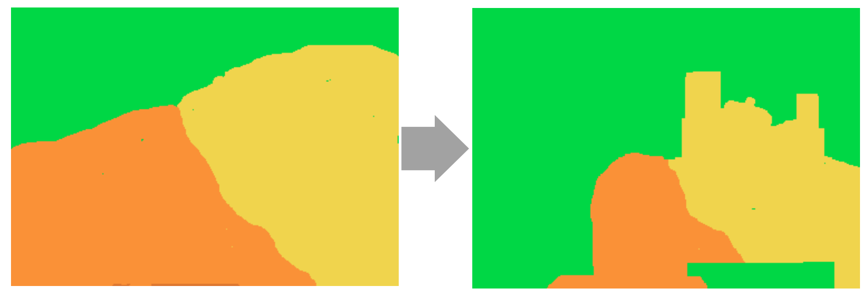
single "one-pixel" move
(simulated annealing, ICM,...)



- Only one pixel can change its label at a time

- Finding an optimal move is computationally trivial

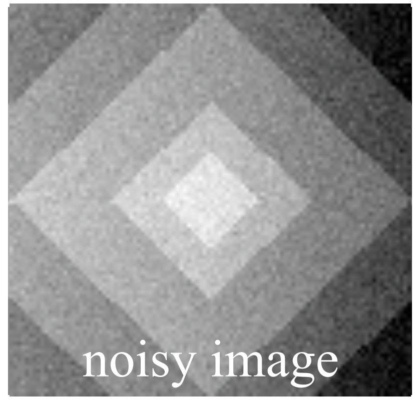
single *a*-expansion move



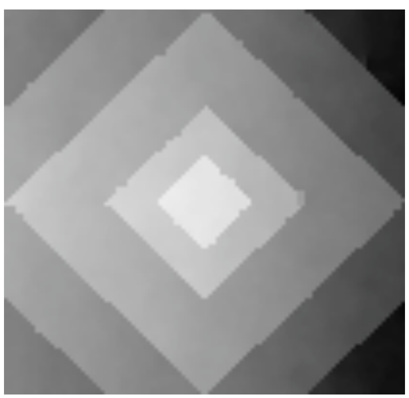
- Large number of pixels can change their labels simultaneously

- Finding an optimal move is computationally intensive O(2^n) (*s*-*t* cuts)

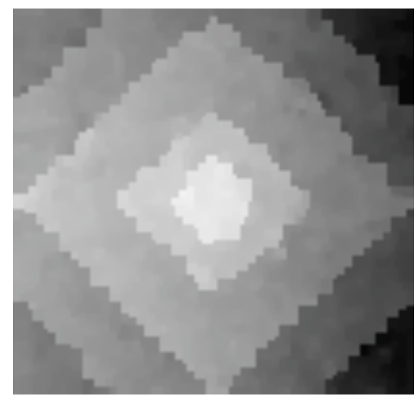
a-expansion move vs. “standard” moves



Potts energy minimization

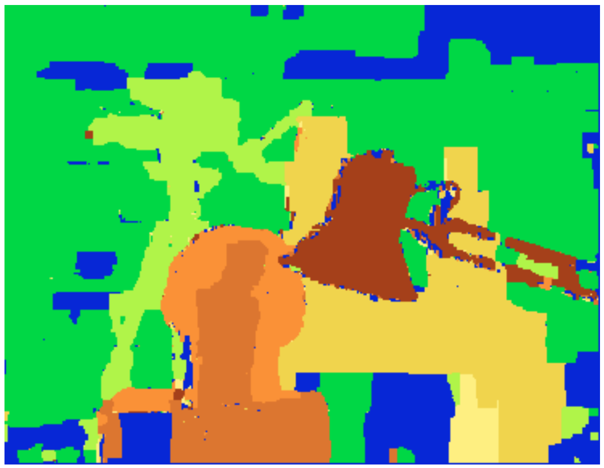


a local minimum
w.r.t. expansion moves



a local minimum
w.r.t. “one-pixel” moves

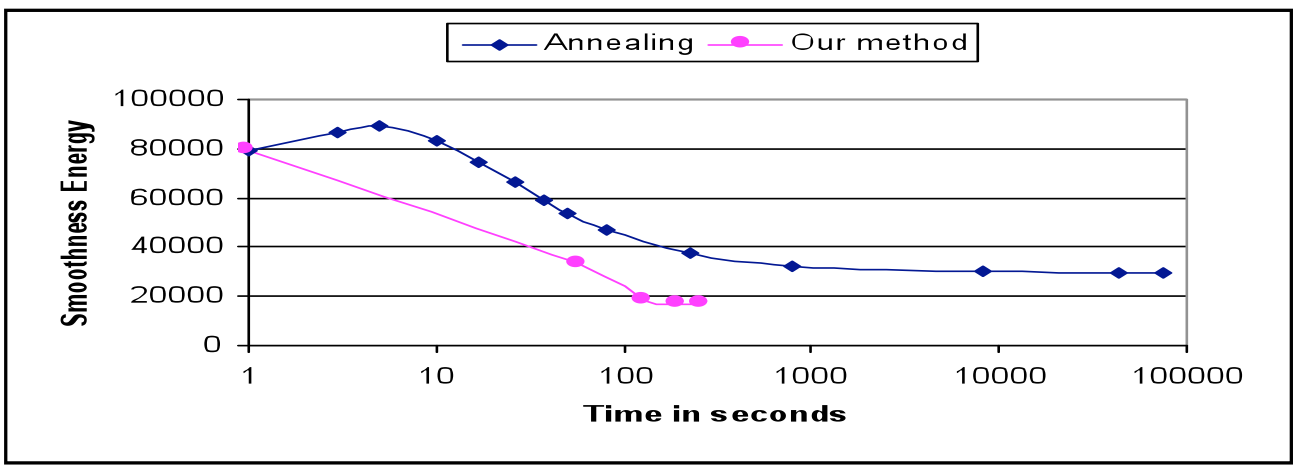
a-expansions vs. simulated annealing



normalized correlation,
start for annealing, 24.7% err

simulated annealing,
19 hours, 20.3% err

a-expansions (BVZ 89,01)
90 seconds, 5.8% err



α-expansion algorithm vs. local-update algorithms (SA, ICM, ...)

α-expansions	simulated annealing, ...
• Finds *local* minimum of energy with respect to very strong *moves*	• Finds *local* minimum of energy with respect to small "one-pixel" *moves*
• In practice, results do not depend on initialization	• Initialization is important
• solution is within the factor of 2 from the global minima	• solution could be arbitrarily far from the global minima
• In practice, one cycle through all labels gives sufficiently good results	• May not know when to stop. Practical complexity may be worse than exhaustive search
• Applies to a restricted class of energies	• Can be applied to anything

General low-level optimization methods strongly "competing" with Graph-Cuts

- Belief Propagation (also discrete optimization)
- Level-sets (variational/continuous optimization)

Geometry of s-t graph cuts on grids

- Boykov & Kolmogorov 2003
  - Minimum *s-t* cuts on an appropriately constructed grid-graphs approximate geodesics/minimum-surfaces in continuous Riemannian metric spaces
  - Graph cuts (BFJ'01,04) can be seen as discrete analogues of geometric contour/surface models (Caselles at.al.'97, Yezzi at.al'97)

s-t graph cuts vs. level sets

Level Sets [Osher&Sethian'88,…]	***Graph Cuts*** [Greig et. al.'89, Ishikawa et. al.'98, BVZ'98,…]
variational optimization method for fairly general continuous energies	combinatorial optimization for a restricted class of energies [e.g. KZ'02]
finds a local minimum near given initial solution	finds a global minimum for a given set of boundary conditions
numerical stability has to be carefully addressed [Osher&Sethian'88]: *continuous formulation -> "finite differences"*	numerical stability is not an issue *discrete formulation ->min-cut algorithms*
anisotropic metrics are harder to deal with (e.g. slower)	anisotropic Riemannian metrics are as easy as isotropic ones

Gradient descent method VS. Global minimization tool

(restricted class of energies)

Summary III (Graph Cuts)

- Graph Cuts vs Shortest Path
- Augmenting Path Algorithm
- Given an energy how to create the graph to be cut.
- Multi Way Cuts
- Dynamic Graph Cuts
- Examples

Comparison of graph cuts and belief propagation

Comparison of Graph Cuts with Belief Propagation for Stereo, using Identical MRF Parameters, ICCV 2003.
Marshall F. Tappen William T. Freeman

(a) Tsukuba Image (b) Graph Cuts (c) Synchronous BP (d) Accelerated BP

Figure 3. Results produced by the three algorithms on the Tsukuba image. The parameters used to generate this field were $s = 50$, $T = 4$, $P = 2$. Again, Graph Cuts produces a much smoother solution. Belief Propagation does maintain some structures that are lost in the Graph Cuts solution, such as the camera and the face in the foreground.

Ground truth, graph cuts, and belief propagation disparity solution energies

Image	Energy of MRF Labelling Returned ($\times 10^3$) Ground-Truth	Graph Cuts	Synchronous Belief Prop	% Energy from Occluded Matching Costs
Map	757	383	442	61%
Sawtooth	6591	1652	1713	79%
Tsukuba	1852	663	775	61%
Venus	5739	1442	1501	76%

Figure 2. Field Energies for the MRF labelled using ground-truth data compared to the energies for the fields labelled using Graph Cuts and Belief Propagation. Notice that the solutions returned by the algorithms consistently have a much lower energy than the labellings produced from the ground-truth, showing a mismatch between the MRF formulation and the ground-truth. The final column contains the percentage of each ground-truth solution's energy that comes from matching costs of occluded pixels.

Unifying View is the LP

- Both TRW (a modified form of BP) and
- PD (a modified form of a-expansion) optimize the following LP:

$$\sum x_i \theta_i + \sum x_{ij} \theta_{ij}$$

$$x_i = 0 \text{ or } 1$$

$$x_{ij} = 0 \text{ or } 1$$

Reduce, Reuse & Recycle
Efficiently Solving Multi-Label MRFs

Karteek Alahari Pushmeet Kohli* Philip H. S. Torr

Oxford Brookes University

*Now at Microsoft Research, Cambridge

Roadmap

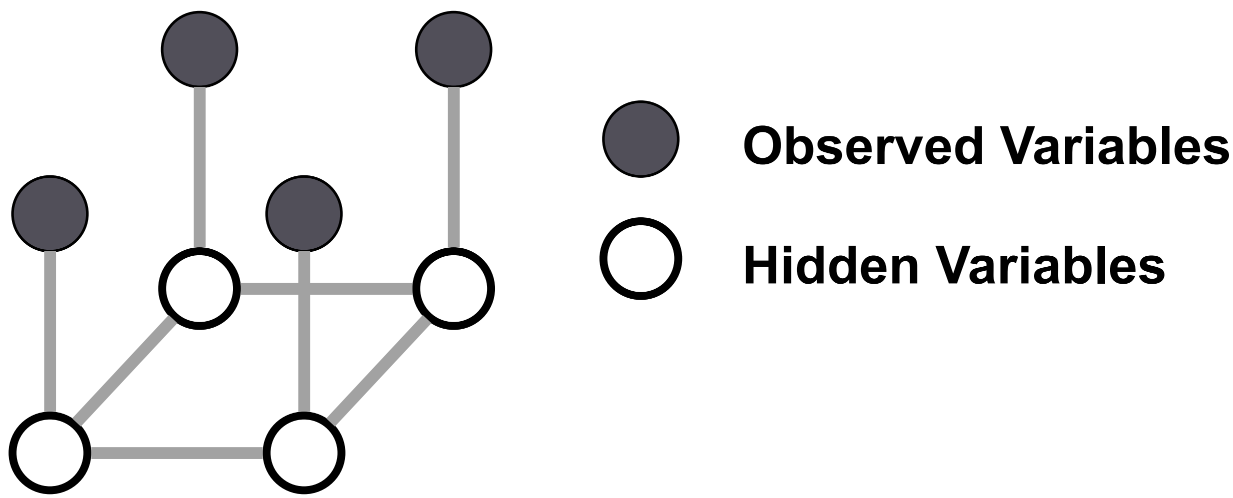
- Energy Functions on MRFs
- Solving MRFs
  - Exact Methods
  - Approximate Methods
- Efficient Energy Minimization
  - Recycling
  - Reducing
  - Reusing
- Results

Roadmap

- Energy Functions on MRFs
- Solving MRFs
  - Exact Methods
  - Approximate Methods
- Efficient Energy Minimization
  - Recycling
  - Reducing
  - Reusing
- Results

Energy Functions on MRFs

MAP Inference



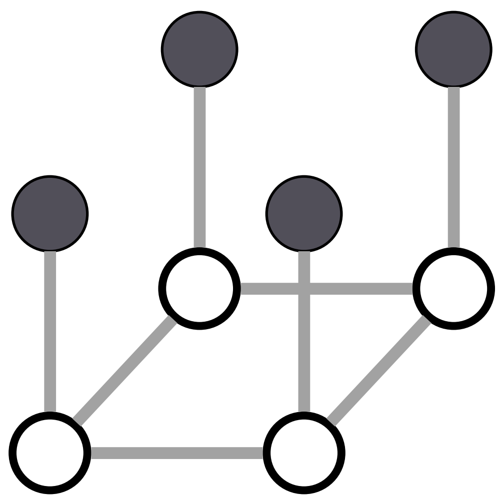
Courtesy: Pushmeet Kohli

Energy Functions on MRFs

MAP Inference

↕

Energy Minimization

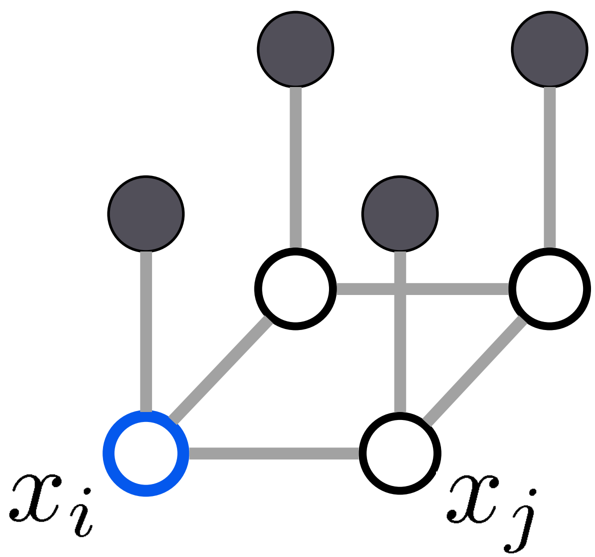


Energy Functions on MRFs

- **Pairwise** Energy Functions

$$E(x) = \boxed{\sum_{i \in \mathcal{V}} \psi_i(x_i)} + \sum_{i \in \mathcal{V}, j \in \mathcal{N}_i} \psi_{ij}(x_i, x_j)$$

Unary **Pairwise**

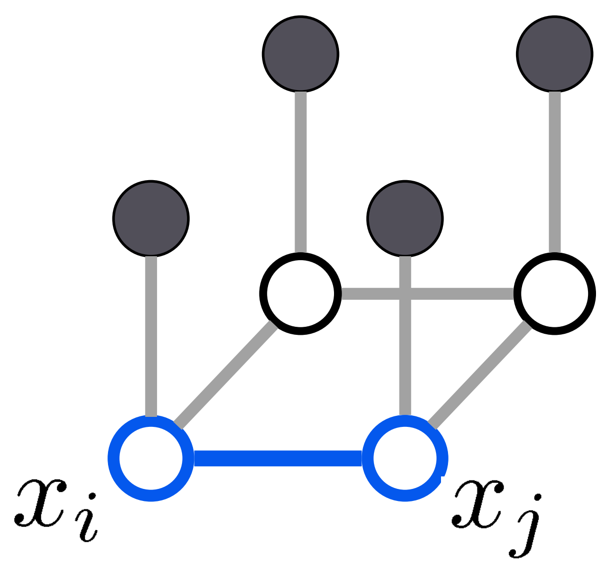


Energy Functions on MRFs

- **Pairwise** Energy Functions

$$E(x) = \sum_{i \in \mathcal{V}} \psi_i(x_i) + \boxed{\sum_{i \in \mathcal{V}, j \in \mathcal{N}_i} \psi_{ij}(x_i, x_j)}$$

Unary **Pairwise**



Roadmap

- Energy Functions on MRFs
- Solving MRFs
  - Exact Methods
  - Approximate Methods
- Efficient Energy Minimization
  - Recycling
  - Reducing
  - Reusing
- Results

Solving MRFs

- Minimizing a general MRF energy -- NP-hard
- Exact solutions exist for certain sub-classes
  - Graphs with no loops [Felzenszwalb & Huttenlocher '04]
  - Submodular energy functions [Ishikawa '03, Kolmogorov & Zabih '04, Schlesinger & Flach '06]

$$f(a,b) + f(a+1,b+1) \leq f(a,b+1) + f(a+1,b), \quad \forall a,b \in \mathcal{L}$$

- What about the rest?

For instance, Potts model

$$f(x_i, x_j) = \begin{cases} 0 & \text{if } x_i = x_j, \\ \gamma & \text{otherwise,} \end{cases}$$

is not submodular

Approximate Methods

- Move making algorithms [Boykov et al. ‘01]
  - Expansion
  - Swap
- Message passing algorithms
  - Belief propagation (BP) [Pearl ‘98]
  - Tree reweighted message passing (TRW) [Wainwright et al. ‘05, Kolmogorov ‘06]
  - Dual decomposition [Komodakis et al. ‘07]

Roadmap

- Energy Functions on MRFs
- Solving MRFs
  - Exact Methods
  - Approximate Methods
- Efficient Energy Minimization
  - Recycling
  - Reducing
  - Reusing
- Results

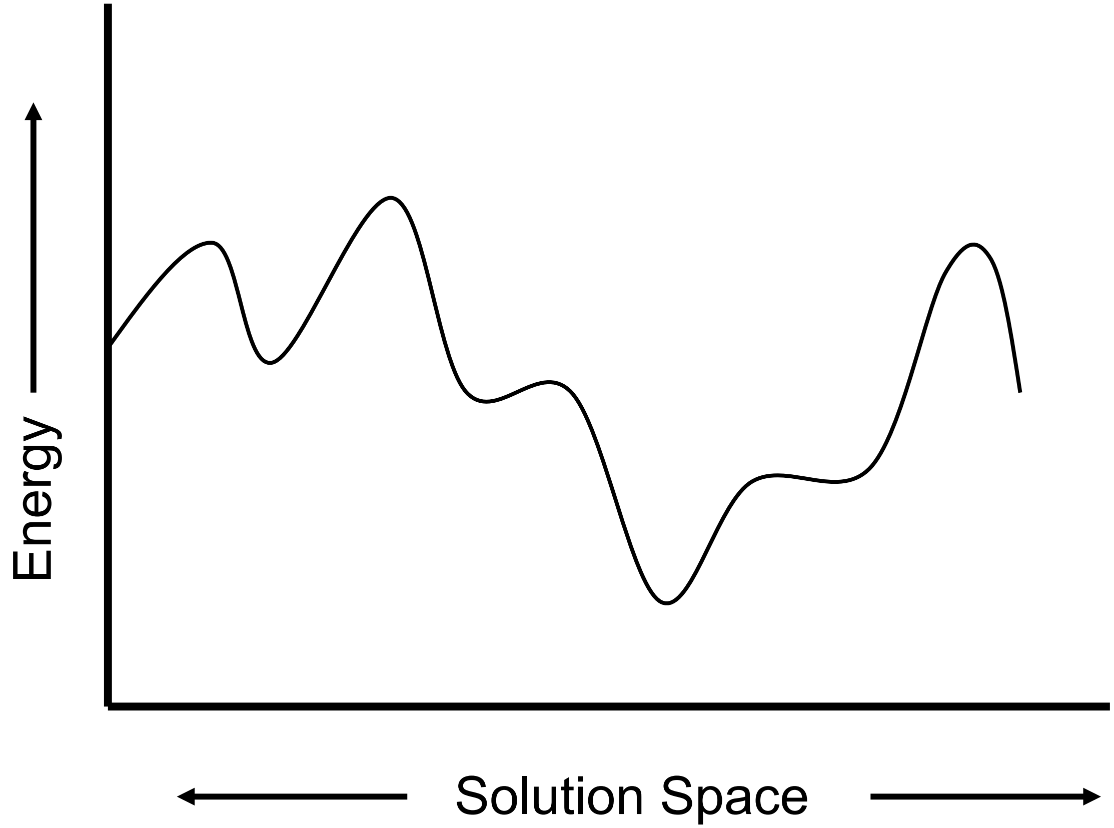
Efficient Energy Minimization

- Take considerable time for large problems
- Running time depends on
  - Initialization used for primal and dual variables
  - The number of variables in the problem
- Efficient methods do exist
  - Limited to submodular dynamic MRFs [Kohli & Torr '05, Juan & Boykov '06]
  - Fast-PD [Komodakis et al. '07]

Efficient Energy Minimization

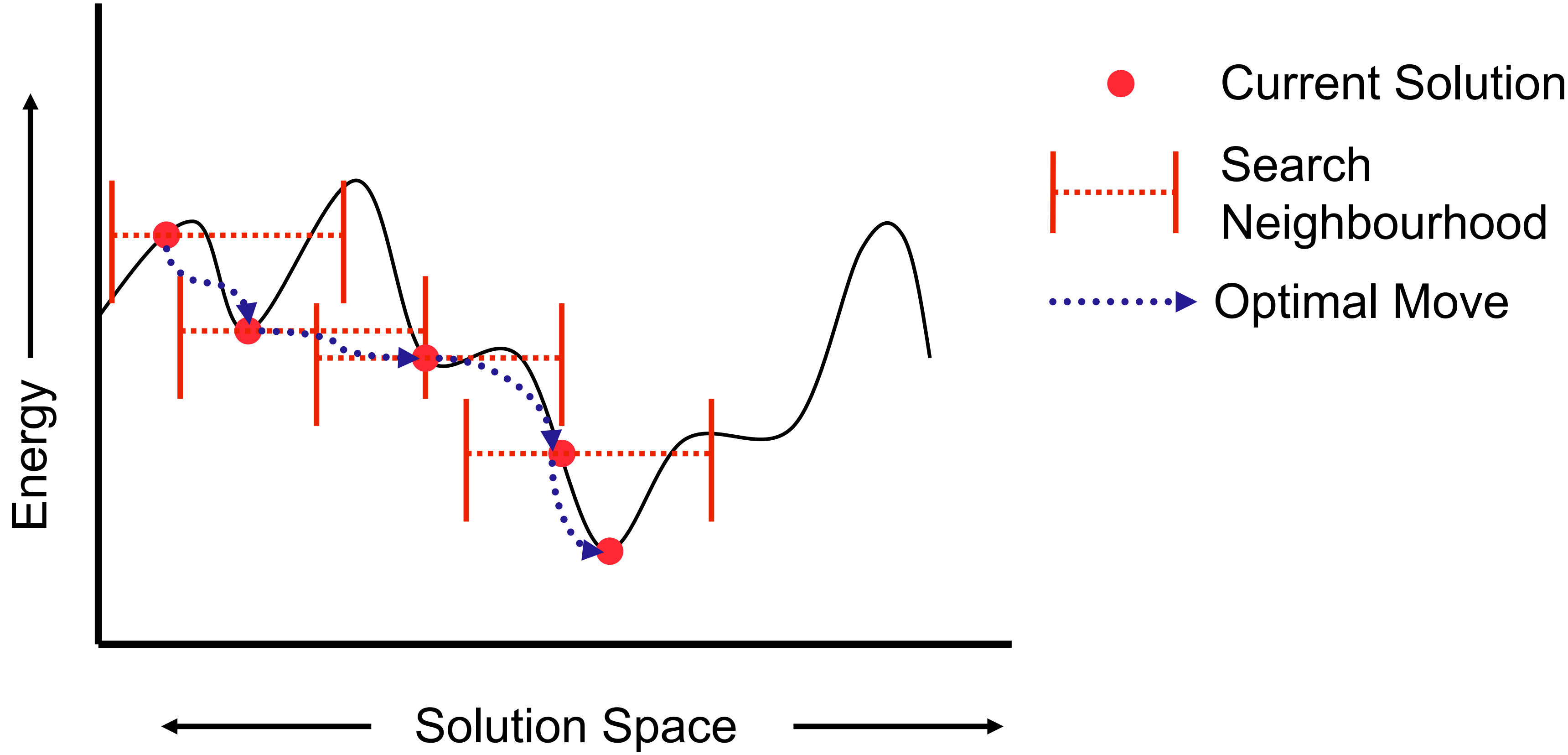
- Our primary goals are
  - To generate a good initialization for the current problem instance **(Recycle & Reuse)**
  - To reduce the number of variables involved in the energy function **(Reduce)**

Move Making Algorithms



Courtesy: Pushmeet Kohli

Move Making Algorithms



Example: α-expansion

Status:

Initialize with Tree / Expand Ground / Expand House / Expand Sky

- Tree
- Ground
- House
- Sky

Courtesy: Pushmeet Kohli

Example: α-expansion

- Variables take label α or retain current label $\bar{\alpha}$
- In one iteration, moves w.r.t. each label $\alpha \in \mathcal{L}$ are made
- Binary energy function for an α move is

$$E^\alpha(x^\alpha) = \sum_{i \in \mathcal{V}} \phi_i^\alpha(x_i^\alpha) + \sum_{i \in \mathcal{V}, j \in \mathcal{N}_i} \phi_{ij}^\alpha(x_i^\alpha, x_j^\alpha),$$

where $x_i^\alpha, x_j^\alpha \in \{0, 1\}$,

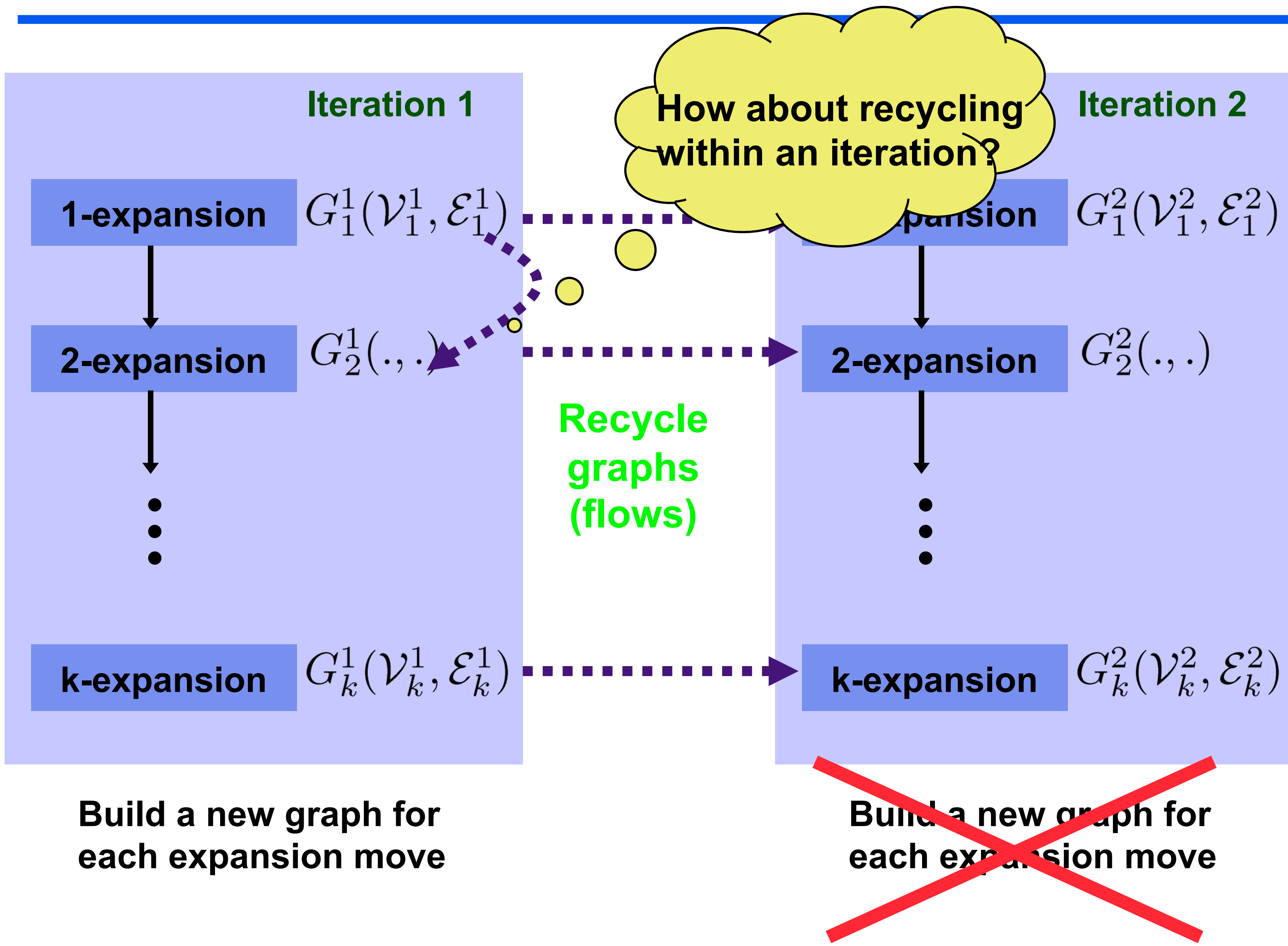
$$\phi_i^\alpha(x_i^\alpha) = \begin{cases} \phi_i(x_i = \alpha) & \text{if } x_i^\alpha = 0, \\ \phi_i(x_i = \bar{\alpha}) & \text{if } x_i^\alpha = 1, \end{cases}$$

$$\phi_{ij}^\alpha(x_i^\alpha, x_j^\alpha) = \begin{cases} 0 & \text{if } x_i^\alpha = 0, x_j^\alpha = 0, \\ \gamma(1 - \delta(x_i - x_j)) & \text{if } x_i^\alpha = 1, x_j^\alpha = 1, \\ \gamma & \text{otherwise.} \end{cases}$$

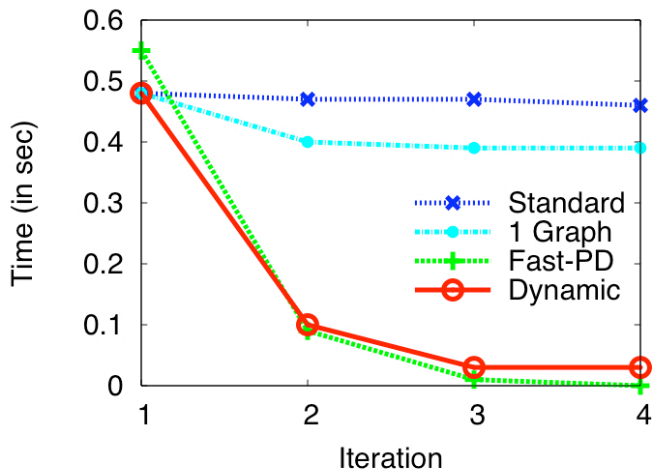
Example: α-expansion

- $E^{\alpha}(x^{\alpha})$ is submodular if $E(x)$ is metric
- Can be solved by the st-mincut/maxflow algorithm
- Primal solution -- labels assigned to x_i^{α}
- Dual solution -- feasible flow solution of the maxflow problem
- How to recycle results
  - Single MRF
  - Dynamic MRF

α-expansion: Single MRF

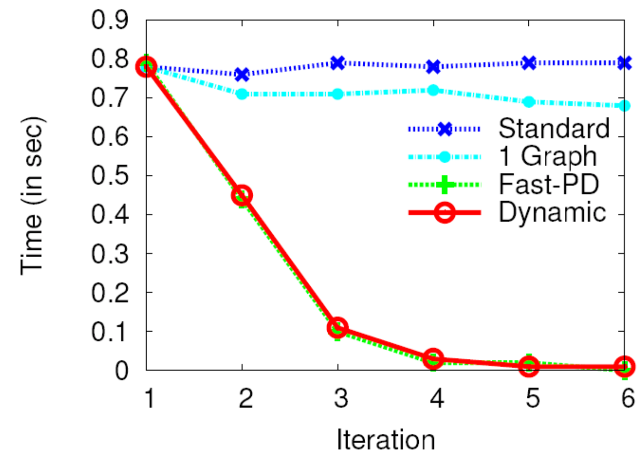


α-expansion: Single MRF



Object Segmentation

[Shotton et al. '06]



Stereo (Tsukuba)

Total times

Standard: 1.88s

1 Graph: 1.66s

Dynamic: 0.64s

Total times

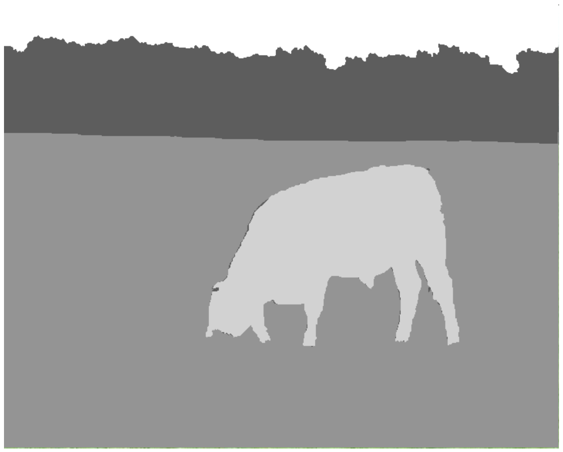
Standard 4.69s

1 Graph: 4.29s

Dynamic: 1.39s

α-expansion: Dynamic MRF

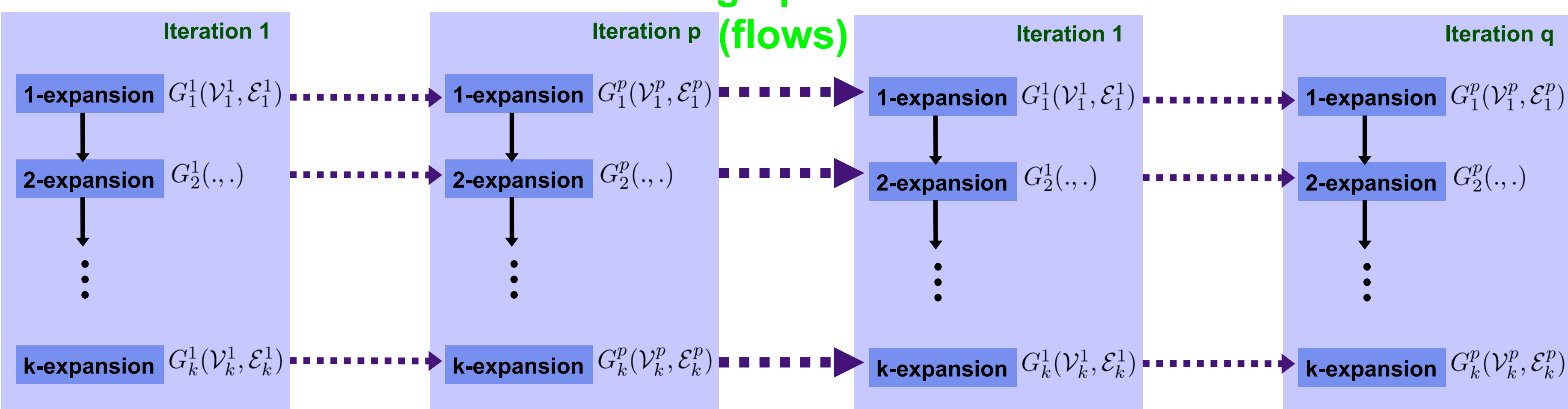
Frame t



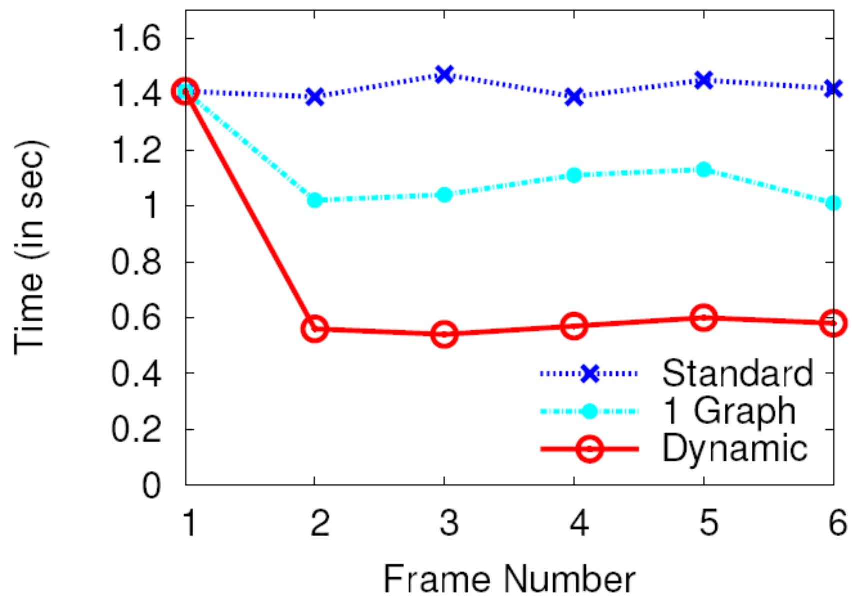
Frame t+1

Recycle Labeling

Recycle graphs (flows)



α-expansion: Dynamic MRF

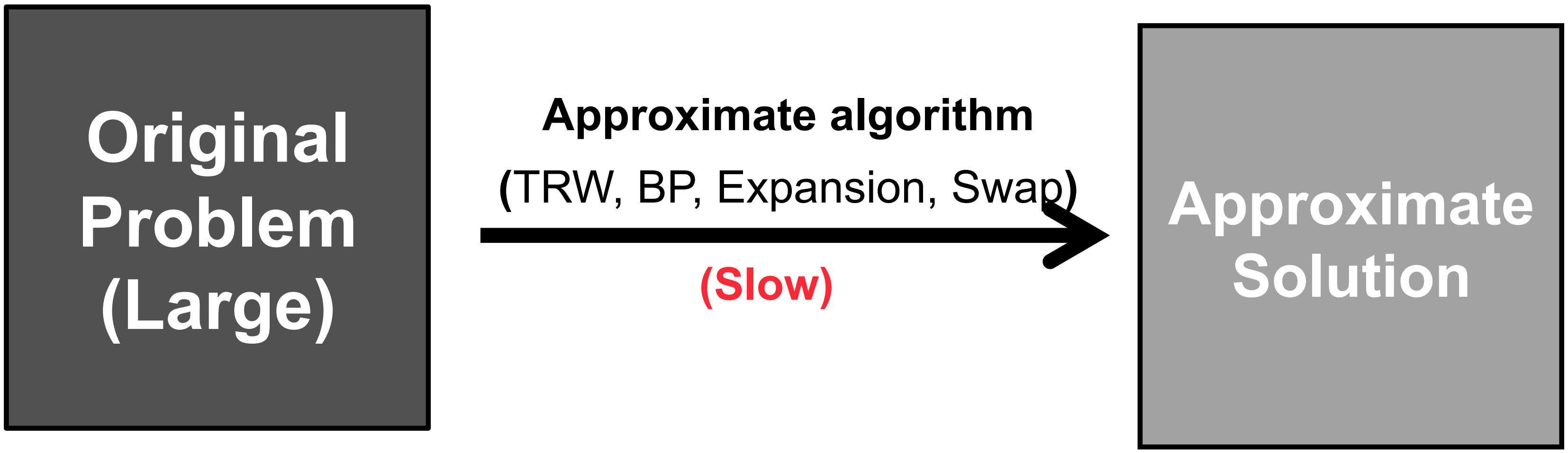


Multi-label Video Segmentation

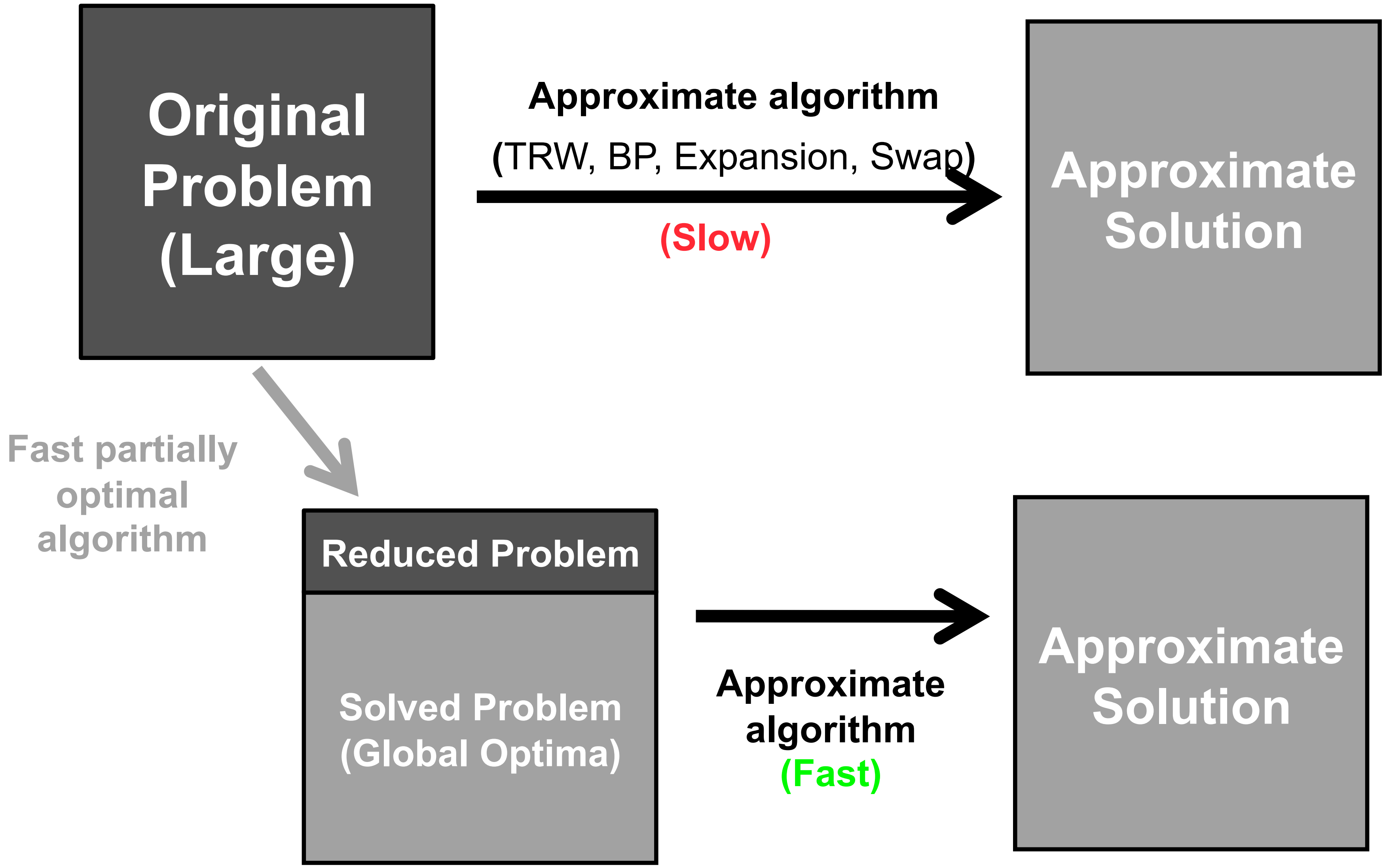
Roadmap

- Energy Functions on MRFs
- Solving MRFs
  - Exact Methods
  - Approximate Methods
- Efficient Energy Minimization
  - Recycling
  - Reducing
  - Reusing
- Results

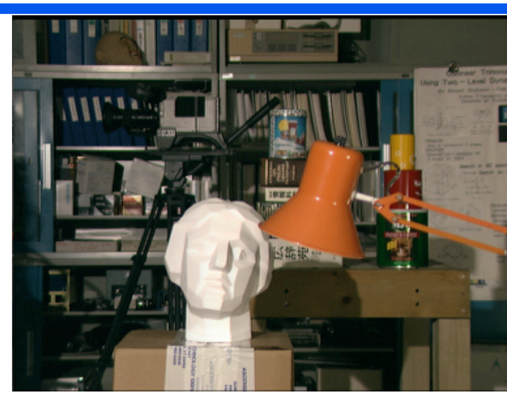
Reducing Energy Functions



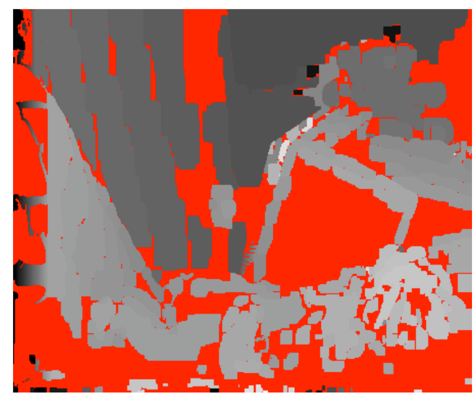
Reducing Energy Functions



Solved Problem : Examples







Partially Optimal Algorithm

- Construction of *k* auxiliary problems
- The energy corresponding to an auxiliary problem $\mathcal{P}_m$

$$E^m(\mathbf{x}^m) = \sum_{i \in \mathcal{V}} \phi_i^m(x_i^m) + \sum_{(i,j) \in \mathcal{E}} \phi_{ij}^m(x_i^m, x_j^m),$$

where $x_i^m, x_j^m \in \{0,1\}$.

$$\phi_i^m(x_i^m) = \begin{cases} \phi_i(x_i = l_m) & \text{if } x_i^m = 0, \\ \phi_i(x_i = l_{\min}) & \text{if } x_i^m = 1, \end{cases}$$

where $l_{\min} = \arg\min_{l \in \mathcal{L} - \{l_m\}} \phi_i(x_i = l)$

$$\phi_{ij}^m(x_i^m, x_j^m) = \begin{cases} 0 & \text{if } x_i^m = 0, x_j^m = 0, \\ 0 & \text{if } x_i^m = 1, x_j^m = 1, \\ \gamma & \text{otherwise.} \end{cases}$$

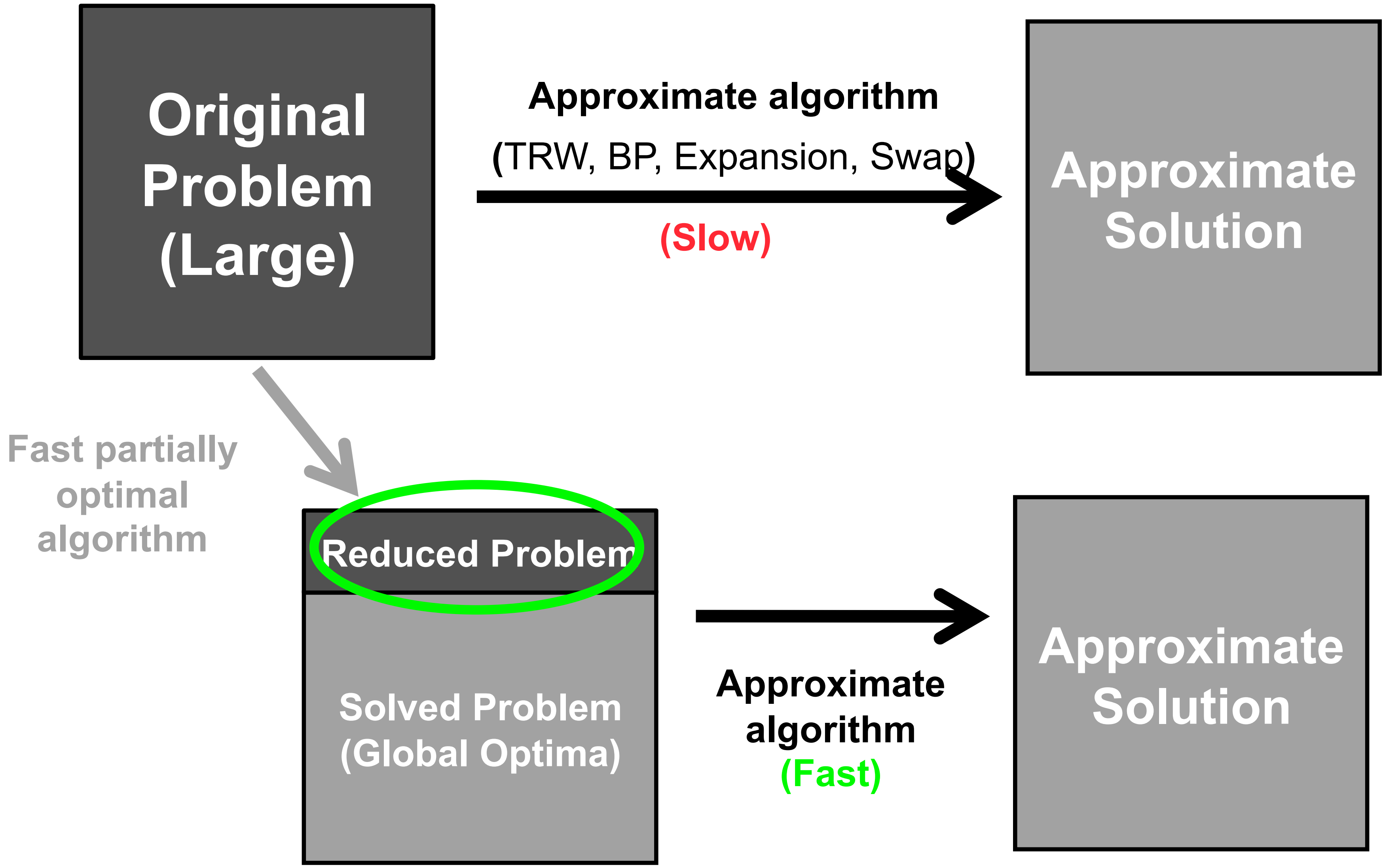
Partially Optimal Algorithm

- $E^m(\mathbf{x}^m)$ defines a submodular energy function
- The partially optimal solution of $E(\mathbf{x})$ w.r.t. $l_m \in \mathcal{L}$ is extracted as:

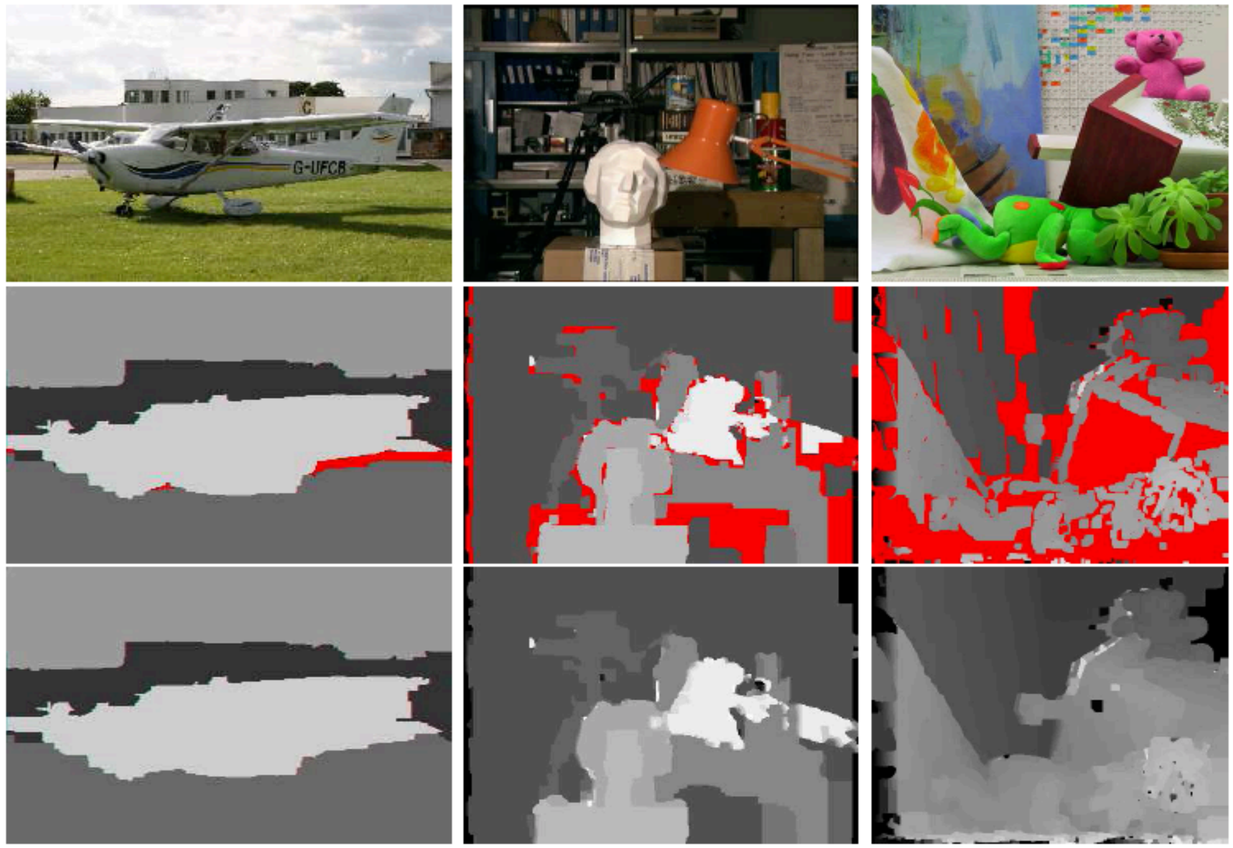
$$x_i = \begin{cases} l_m & \text{if } x_i^m = 0, \\ \epsilon & \text{otherwise.} \end{cases}$$

- Repeat this for all labels
- For further efficiency: *'Fix'* the partially optimal variables (Project)

Reducing Energy Functions



Reducing : Results



Image

Partial Labelling

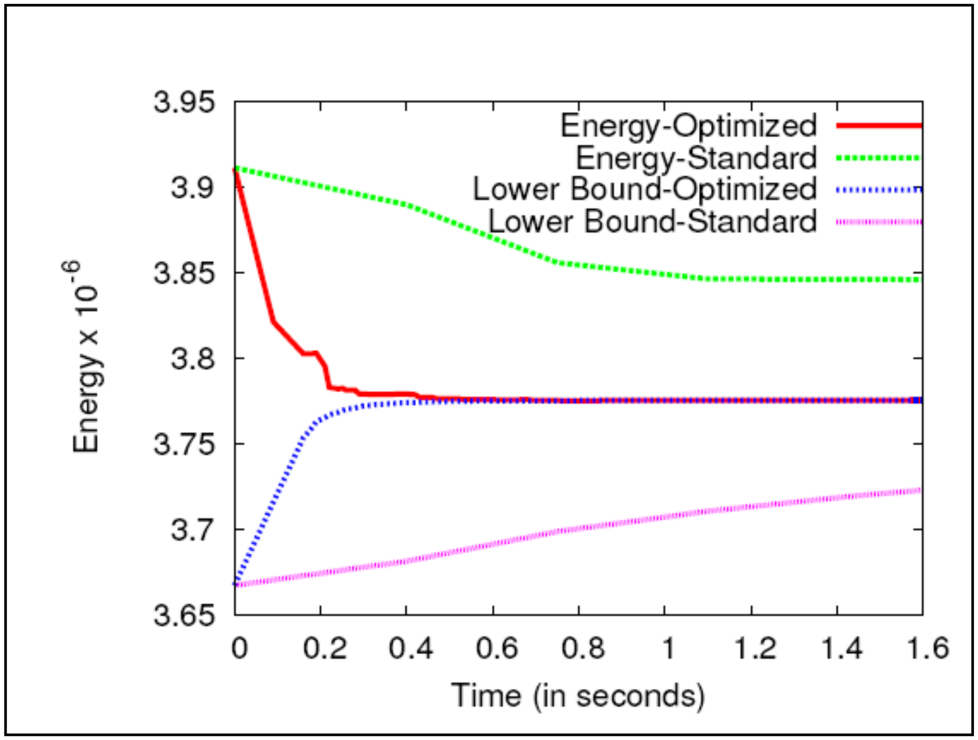
Partial Labelling + TRW

Times (s)						
	0.30	9.89	**4.67**	41.74	**63.77**	182.50

Reducing : Results

TRW-S

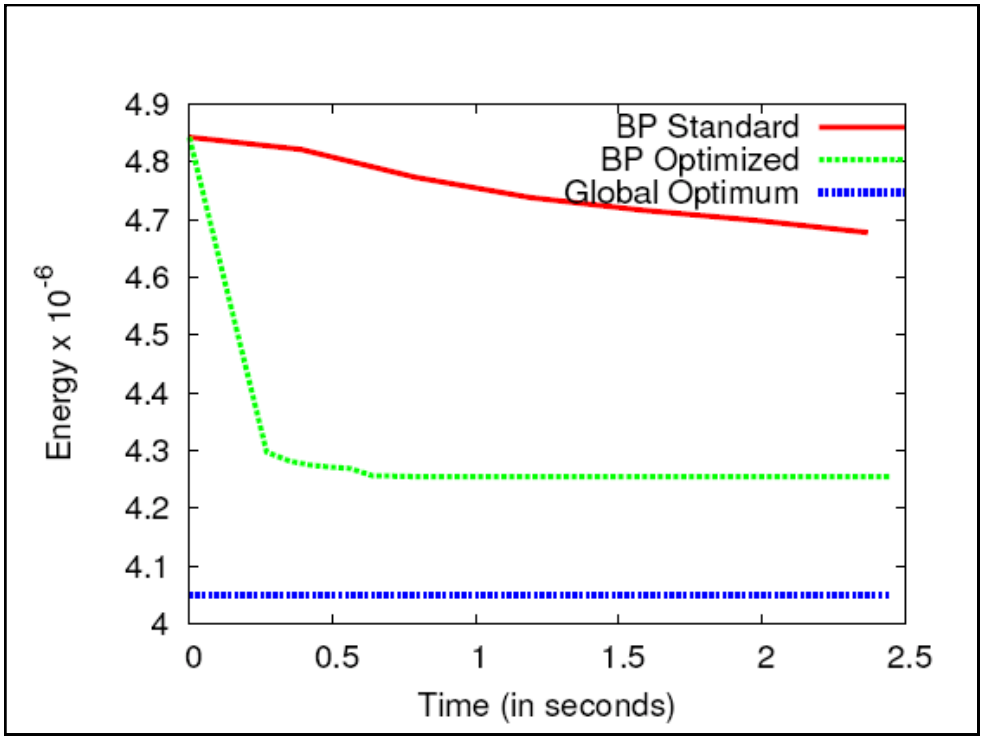
Object Segmentation
Problem
(Labels: 5)



Reducing : Results

BP

Object Segmentation Problem
(Labels: 7)



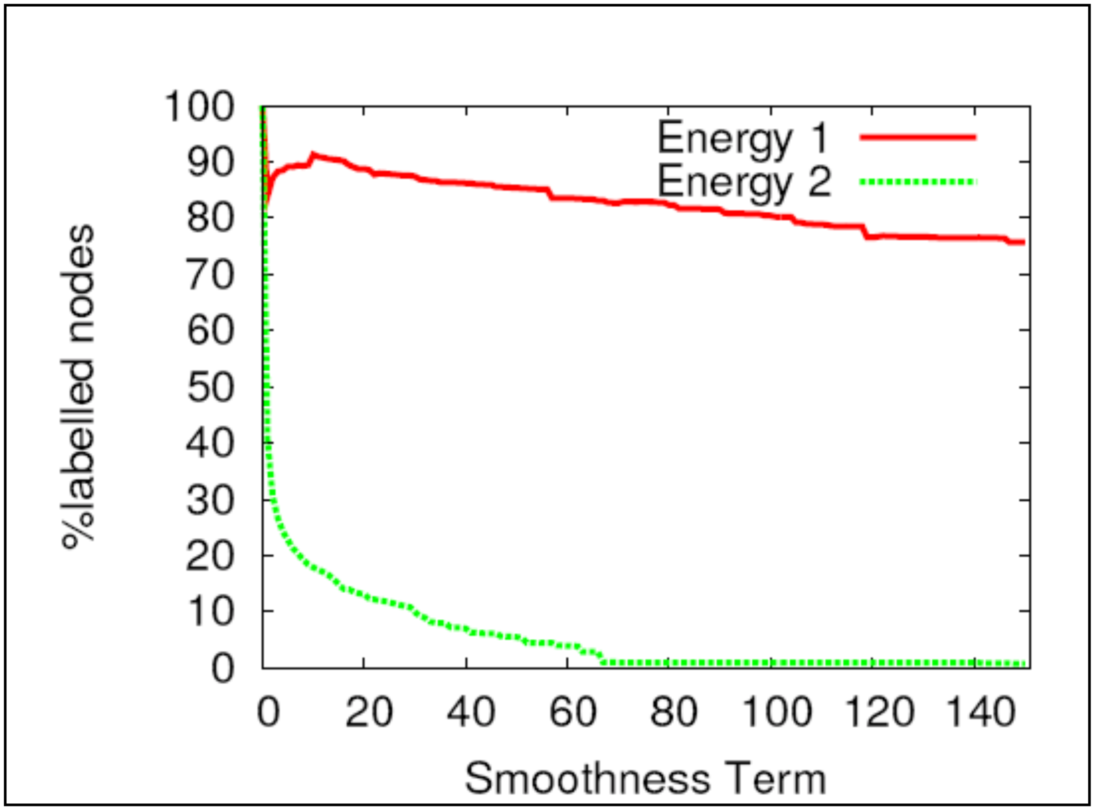
Performance of Par. Opt.

- Effect of increase in '*difficulty*'

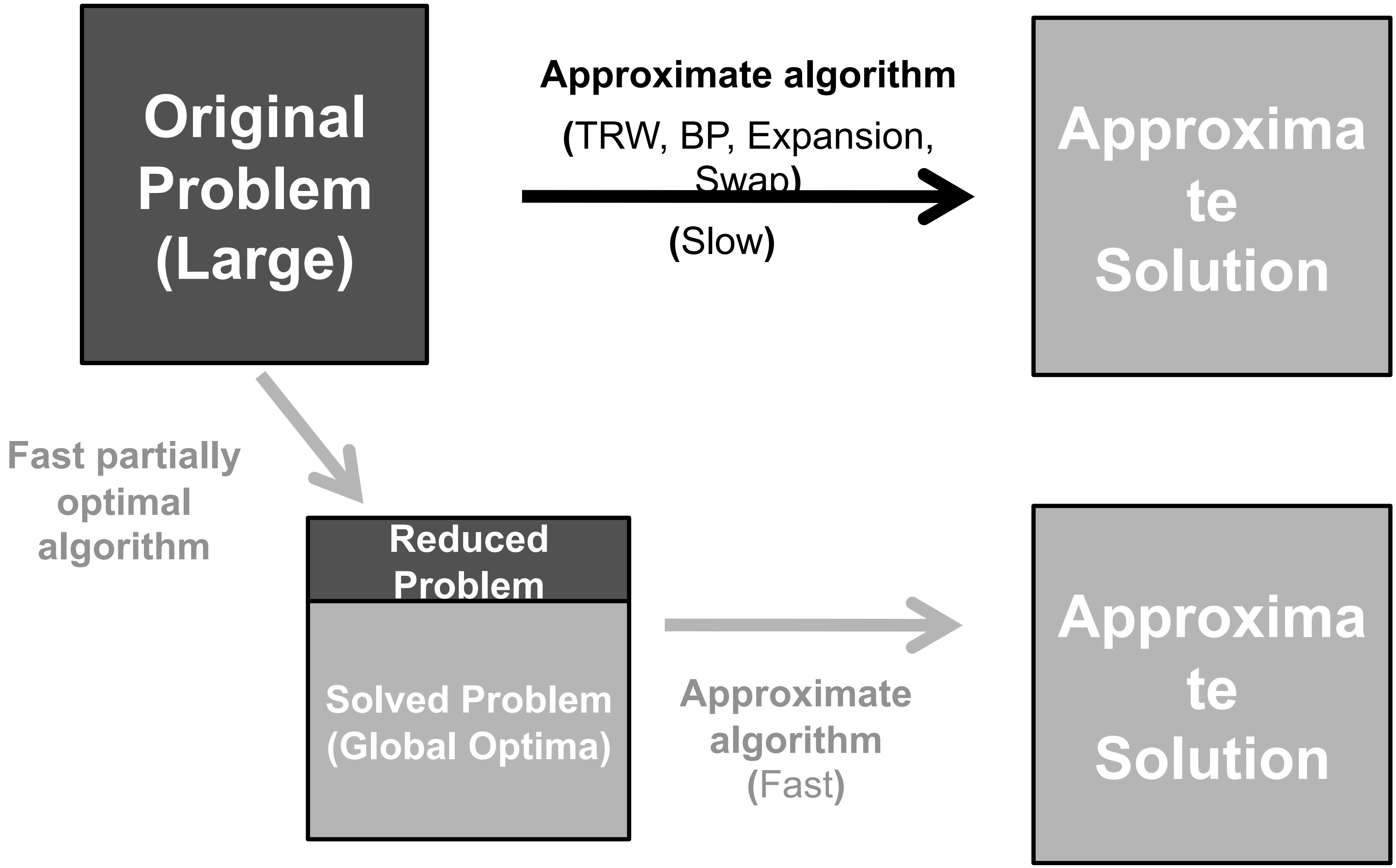
Stereo Problem

Energy 1:
Kovtun, DAGM'03

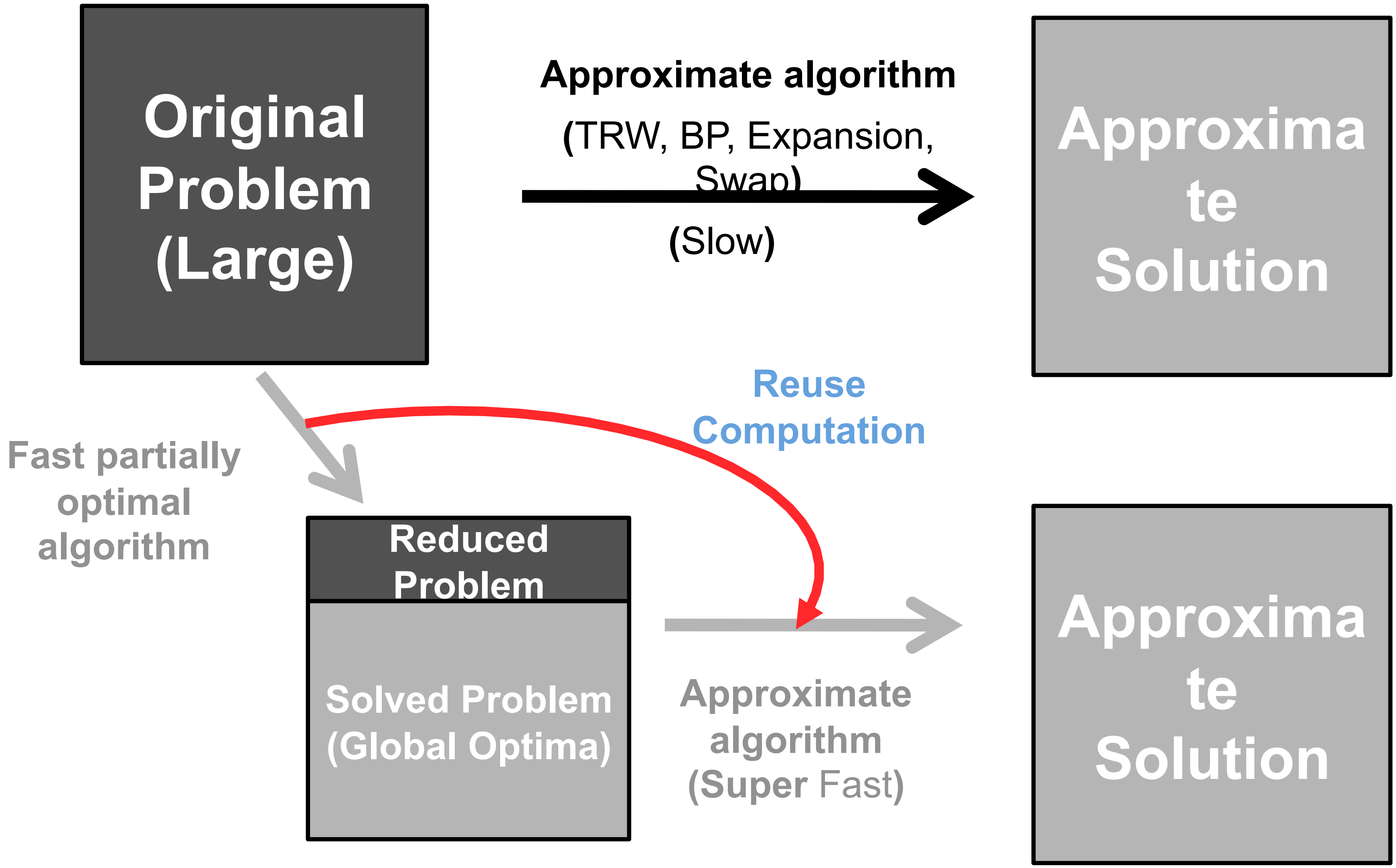
Energy 2:
Szeliski et al., ECCV'06



Reducing


Original Problem (Large)
Approximate algorithm
(TRW, BP, Expansion, Swap)
(Slow)
Approximate Solution
Fast partially optimal algorithm
Reduced Problem
Solved Problem (Global Optima)
Approximate algorithm
(Fast)
Approximate Solution

Reducing


Original Problem (Large)
Approximate algorithm
(TRW, BP, Expansion, Swap)
(Slow)
Approximate Solution
Fast partially optimal algorithm
Reuse Computation
Reduced Problem
Solved Problem (Global Optima)
Approximate algorithm (Super Fast)
Approximate Solution

Reusing Flow from PO

- Key Observation
  - Sub-problems of **PO** and **Expansion** have same form
  - Can be made similar by choosing particular starting configurations for expansions (Reuse flow)
  - Example: Potts (again)

Expansion

$$\phi_i^\alpha(x_i^\alpha) = \begin{cases} \phi_i(\alpha) & \text{if } x_i^\alpha = 0, \\ \phi_i(x_i) & \text{if } x_i^\alpha = 1, \end{cases}$$

$$\phi_{ij}^\alpha(x_i^\alpha, x_j^\alpha) = \begin{cases} 0 & \text{if } x_i^\alpha = 0, x_j^\alpha = 0, \\ \gamma\delta(x_i - x_j) & \text{if } x_i^\alpha = 1, x_j^\alpha = 1, \\ \gamma & \text{otherwise}, \end{cases}$$

PO auxiliary problem

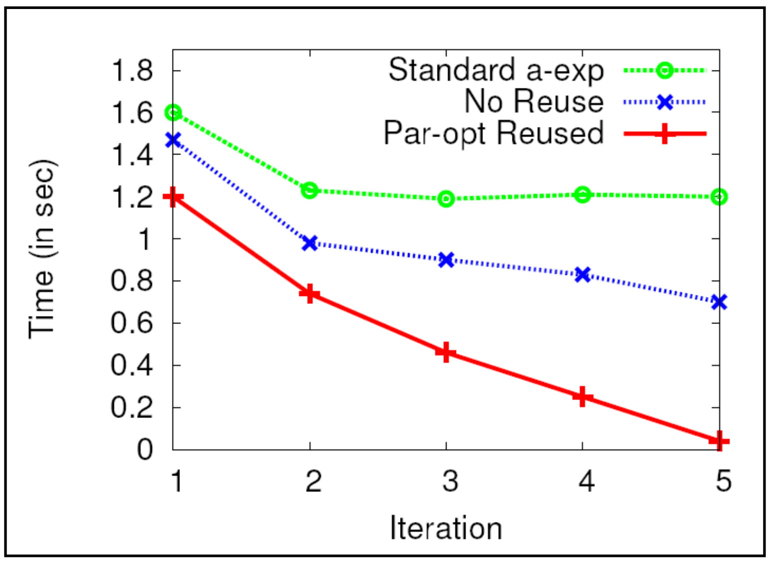
$$\phi_i^m(x_i^m) = \begin{cases} \phi_i(x_i = l_m) & \text{if } x_i^m = 0, \\ \phi_i(x_i = l_{\min}) & \text{if } x_i^m = 1, \end{cases}$$

$$l_{\min} = \arg\min_{l \in \mathcal{L} - \{l_m\}} \phi_i(x_i = l)$$

$$\phi_{ij}^m(x_i^m, x_j^m) = \begin{cases} 0 & \text{if } x_i^m = 0, x_j^m = 0, \\ 0 & \text{if } x_i^m = 1, x_j^m = 1, \\ \gamma & \text{otherwise}. \end{cases}$$

Reusing Flow from PO

- Key Observation
  - Sub-problems of **PO** and **Expansion** have same form
  - Can be made similar by choosing particular starting configurations for expansions (Reuse flow)
  - Example: Potts (again)



Reducing & Reusing : Results

	Time (in seconds)						
	α-exp	Fast-PD	opt α-exp	BP	opt BP	TRW-S	opt TRW-S
Colour-based Segmentation:							
Cow (3)	2.53	1.31	**0.21**	95.93	**0.32**	98.36	**0.33**
Cow (4)	3.75	1.72	**0.38**	108.32	**0.42**	111.69	**0.43**
Garden (4)	0.28	0.14	**0.04**	5.59	**0.17**	5.89	**0.21**
Stereo:							
Tsukuba (16)	5.74	1.47	**0.84**	38.19	**4.47**	41.74	**4.67**
Venus (20)	11.87	3.07	**3.03**	67.04	**14.97**	71.46	**16.02**
Cones (60)	42.23	9.48	**4.36**	173.35	**29.41**	182.66	**30.70**
Teddy (60)	44.25	9.56	**8.27**	172.30	**60.35**	182.50	**63.77**
Texture-based Segmentation:							
Plane (4)	0.39	0.35	**0.15**	9.41	**0.29**	9.89	**0.30**
Bikes (5)	0.82	0.54	**0.22**	10.69	**0.64**	11.19	**0.70**
Road (5)	0.91	0.51	**0.18**	10.67	**0.60**	11.26	**0.62**
Building (7)	1.32	0.89	**0.38**	12.70	**2.57**	13.52	**2.66**
Car (8)	0.99	0.53	**0.11**	13.68	**0.23**	14.42	**0.24**

P^3 & Beyond

Solving Energies with Higher Order Cliques

Pushmeet Kohli **Pawan Kumar** **Philip H. S. Torr**

Oxford Brookes University

Talk Overview

- **Higher Order Energy Functions**
- **Optimization Algorithms**
- **Move making Algorithms**
- **Optimal moves for Higher Order Energies**
- **P^N Potts Model**
- **Experiments**

Talk Overview

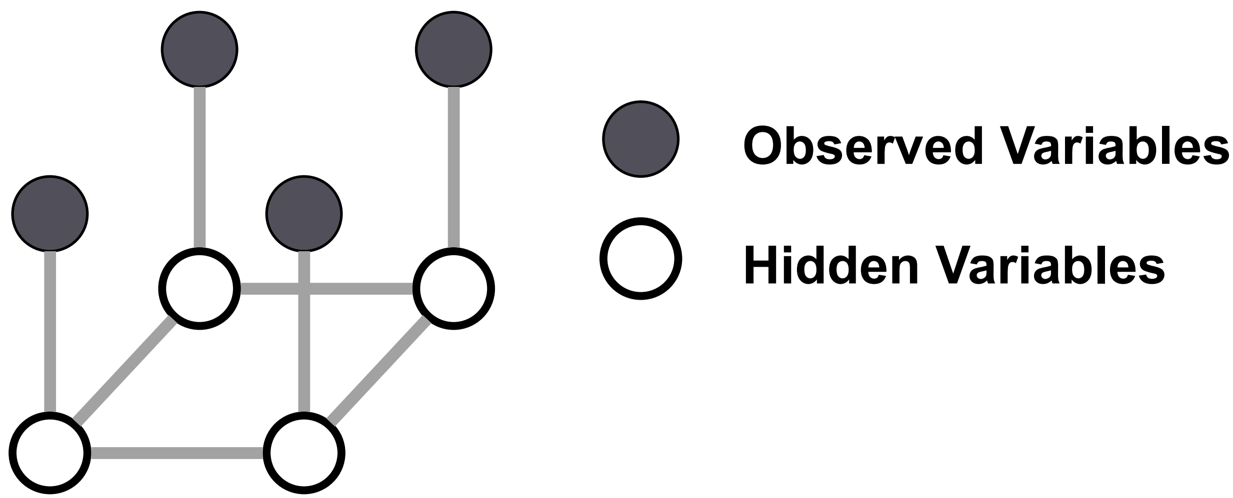
- Higher Order Energy Functions
- Optimization Algorithms
- Move making Algorithms
- Optimal moves for Higher Order Energies
- P^N Potts Model
- Experiments

Pairwise Energy Functions

$$E(\mathbf{x}) = \sum_i \psi_i(x_i) + \sum_{i \in \mathcal{V}, j \in \mathcal{N}_i} \psi_{ij}(x_i, x_j)$$

Unary **Pairwise**

Markov Random Field

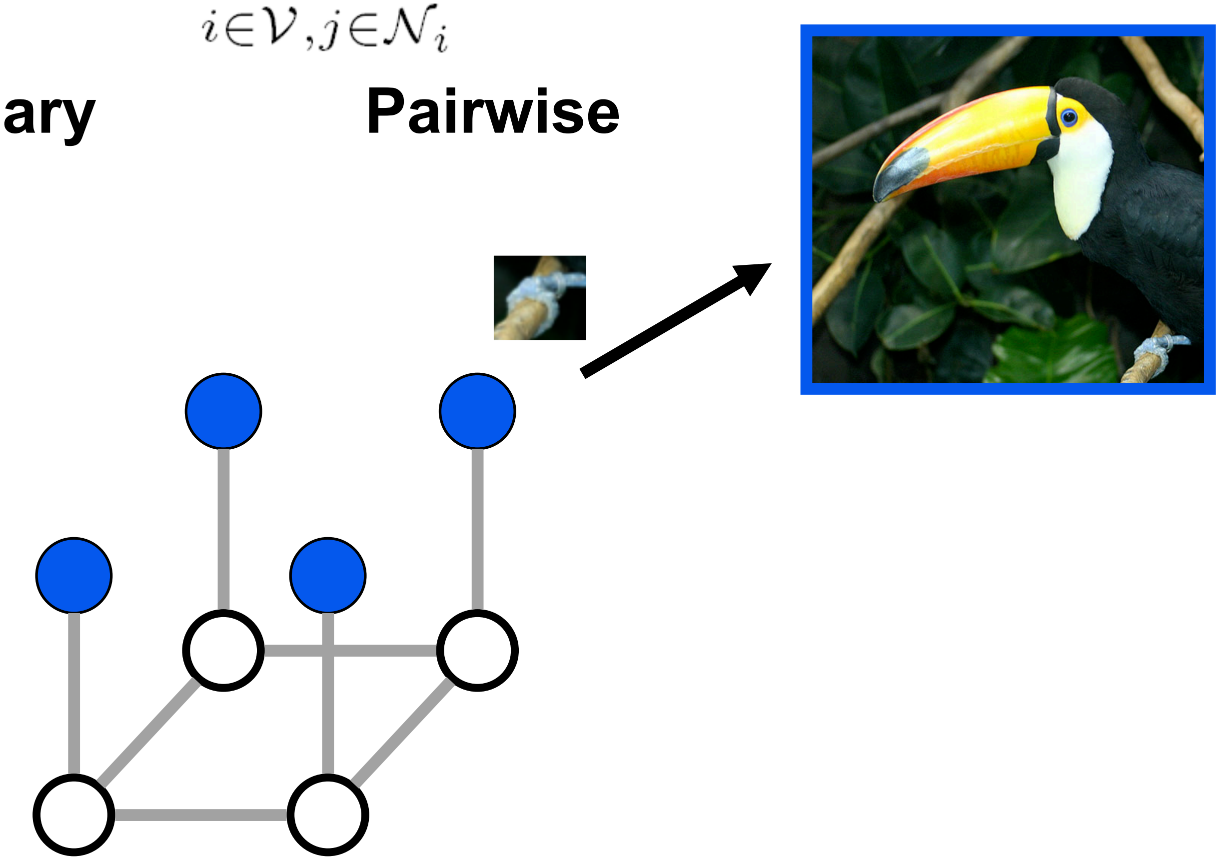


Pairwise Energy Functions

$$E(\mathbf{x}) = \sum_i \psi_i(x_i) + \sum_{i \in \mathcal{V}, j \in \mathcal{N}_i} \psi_{ij}(x_i, x_j)$$

Unary **Pairwise**

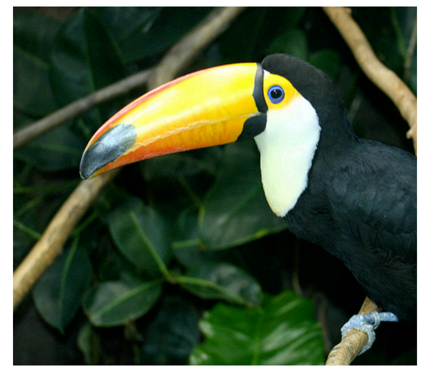
Markov Random Field



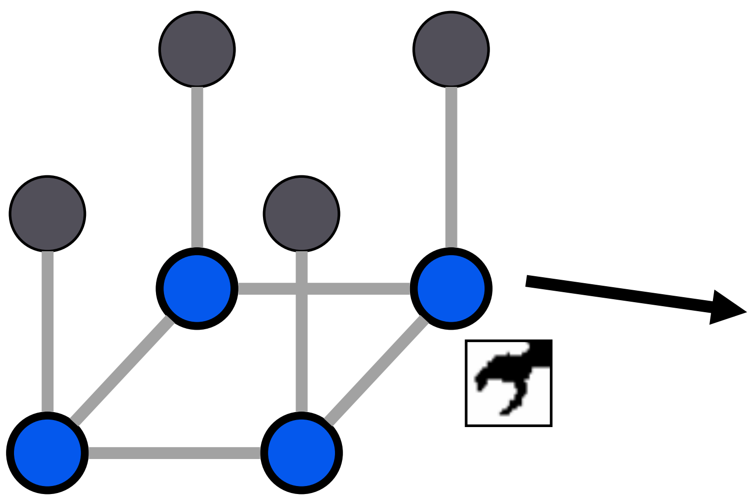
Energy Functions

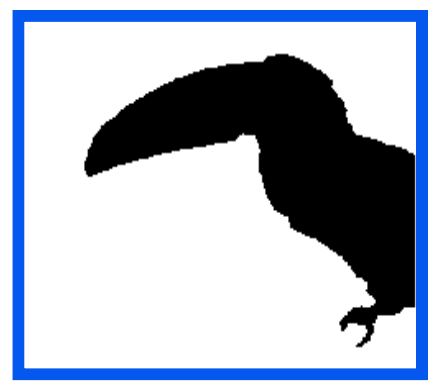
$$E(\mathbf{x}) = \sum_i \psi_i(x_i) + \sum_{i \in \mathcal{V}, j \in \mathcal{N}_i} \psi_{ij}(x_i, x_j)$$

Unary **Pairwise**



Markov Random Field

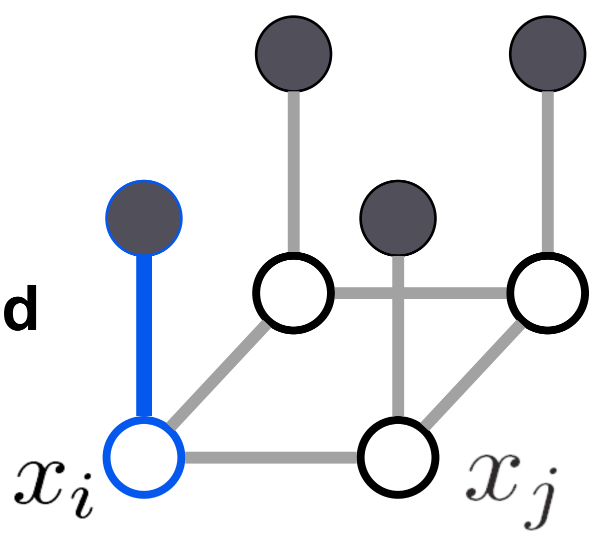




Pairwise Energy Functions

$$E(\mathbf{x}) = \boxed{\sum_i \psi_i(x_i)} + \sum_{i \in \mathcal{V}, j \in \mathcal{N}_i} \psi_{ij}(x_i, x_j)$$

Unary **Pairwise**

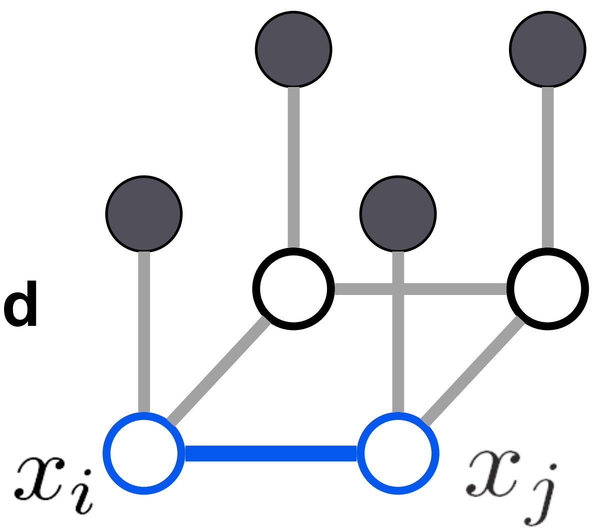


Pairwise Energy Functions

$$E(\mathbf{x}) = \sum_i \psi_i(x_i) + \boxed{\sum_{i \in \mathcal{V}, j \in \mathcal{N}_i} \psi_{ij}(x_i, x_j)}$$

Unary **Pairwise**

Markov Random Field



- **Efficient Algorithms for Minimization**
- **Restricted Expressive Power!**

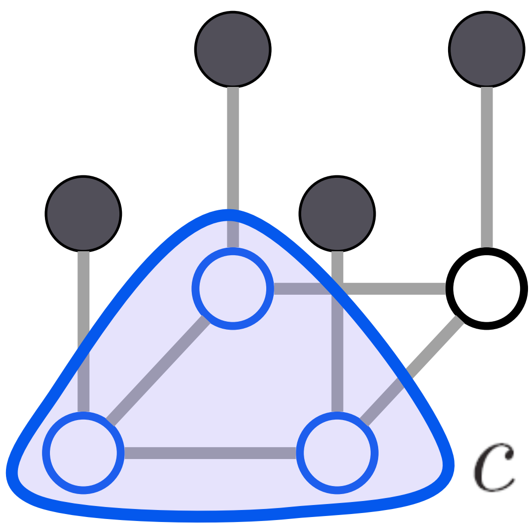
Higher Order Energy Functions

$$E(\mathbf{x}) = \sum_i \psi_i(x_i) + \sum_{i \in \mathcal{V}, j \in \mathcal{N}_i} \psi_{ij}(x_i, x_j) + \boxed{\sum_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c)}$$

Unary **Pairwise** **Higher order**

More expressive than pairwise

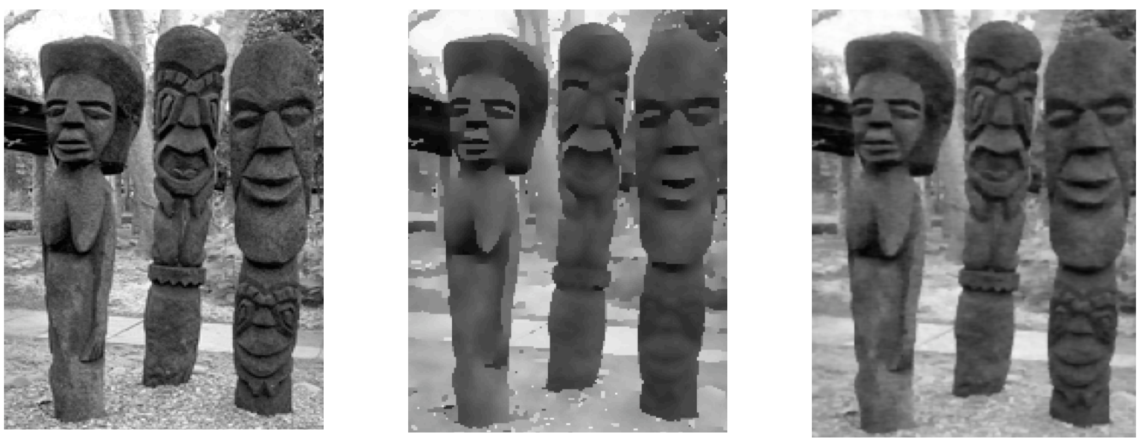
FOE: Field of Experts
(Roth & Black CVPR05)



Higher Order Energy Functions

$$E(\mathbf{x}) = \sum_i \psi_i(x_i) + \sum_{i \in \mathcal{V}, j \in \mathcal{N}_i} \psi_{ij}(x_i, x_j) + \boxed{\sum_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c)}$$

Unary **Pairwise** **Higher order**



Original **Pairwise MRF** **Higher order MRF**

MRF for Image Denoising

Images Courtesy: Lan *et al.* ECCV06

Higher Order Energy Functions

$$E(\mathbf{x}) = \sum_i \psi_i(x_i) + \sum_{i \in \mathcal{V}, j \in \mathcal{N}_i} \psi_{ij}(x_i, x_j) + \boxed{\sum_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c)}$$

Unary **Pairwise** **Higher order**

- **Computationally expensive to minimize!**
- **Exponential Complexity $O(L^N)$**
  - **L = Number of Labels**
  - **N = Size of Clique**

Higher Order Energy Functions

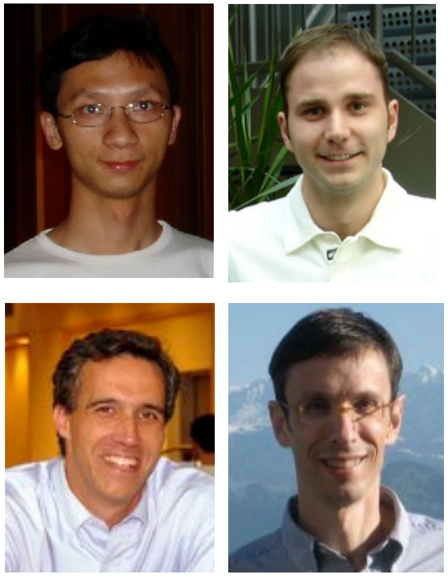
$$E(\mathbf{x}) = \sum_i \psi_i(x_i) + \sum_{i \in \mathcal{V}, j \in \mathcal{N}_i} \psi_{ij}(x_i, x_j) + \boxed{\sum_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c)}$$

Unary **Pairwise** **Higher order**

Efficient BP in Higher Order MRFs

ECCV06 (Lan, Roth, Huttenlocher, Black)

- **2x2 cliques learned using FOE model**
- **Approximation methods to make BP feasible**
- **Search a restricted state space**
- **16 minutes per iteration**



Higher Order Energy Functions

$$E(\mathbf{x}) = \sum_i \psi_i(x_i) + \sum_{i\in\mathcal{V}, j\in\mathcal{N}_i} \psi_{ij}(x_i, x_j) + \boxed{\sum_{c\in\mathcal{C}} \psi_c(\mathbf{x}_c)}$$

Unary **Pairwise** **Higher order**

- **Our Method**
  - **Can handle cliques of thousand of variables**
  - **Extremely Efficient (works in seconds)**

Talk Overview

- **Higher Order Energy Functions**
- **Optimization Algorithms**
- **Move making Algorithms**
- **Optimal moves for Higher Order Energies**
- **P^N Potts Model**
- **Experiments**

Optimization Algorithms

- **General Energy Functions**
  - **NP-hard to minimize**
  - **Algorithms for Approximate Minimization**

- **Easy energy functions**
  - **Global minima in polynomial time**
  - **Tree topology**
  - **Submodular functions**

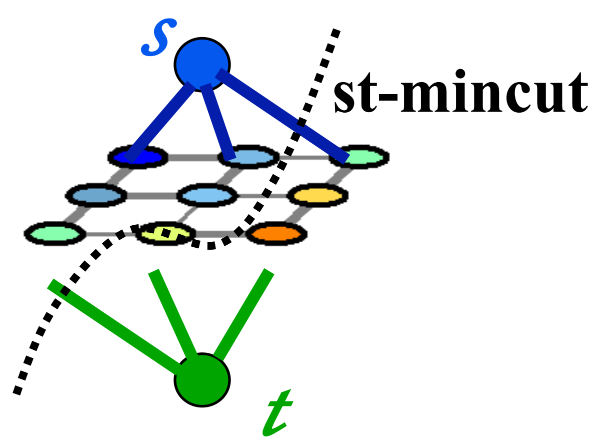
Submodular functions

- All projections on two variables are submodular.
- Any function f: $\{0,1\}^2 \rightarrow$ R is submodular if:

$$f(0,0) + f(1,1) \leq f(0,1) + f(1,0)$$

- In certain cases minimization equivalent to a st-mincut problem:

$$\arg\min_{\mathbf{x} \in \mathbf{L}} E(\mathbf{x}) \quad \Longleftrightarrow$$

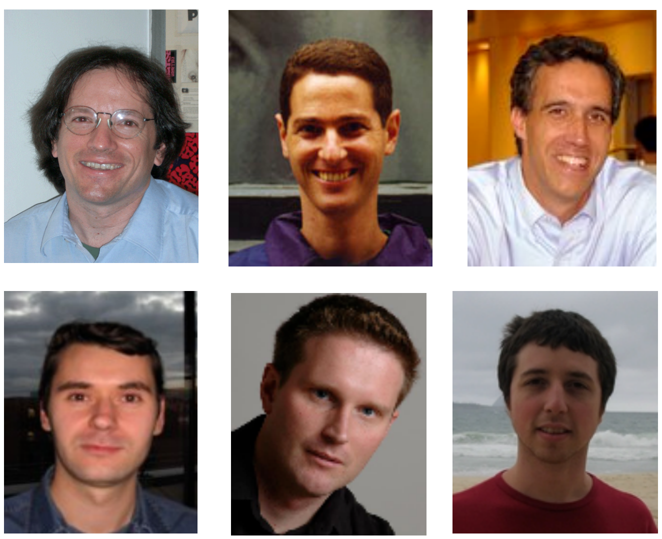


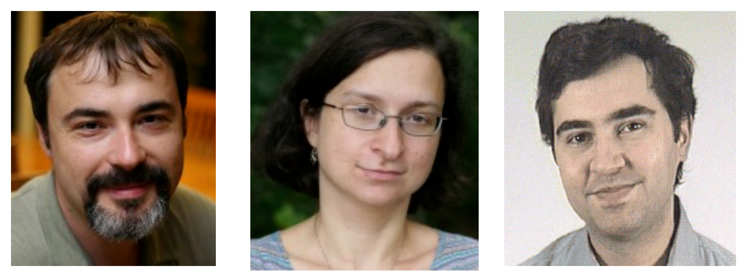
Minimization Problem

Graph Cut

Approximate Energy Functions

Message Passing Algorithms	**Move making Algorithms**
Belief Propagation (BP)	α−Expansion
Tree Reweighted (TRW)	$\alpha\beta$−Swap

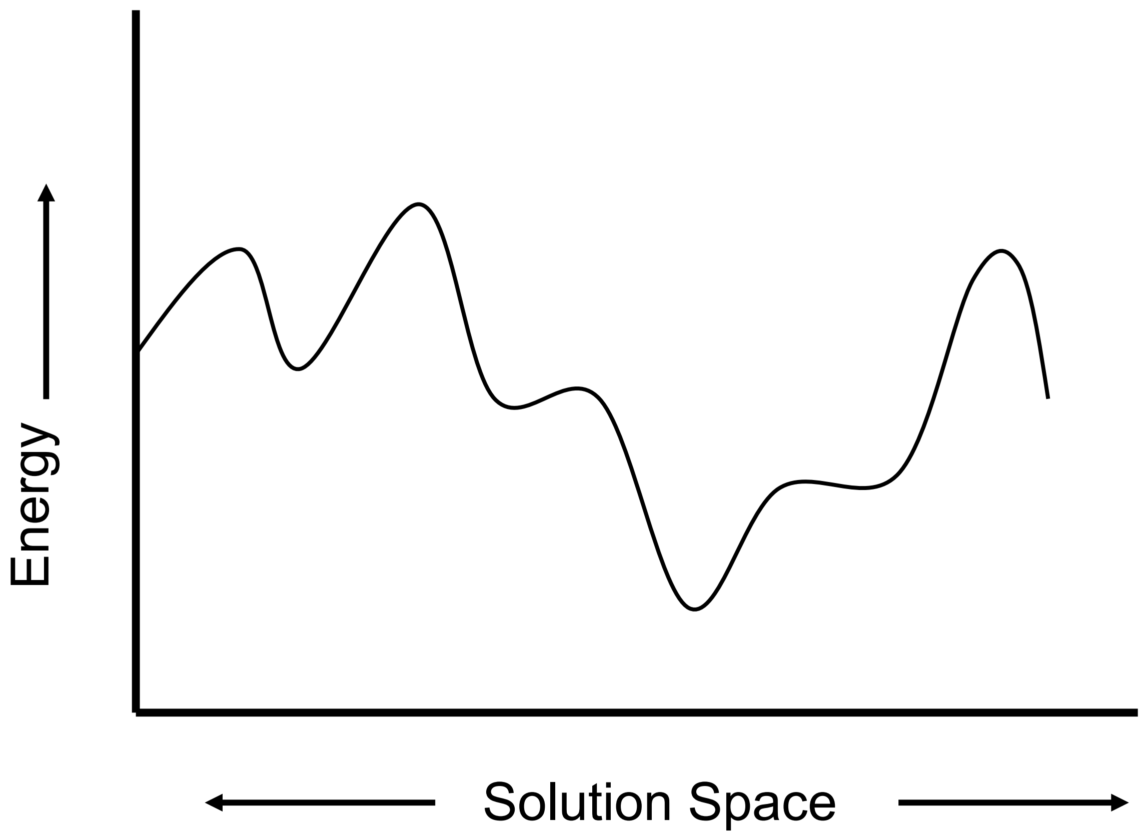




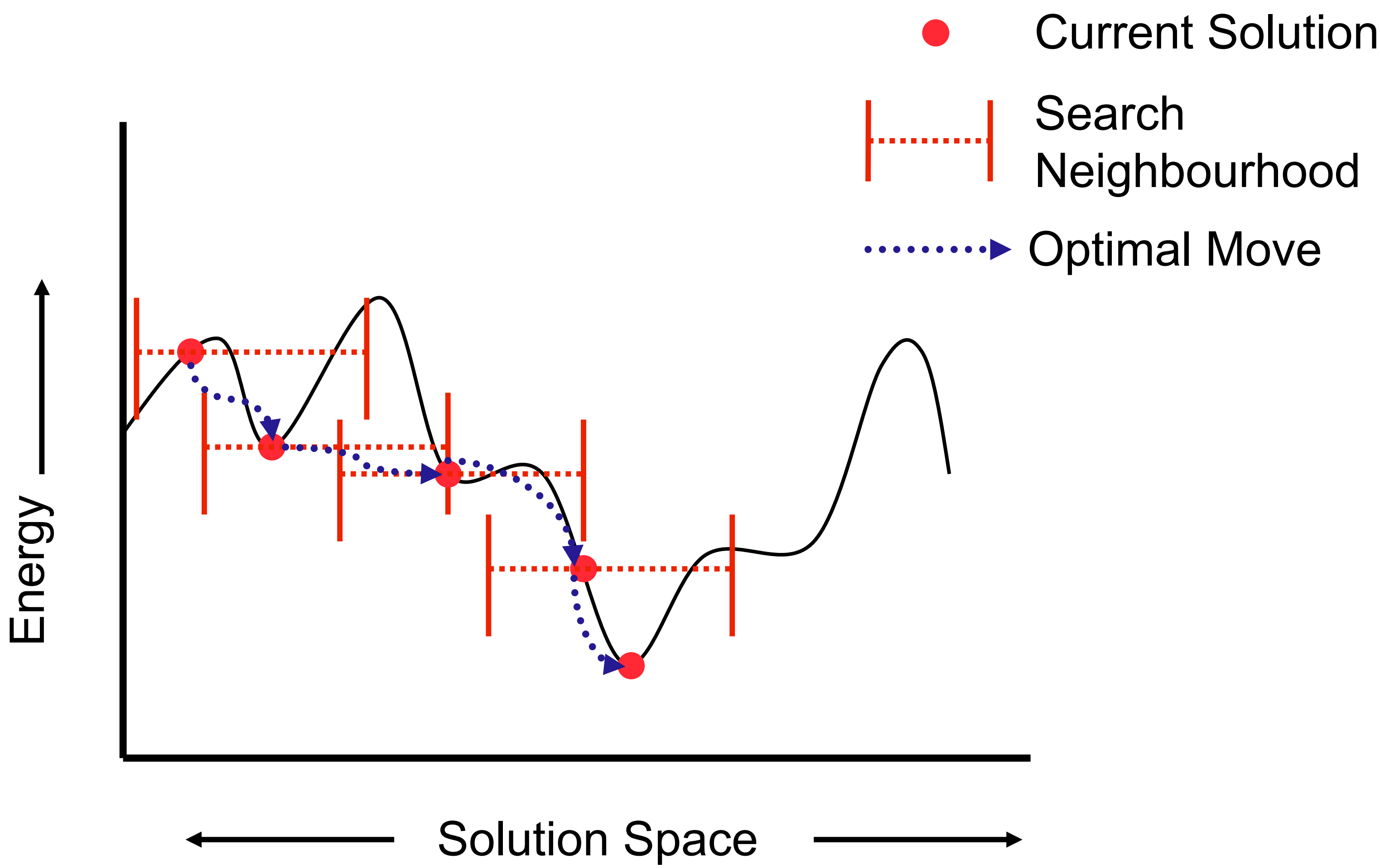
Talk Overview

- **Higher Order Energy Functions**
- **Optimization Algorithms**
- **Move making Algorithms**
- **Optimal moves for Higher Order Energies**
- **P^N Potts Model**
- **Experiments**

Move Making Algorithms



Move Making Algorithms



Move Making Algorithms

- **Moves using graph cuts**
- **[Boykov, Veksler, Zabih] PAMI 2001**

- **Algorithm**
  - **Encode move by vector t**
  - **Transformation function** $T(\mathbf{x},\mathbf{t}) \rightarrow \mathbf{x}`$
  - **Move Energy**

$$\boxed{E_m(\mathbf{t}) = E(T(\mathbf{x},\mathbf{t}))}$$

Submodular

  - **Optimal move t***

$$\mathbf{t}^* = \arg\min_{\mathbf{t}} E(T(\mathbf{x},\mathbf{t}))$$

Expansion Move

Characteristics

- **Move**
  - **Variables take label α or retain current label**

$$T_\alpha(x_i, t_i) = \begin{cases} x_i & \text{if} \quad t_i = 0 \\ \alpha & \text{if} \quad t_i = 1 \end{cases}$$

- **Algorithm**
  - **Make a move for all α in $\mathcal{L}$**

Expansion Move

Status:

Initialize with Tree
Expand Ground / House / Sky

- Tree
- Ground
- House
- Sky

Expansion Move

Characteristics

- **Neighbourhood Size**
  - **2^N where N is the number of variables**

- **Guarantee**
  - **Move energy is submodular for all metric energy functions.** **[Boykov, Veksler, Zabih] PAMI 2001**

$$\boxed{\begin{aligned} \psi_{ij}(a,b) &= 0 \iff a = b \\ \psi_{ij}(a,b) &= \psi_{ij}(b,a) \geq 0 \\ \psi_{ij}(a,d) &\leq \psi_{ij}(a,b) + \psi_{ij}(b,d) \end{aligned}}$$

Swap Move

Characteristics

- **Move**
  - **Variables labeled α, β can swap their labels**

$$T_{\alpha\beta}(x_i, t_i) = \begin{cases} \alpha & \text{if} \quad x_i = \alpha \text{ or } \beta \text{ and } t_i = 0 \\ \beta & \text{if} \quad x_i = \alpha \text{ or } \beta \text{ and } t_i = 1 \end{cases}$$

- **Algorithm**
  - **Make a move for all α, β in $\mathcal{L}$**

Swap Move

Tree
Ground
House
Sky

Swap Sky, House

Swap Move

Characteristics

- **Neighbourhood Size**
  - **2^N where N is the number of variables**
- **Guarantee**
  - **Move energy is submodular for all semi-metric energy functions.** **[Boykov, Veksler, Zabih] PAMI 2001**

$$\boxed{\begin{aligned} \psi_{ij}(a,b) &= 0 \iff a = b \\ \psi_{ij}(a,b) &= \psi_{ij}(b,a) \geq 0 \end{aligned}}$$

Talk Overview

- **Higher Order Energy Functions**
- **Optimization Algorithms**
- **Move making Algorithms**
- **Optimal moves for Higher Order Energies**
- **P^N Potts Model**
- **Experiments**

Moves for Higher Order Cliques

- **Form of the Higher Order Potentials**

$$\psi_c(\mathbf{x}_c) = f_c(\mathcal{Q}_c(\mathbf{x}_c))$$

- **Sum Form**

$$\mathcal{Q}_c(\mathbf{x}_c) = \sum_{i,j \in c} \phi_c(x_i, x_j)$$

- **Max Form**

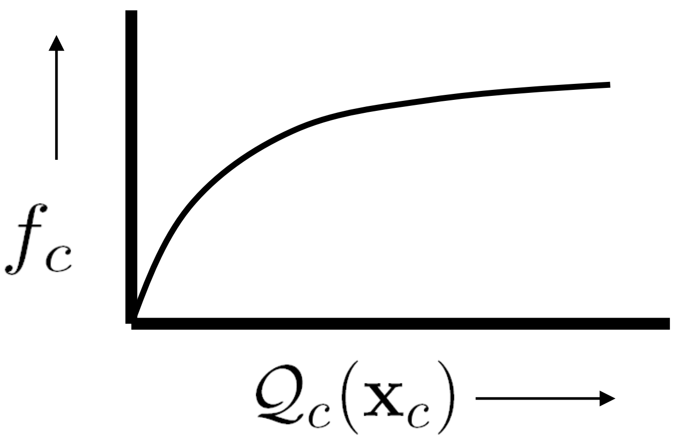
$$\mathcal{Q}_c(\mathbf{x}_c) = \max_{i,j \in c} \phi_c(x_i, x_j)$$

Theoretical Results: Swap

- **Move energy is always submodular if**

$$\boxed{\psi_c(\mathbf{x}_c) = f_c(\mathcal{Q}_c(\mathbf{x}_c))}$$

non-decreasing concave.



$$\boxed{\mathcal{Q}_c(\mathbf{x}_c) = \sum_{i,j \in c} \phi_c(x_i, x_j)}$$

$$\boxed{\begin{array}{ll} \phi_c(a,b) = \phi_c(b,a) & \forall a,b \in \mathcal{L} \\ \phi_c(a,b) \geq \phi_c(d,d) & \forall a,b,d \in \mathcal{L} \end{array}}$$

See paper for proofs

Theoretical Results: Expansion

- **Move energy is always submodular if**

$$\psi_c(\mathbf{x}_c) = f_c(\mathcal{Q}_c(\mathbf{x}_c))$$

increasing linear

$$\mathcal{Q}_c(\mathbf{x}_c) = \max_{i,j \in c} \phi_c(x_i, x_j)$$

$$\phi_c(x_i, x_j) = \begin{cases} \gamma_k & \text{if} \quad x_i = x_j = l_k \\ \gamma_{\max} & \text{otherwise.} \end{cases}$$

See paper for proofs

Talk Overview

- **Higher Order Energy Functions**
- **Optimization Algorithms**
- **Move making Algorithms**
- **Optimal moves for Higher Order Energies**
- **P^N Potts Model**
- **Experiments**

$\mathbf{P^N}$ Potts Model

$$\psi_c(\mathbf{x}_c) = \begin{cases} \gamma_k & \text{if} \quad x_i = l_k, \forall i \in c \\ \gamma_{\max} & \text{otherwise.} \end{cases}$$

$$\gamma_{\max} > \gamma_k$$

- **Generalization of the Potts Model:**

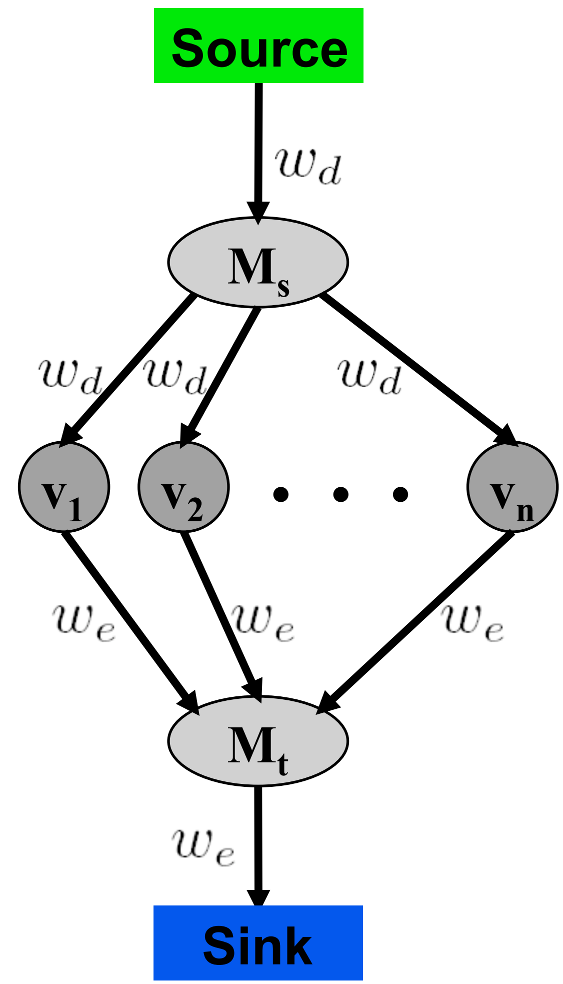
$$\phi_c(x_i, x_j) = \begin{cases} \gamma_k & \text{if} \quad x_i = x_j = l_k \\ \gamma_{\max} & \text{otherwise.} \end{cases}$$

$$\gamma_k = 0$$

Solving the P^N Potts Model

- Computing the optimal **swap** move

$$\psi_c(T_{\alpha\beta}(\mathbf{x}_c,\mathbf{t}_c)) = \begin{cases} \gamma_\alpha & \text{if } t_i = 0, \forall i \in c \\ \gamma_\beta & \text{if } t_i = 1, \forall i \in c \\ \gamma_{\max} & \text{otherwise.} \end{cases}$$



$$\kappa = \gamma_{\max} - \gamma_\alpha - \gamma_\beta$$
$$w_e = \gamma_\alpha + \kappa$$
$$w_d = \gamma_\beta + \kappa$$

$t_i = 0 \longleftrightarrow v_i \in$ **Source**

$t_j = 1 \longleftrightarrow v_j \in$ **Sink**

Solving the P^N Potts Model

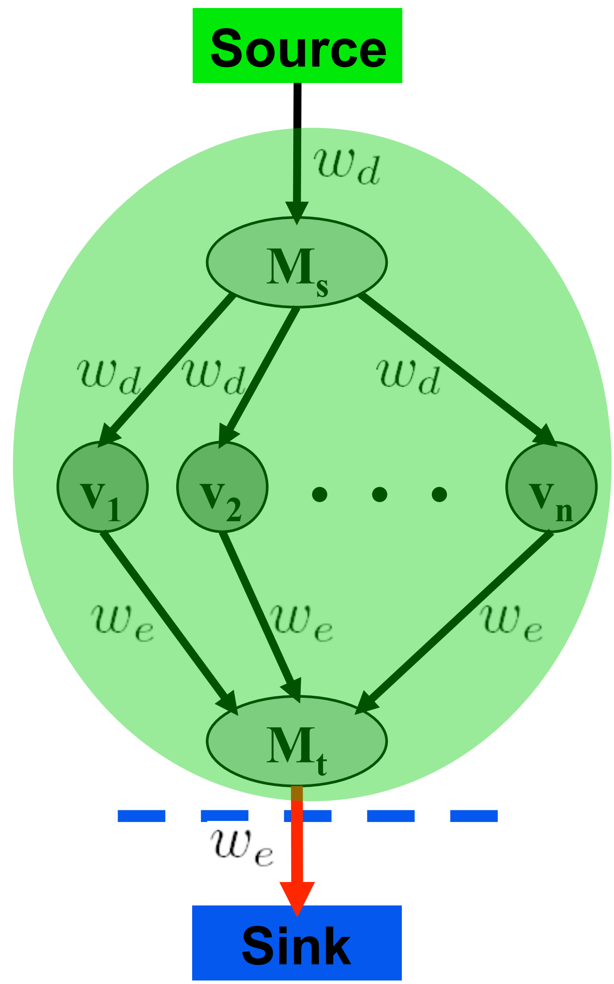
- Computing the optimal **swap** move

$$\psi_c(T_{\alpha\beta}(\mathbf{x}_c, \mathbf{t}_c)) = \begin{cases} \gamma_\alpha & \text{if } t_i = 0, \forall i \in c \\ \gamma_\beta & \text{if } t_i = 1, \forall i \in c \\ \gamma_{\max} & \text{otherwise.} \end{cases}$$

$$\kappa = \gamma_{\max} - \gamma_\alpha - \gamma_\beta$$
$$w_e = \gamma_\alpha + \kappa$$
$$w_d = \gamma_\beta + \kappa$$

Case 1: all $t_i = 0$ ($x_i = \alpha$)

Cost: $\gamma_\alpha + \kappa$



Solving the $\mathbf{P^N}$ Potts Model

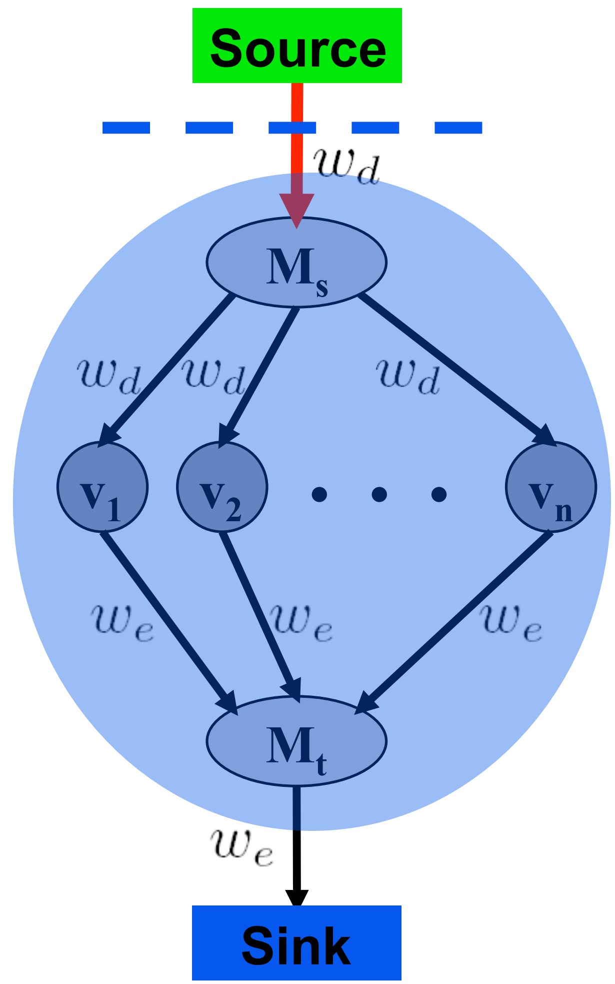
- Computing the optimal **swap** move

$$\psi_c(T_{\alpha\beta}(\mathbf{x}_c,\mathbf{t}_c)) = \begin{cases} \gamma_\alpha & \text{if } t_i = 0, \forall i \in c \\ \gamma_\beta & \text{if } t_i = 1, \forall i \in c \\ \gamma_{\max} & \text{otherwise.} \end{cases}$$

$$\kappa = \gamma_{\max} - \gamma_\alpha - \gamma_\beta$$
$$w_e = \gamma_\alpha + \kappa$$
$$w_d = \gamma_\beta + \kappa$$

Case 2: all $t_i = 1$ ($x_i = \beta$)

Cost: $\gamma_\beta + \kappa$



Solving the P^N Potts Model

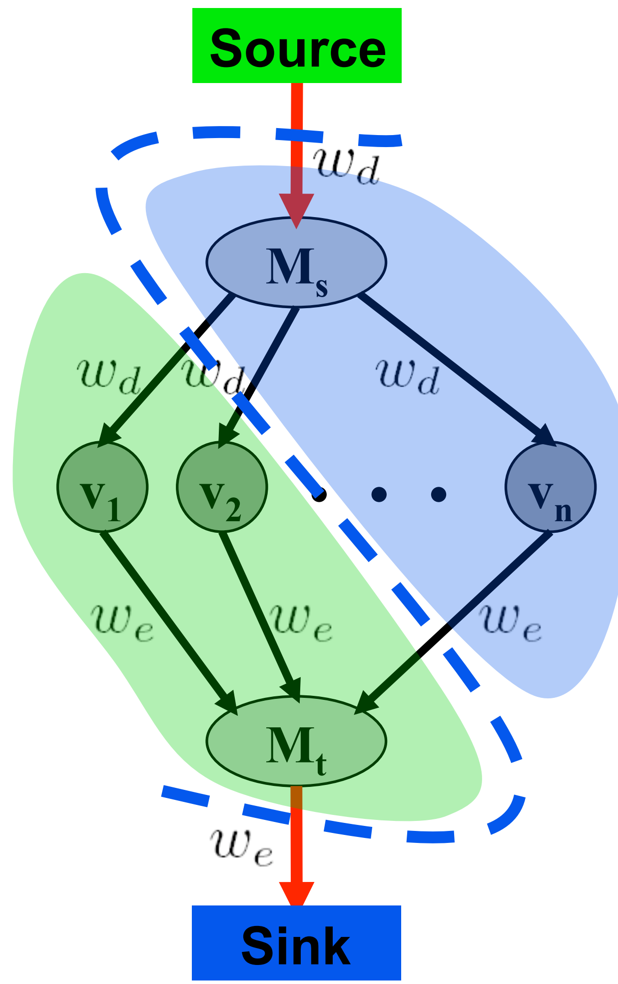
- Computing the optimal **swap** move

$$\psi_c(T_{\alpha\beta}(\mathbf{x}_c, \mathbf{t}_c)) = \begin{cases} \gamma_\alpha & \text{if } t_i = 0, \forall i \in c \\ \gamma_\beta & \text{if } t_i = 1, \forall i \in c \\ \gamma_{\max} & \text{otherwise.} \end{cases}$$

$$\kappa = \gamma_{\max} - \gamma_\alpha - \gamma_\beta$$
$$w_e = \gamma_\alpha + \kappa$$
$$w_d = \gamma_\beta + \kappa$$

Case 3: t_i = 0,1 ($x_i = \alpha, \beta$)

Cost: $\gamma_{\max} + \kappa$



Solving the $\mathbf{P}^N$ Potts Model

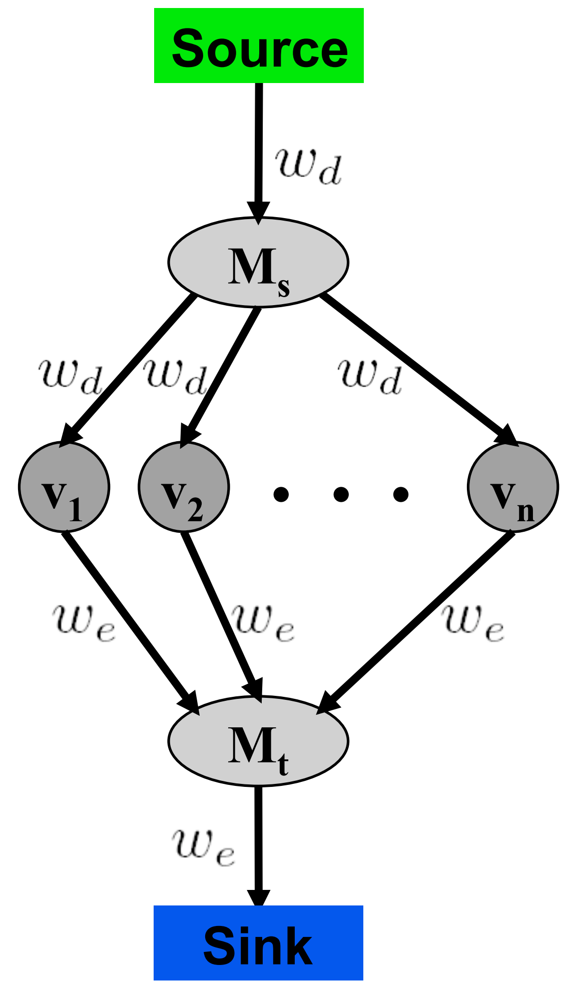
- Computing the optimal **expansion** move

$$\psi_c(T_\alpha(\mathbf{x}_c, \mathbf{t}_c)) = \begin{cases} \gamma & \text{if } t_i = 0, \forall i \in c \\ \gamma_\alpha & \text{if } t_i = 1, \forall i \in c \\ \gamma_{\max} & \text{otherwise,} \end{cases}$$

$$\kappa = \gamma_{\max} - \gamma_\alpha - \gamma$$

$$w_d = \gamma_\alpha + \kappa$$

$$w_e = \gamma + \kappa$$



Solving the $\mathbf{P}^{\mathbf{N}}$ Potts Model

- Computing the optimal **expansion** move

$$\psi_c(T_\alpha(\mathbf{x}_c, \mathbf{t}_c)) = \begin{cases} \gamma & \text{if } t_i = 0, \forall i \in c \\ \gamma_\alpha & \text{if } t_i = 1, \forall i \in c \\ \gamma_{\max} & \text{otherwise,} \end{cases}$$

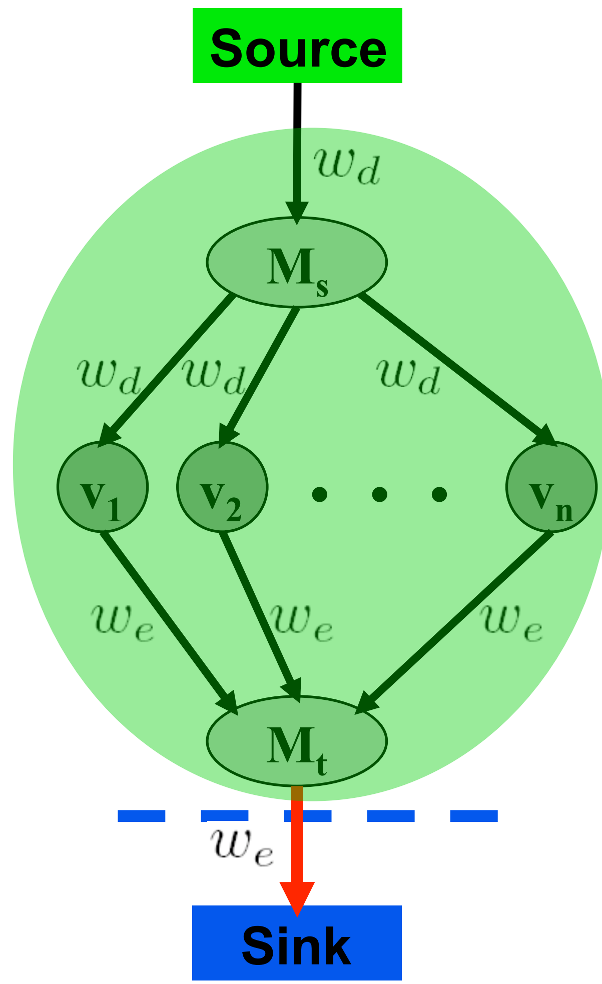
$$\kappa = \gamma_{\max} - \gamma_\alpha - \gamma$$

$$w_d = \gamma_\alpha + \kappa$$

$$w_e = \gamma + \kappa$$

Case 1: all $t_i = 0$ ($x_i = x_i$)

Cost: $\gamma + \kappa$



Solving the $\mathbf{P}^N$ Potts Model

- Computing the optimal **expansion** move

$$\psi_c(T_\alpha(\mathbf{x}_c, \mathbf{t}_c)) = \begin{cases} \gamma & \text{if } t_i = 0, \forall i \in c \\ \gamma_\alpha & \text{if } t_i = 1, \forall i \in c \\ \gamma_{\max} & \text{otherwise,} \end{cases}$$

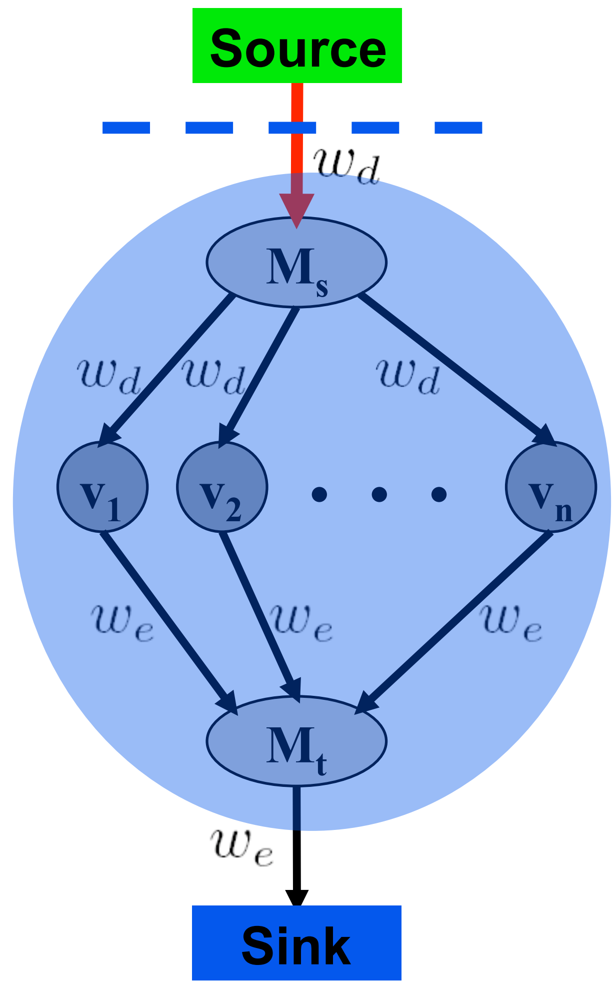
$$\kappa = \gamma_{\max} - \gamma_\alpha - \gamma$$

$$w_d = \gamma_\alpha + \kappa$$

$$w_e = \gamma + \kappa$$

Case 2: all $t_i = 1$ ($x_i = \alpha$)

Cost: $\gamma_\alpha + \kappa$



Solving the $\mathbf{P}^{\mathbf{N}}$ Potts Model

- Computing the optimal **expansion** move

$$\psi_c(T_\alpha(\mathbf{x}_c, \mathbf{t}_c)) = \begin{cases} \gamma & \text{if } t_i = 0, \forall i \in c \\ \gamma_\alpha & \text{if } t_i = 1, \forall i \in c \\ \gamma_{\max} & \text{otherwise,} \end{cases}$$

$$\kappa = \gamma_{\max} - \gamma_\alpha - \gamma$$

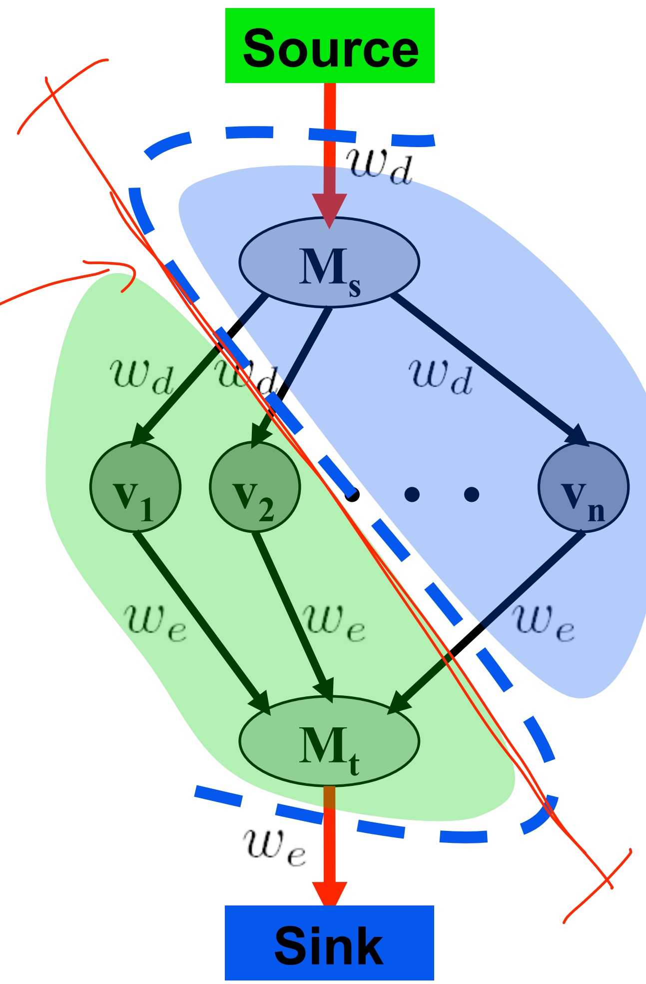
$$w_d = \gamma_\alpha + \kappa$$

$$w_e = \gamma + \kappa$$

Case 3: t_i = 0,1 ($x_i = x_i$, α)

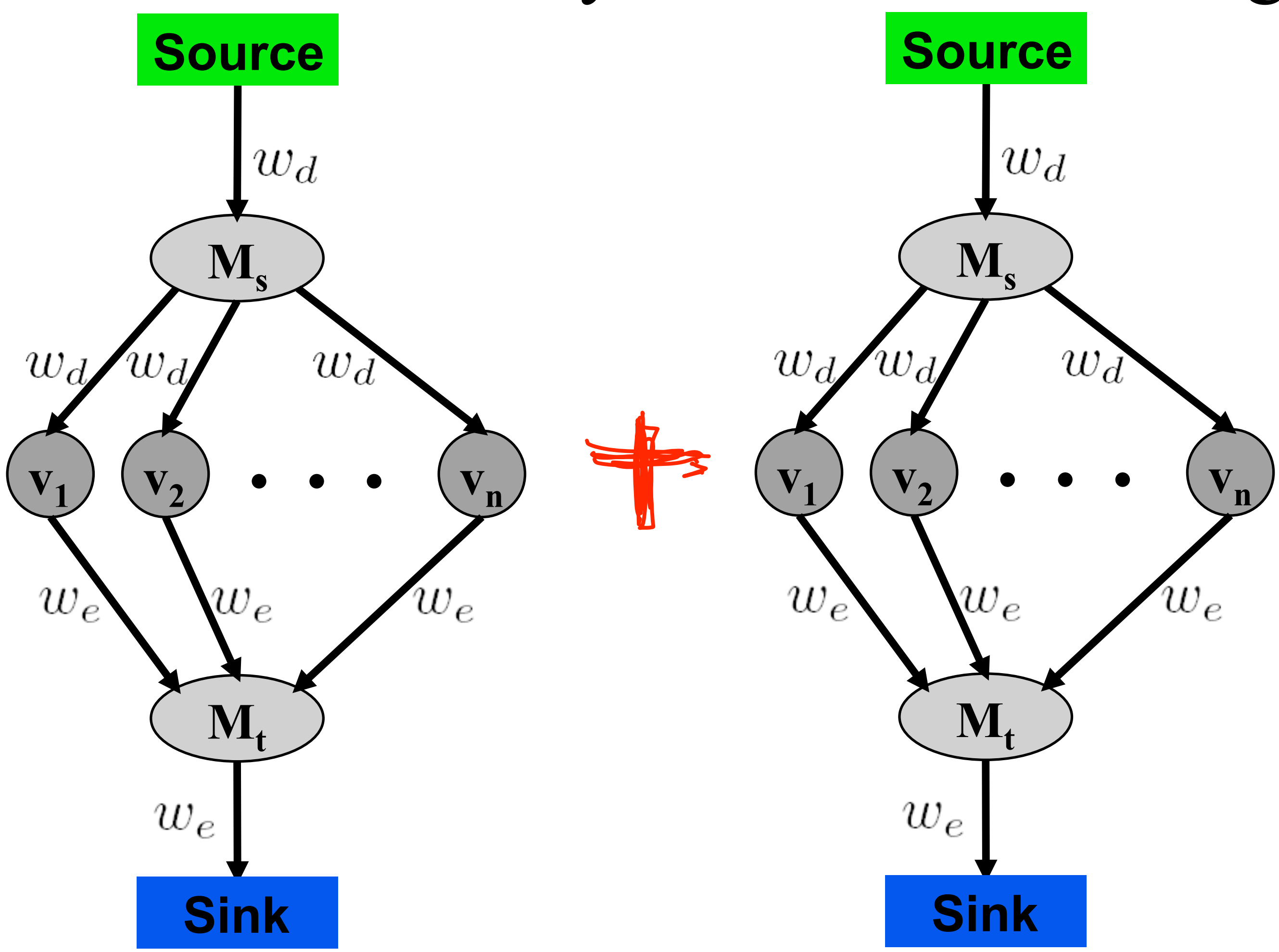
Cost: $\gamma_{\max} + \kappa$

cut



Additivity:

- We can add any number of these together



+ UNARIES

& c.

Talk Overview

- Higher Order Energy Functions
- Optimization Algorithms
- Move making Algorithms
- Optimal moves for Higher Order Energies
- P^N Potts Model
- Experiments

Exemplar based Texture Segmentation

$$E(\mathbf{x}) = \sum_i \psi_i(x_i) + \sum_{i \in \mathcal{V}, j \in \mathcal{N}_i} \psi_{ij}(x_i, x_j) + \sum_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c)$$

Unary (Colour)

Pairwise (Smoothness)

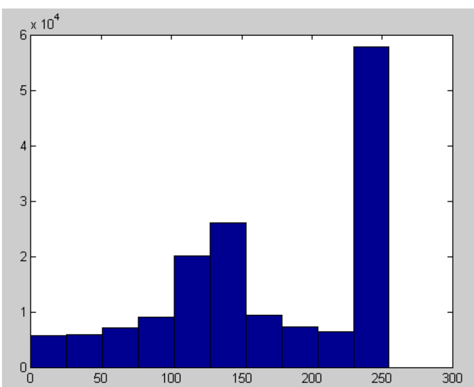
Higher Order (Texture)

Original Image

Exemplar based Texture Segmentation

$$E(\mathbf{x}) = \sum_i \psi_i(x_i) + \sum_{i \in \mathcal{V}, j \in \mathcal{N}_i} \psi_{ij}(x_i, x_j) + \sum_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c)$$

Unary (Colour) | **Pairwise (Smoothness)** | **Higher Order (Texture)**



Colour Histograms

Unary Cost: Tree

Exemplar based Texture Segmentation

$$E(\mathbf{x}) = \sum_i \psi_i(x_i) + \sum_{i \in \mathcal{V}, j \in \mathcal{N}_i} \psi_{ij}(x_i, x_j) + \sum_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c)$$

Unary (Colour)

Pairwise (Smoothness)

Higher Order (Texture)

Edge Sensitive Smoothness Cost

Exemplar based Texture Segmentation

$$E(\mathbf{x}) = \sum_i \psi_i(x_i) + \sum_{i \in \mathcal{V}, j \in \mathcal{N}_i} \psi_{ij}(x_i, x_j) + \sum_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c)$$

Unary (Colour)

Pairwise (Smoothness)

Higher Order (Texture)

MAP Solution

Exemplar based Texture Segmentation

$$E(\mathbf{x}) = \sum_i \psi_i(x_i) + \sum_{i\in\mathcal{V}, j\in\mathcal{N}_i} \psi_{ij}(x_i, x_j) + \sum_{c\in\mathcal{C}} \psi_c(\mathbf{x}_c)$$

Unary (Colour)

Pairwise (Smoothness)

Higher Order (Texture)

Patch Dictionary

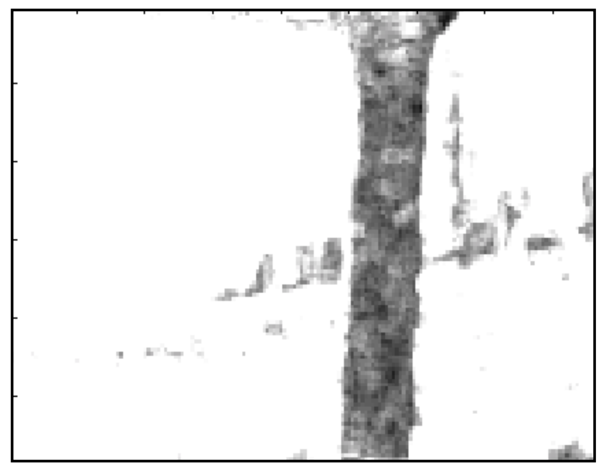
Higher Order Cost: Tree

Exemplar based Texture Segmentation

$$E(\mathbf{x}) = \sum_{i} \psi_i(x_i) + \sum_{i \in \mathcal{V}, j \in \mathcal{N}_i} \psi_{ij}(x_i, x_j) + \sum_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c)$$

Unary (Colour) | **Pairwise (Smoothness)** | **Higher Order (Texture)**

Unary Cost: Tree



Higher Order Cost: Tree

Exemplar based Texture Segmentation

$$E(\mathbf{x}) = \sum_i \psi_i(x_i) + \sum_{i \in \mathcal{V}, j \in \mathcal{N}_i} \psi_{ij}(x_i, x_j) + \sum_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c)$$

Unary (Colour) | **Pairwise (Smoothness)** | **Higher Order (Texture)**

Original

Pairwise

Higher order

Experimental Results

Pairwise | **Higher Order**

Original

Swap (3.2 sec)

Swap (4.2 sec)

Ground Truth

Expansion (2.5 sec)

Expansion (3.0 sec)

Experimental Results

Pairwise | **Higher Order**

Original

Swap (4.7 sec)

Swap (5.0 sec)

Ground Truth

Expansion (3.7sec)

Expansion (4.4 sec)

Conclusions & Future Work

- **Efficient minimization of certain higher order energies**
- **Can handle very large cliques**
- **Useful for many Computer Vision problems**
- **Explore other interesting family of potential functions**

UNIVERSITY OF CAMBRIDGE

Microsoft® Research Cambridge

TextonBoost :

Joint Appearance, Shape and Context Modeling for Multi-Class Object Recognition and Segmentation

J. Shotton*, **J. Winn**†, **C. Rother**†, and **A. Criminisi**†

* University of Cambridge

† Microsoft Research Ltd, Cambridge, UK

Introduction

- Simultaneous recognition and segmentation
  - Explain every pixel (dense features)
  - Appearance + shape + context
  - Class generalities + image specifics

Example Results

Structure of Presentation

- The MSRC 21-Class Object Recognition Database
- New ‘Shape Filter’ Features
- Conditional Random Field (CRF) Model
- Comparative Results

Image Databases

-]
  - 591 hand-labelled images (45% train, 10% validation, 45% test)
- **Corel** (7-class) and **Sowerby** (7-class) [He *et al.* CVPR 04]

Conditional Random Field Model

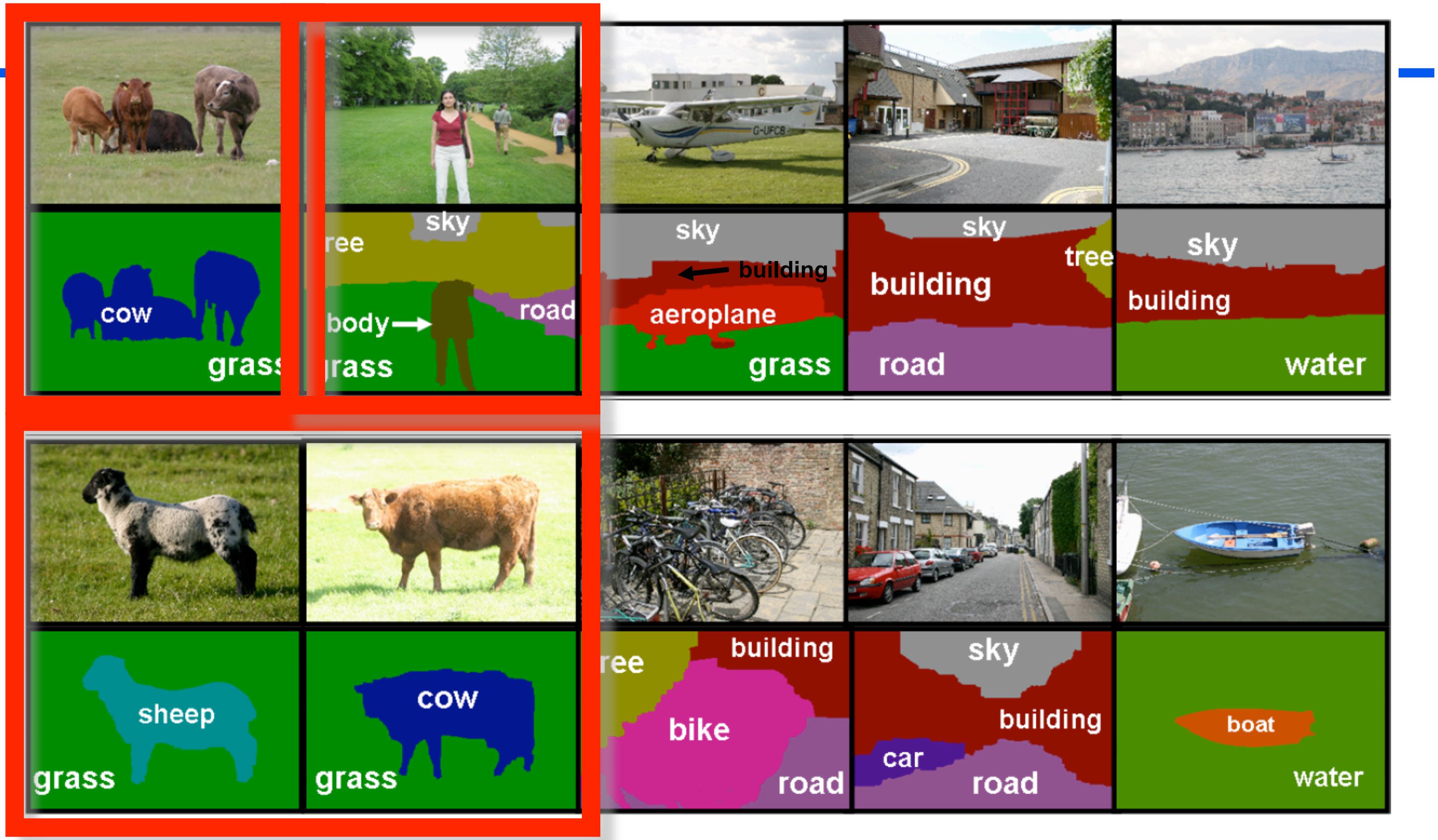
$$\log P(\mathbf{c}|\mathbf{x},\theta) = \sum_i \psi_i(c_i,\mathbf{x};\theta_\psi) + \pi(c_i,\mathbf{x}_i;\theta_\pi) + \lambda(c_i,i;\theta_\lambda) + \underbrace{\sum_{(i,j)\in\mathcal{E}}}_{\text{sum over neighbouring pixels}} \underbrace{\phi(c_i,c_j,\mathbf{g}_{ij}(\mathbf{x});\theta_\phi)}_{\text{edge potentials}}$$

[redacted]

- Potts model
  - encourages neighbouring pixels to have same label
- Contrast sensitivity
  - encourages segmentation to follow image edges

image edge map

Results on 21-Class Database



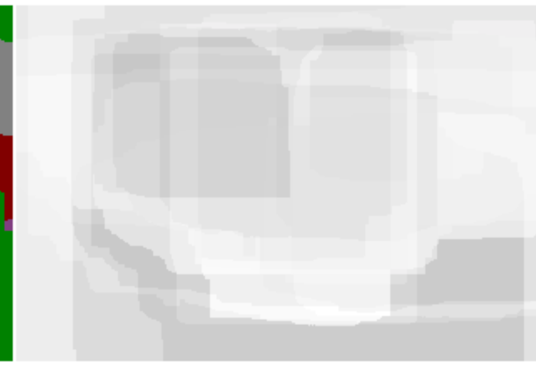
Object classes	Building	Grass	Tree	Cow	Sheep	Sky	Aeroplane	Water	Face	Car
Bike	Flower	Sign	Bird	Book	Chair	Road	Cat	Dog	Body	Boat

Joint Boosting for Feature Selection

- Boosted classifier provides *bulk* segmentation/recognition only
- Edge accurate segmentation will be provided by CRF model

test image

inferred segmentation
colour = most likely label



confidence
white = low confidence
black = high confidence

Object classes	Building	Grass	Tree	Cow	Sheep	Sky	Aeroplane	Water	Face	Car
Bike	Flower	Sign	Bird	Book	Chair	Road	Cat	Dog	Body	Boat

Using Joint Boost: [Torralba *et al.* CVPR 2004]

Our Work, CVPR 2008

- Idea to use super pixels to help the segmentation process…

Minimizing Higher Order Energy Functions using Graph Cuts

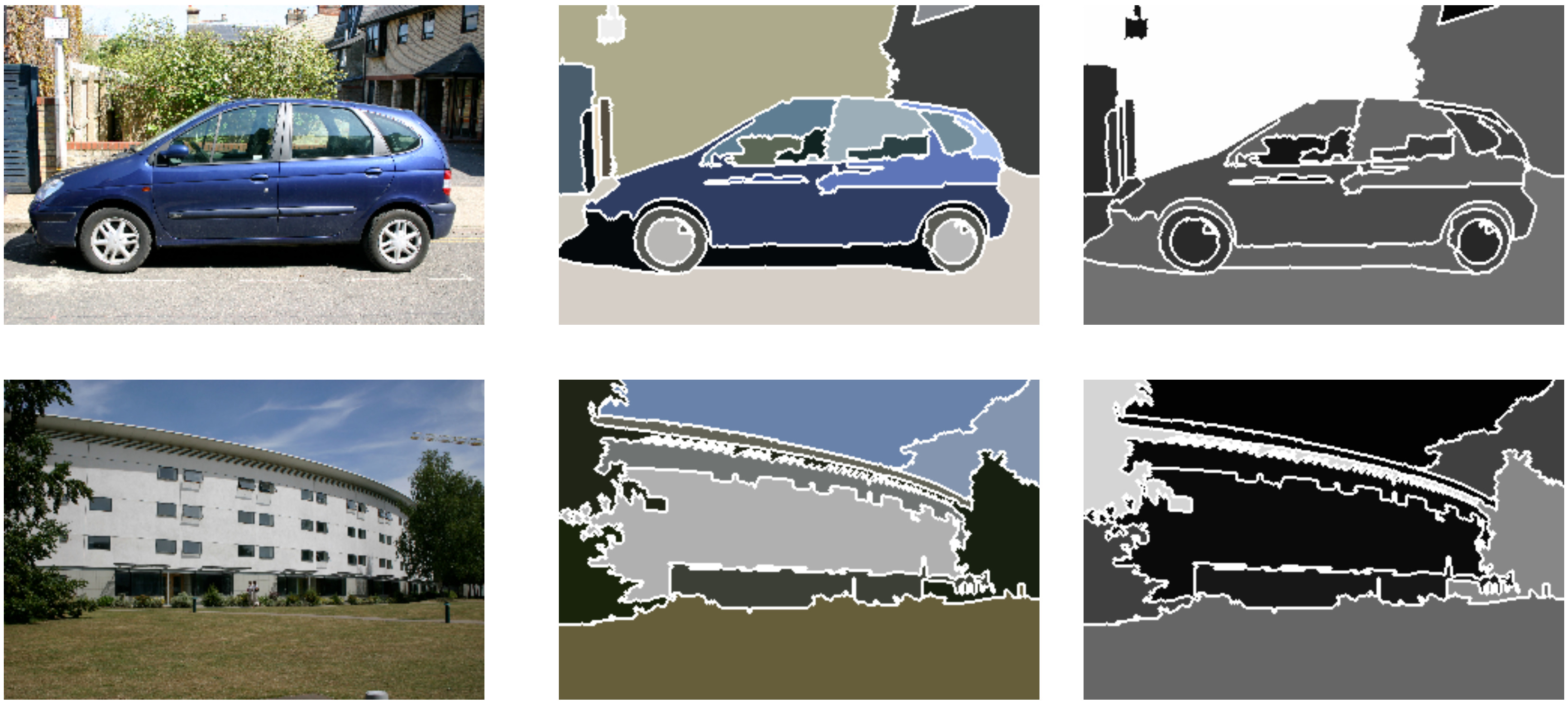
Pushmeet Kohli

Joint work with:

Lubor Ladicky   Pawan Kumar Philip Torr

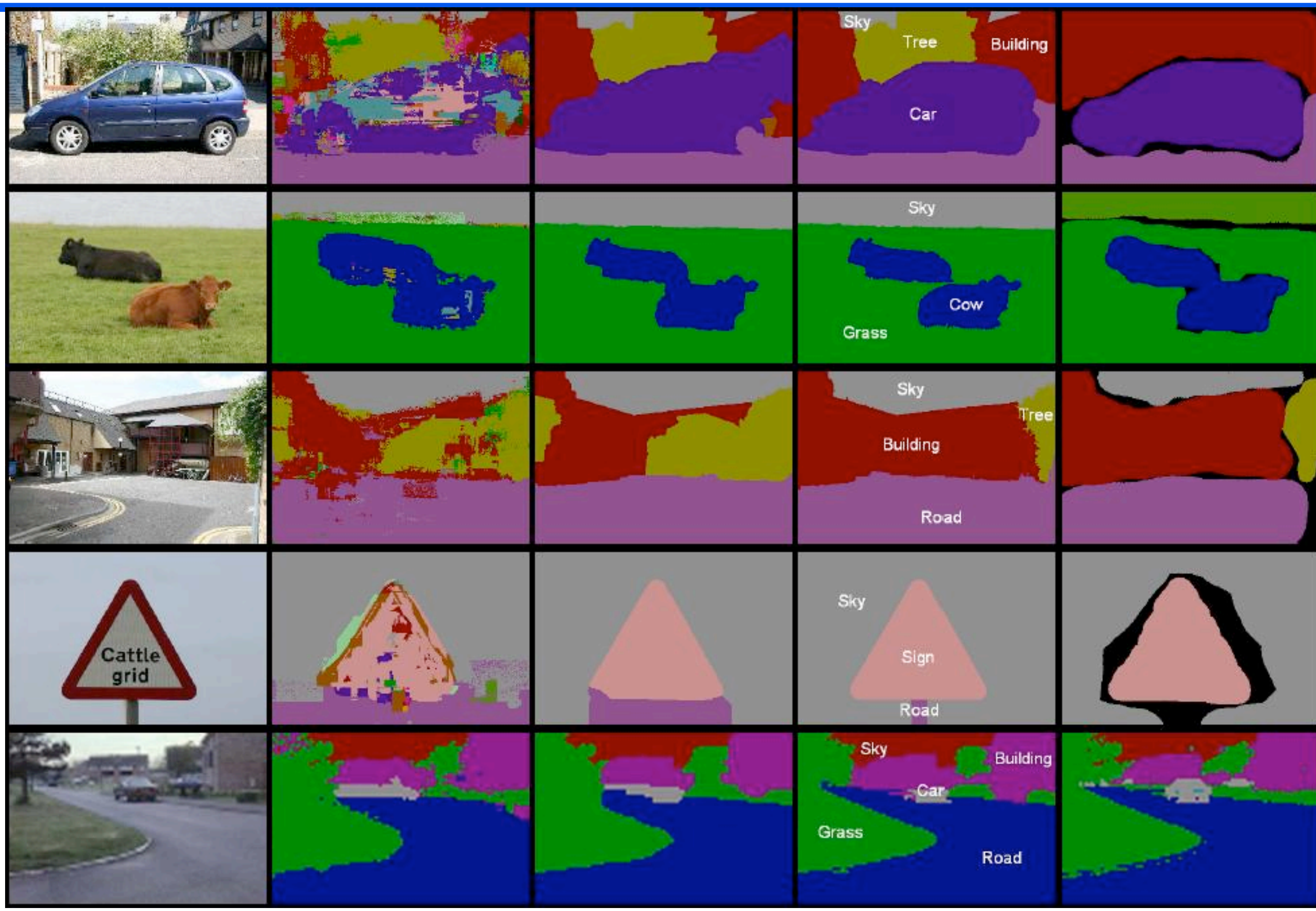
IPAM, February 25th, 2008

Variance-sensitive consistency potential



MSRC image　　Meanshift segmentation　　Variance

Labels Correspond to different objects



Original Image Unary Potential Pairwise CRF Higher Order CRF Ground Truth

Thanks

- Questions?

Conclusion

- Combinatorial Optimization is an exciting field with many applications.
- New Developments all the time (e.g. Tree Re-weighted message passing)
- Deterministic methods taking the lead over stochastic for discrete labels.
- Challenge to extend success to continuous cases.