UCU Summer School

Alexander Shekhovtsov
Kostiantyn Antonuk

Outline

- D1 Starting toolbox for statistical recognition

- D2 Structured prediction
  - Hidden Markov model, Markov random field (MRF)
  - Inference problems, EM with MAR
  - Support Vector Machine

- D3 Learning for structured prediction
  - Structured output SVM, advanced examples
  - Cutting Plane methods

Introduction

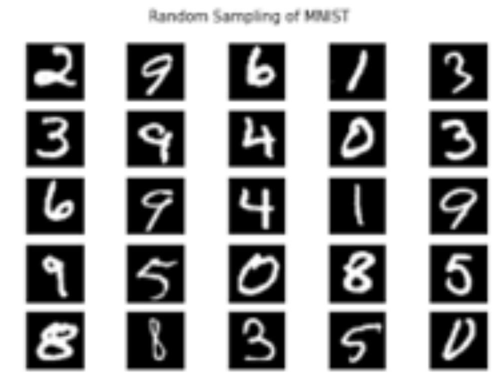
Classification

- SVM era

$\{0, 1\}$

⋮

$\{1,..K\}$



- Deep NN era

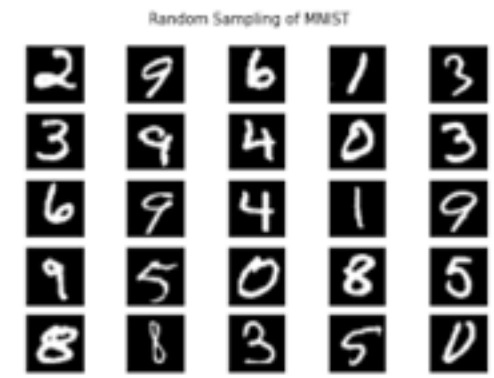
Classification

- SVM era

$\{0, 1\}$

$\vdots$

$\{1,..K\}$



  - Design measurements, represent them as a feature vector

- Deep NN era

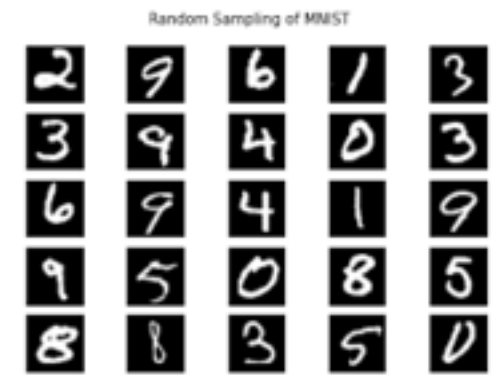
Classification

- SVM era

$\{0, 1\}$

⋮

$\{1,..K\}$



  - Design measurements, represent them as a feature vector
  - Learn the best discriminant function

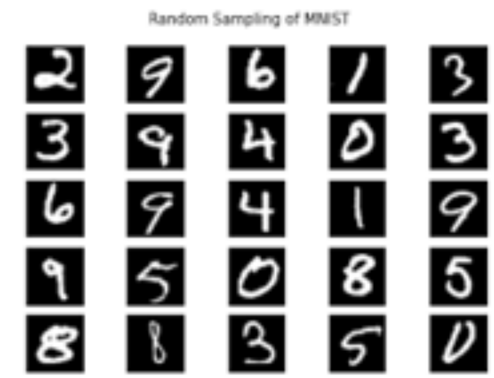
- Deep NN era

Classification

- SVM era

$\{0, 1\}$

$\{1,..K\}$



  - Design measurements, represent them as a feature vector
  - Learn the best discriminant function

- Deep NN era

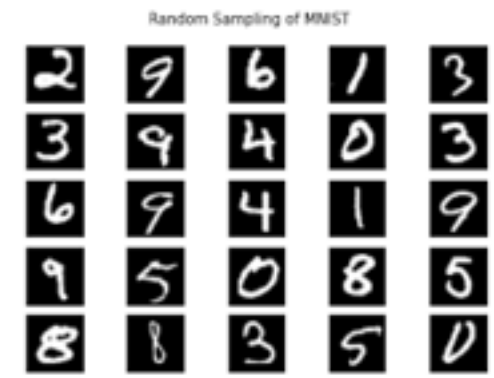
  - Learn some feature vector

Classification

- SVM era

$\{0, 1\}$

$\{1,..K\}$

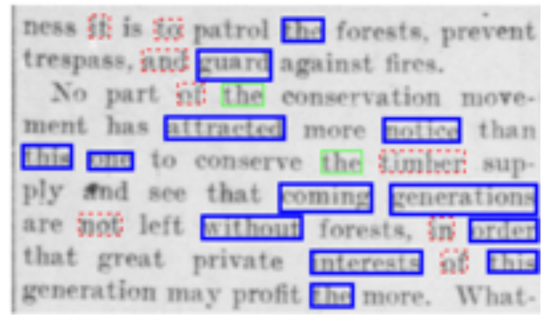


  - Design measurements, represent them as a feature vector
  - Learn the best discriminant function

- Deep NN era
  - Learn some feature vector
  - Apply SVM

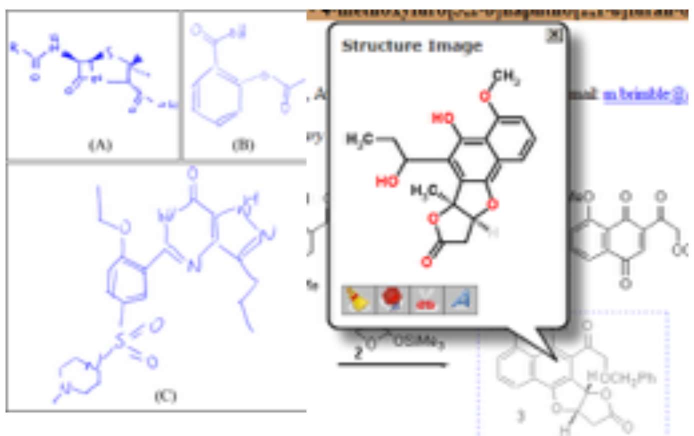
Structured Prediction

- Text Recognition


ness it is to patrol the forests, prevent
trespass, and guard against fires.
No part of the conservation move-
ment has attracted more notice than
this one to conserve the timber sup-
ply and see that coming generations
are not left without forests, in order
that great private interests of this
generation may profit the more. What-

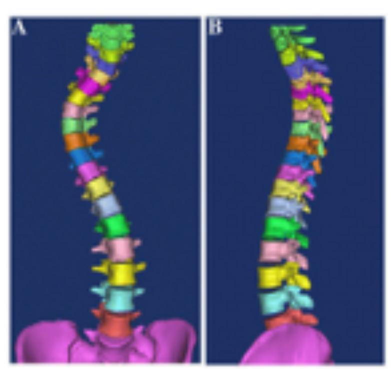
{space of text sentences}

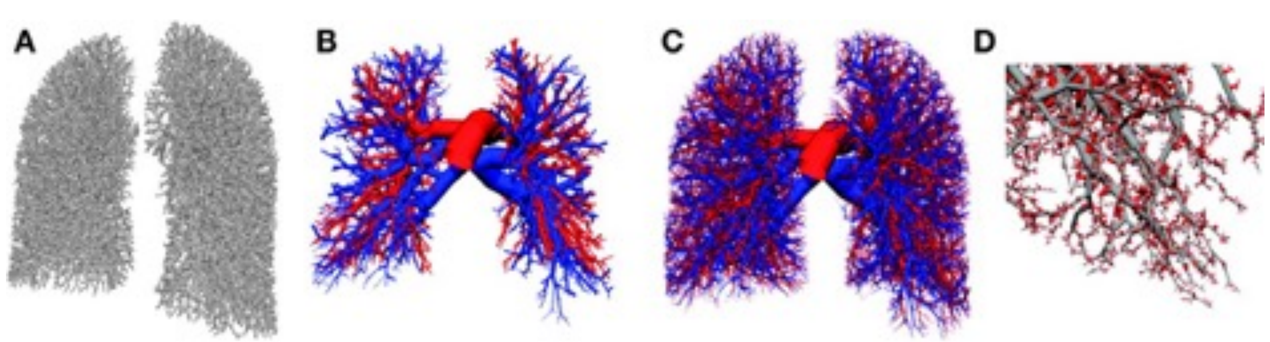
- Optical Structure Recognition



- Image Segmentation

- Body Parts Segmentation





- Facial Landmarks Detection

Markov Models

Outline

- Statistical models over large (structured) state spaces
  - Conditional independences, p.d.f. factorization
- Hidden Markov model (chain)
  - Connection: recurrent NNs
- Markov random field, conditional random field
  - Connection: CNN, deep Bolzman machine

The Modeling Problem

- Consider a collection of random variables describing hidden state

$$x_1, x_2, \dots x_n, \qquad x_i \in D$$

- Wish to have a statistical model:

$$p(x) = p(x_1, x_2, \dots x_n)$$

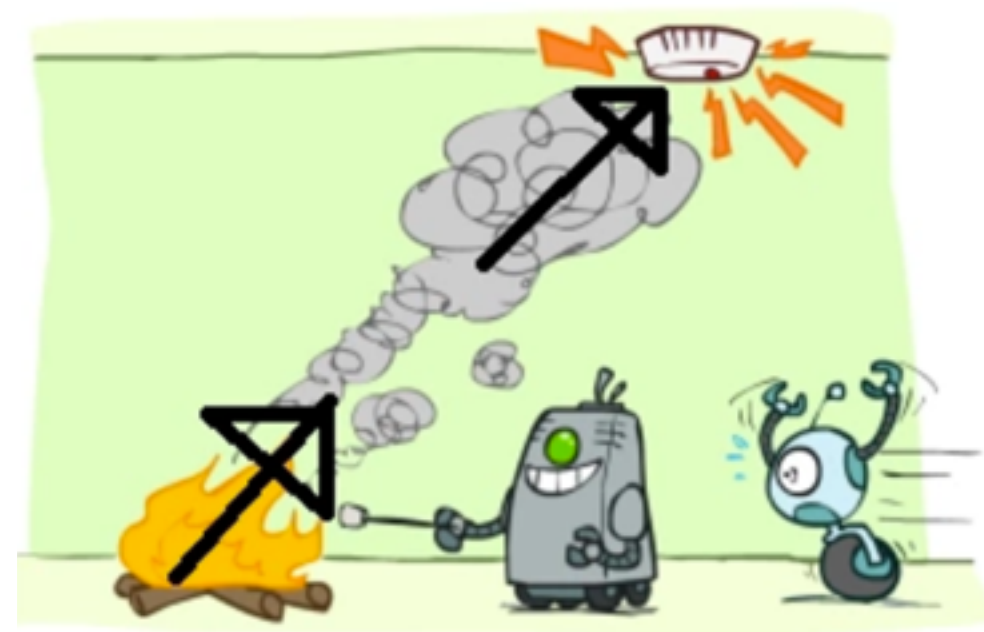
- But how to represent complicated function of many variables?
- Trivial observation for independent variables:
    - The distribution factors:

$$p(x) = p(x_1)p(x_2)\dots p(x_n)$$

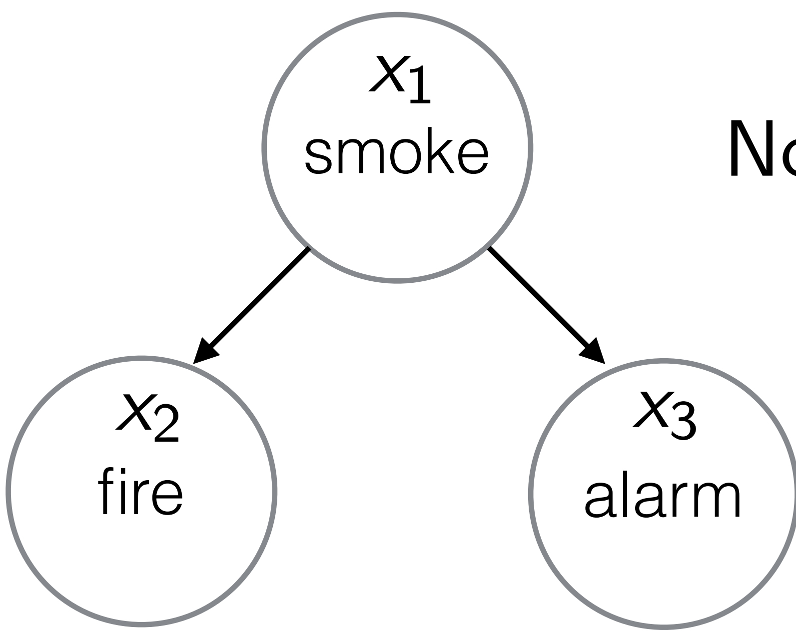
    - it is easy to evaluate maximize or integrate
- Something in between?

Conditional Independence

- Conditional Independence
    - Example: smoke, fire, alarm
    - all 3 correlated, but
    - smoke => fire and alarm are independent



- Conveniently represented in a graph diagram



No edge = conditionally independent

$$p(x_2, x_3 | x_1) = p(x_2 \mid x_1) p(x_3 | x_1)$$

$$p(x_2 | x_1, x_3) = p(x_2 \mid x_1)$$

- Factorization: $p(x_1, x_2, x_3) = p(x_2 \mid x_1) p(x_3 | x_1) p(x_1)$
- A directed graphical model (Bayes Network)

In Images

- A region is independent of the rest given some neighborhood

Random Field

- Collection of random variables

$$x_1, x_2, \ldots x_n, \qquad x_i \in D$$

Definition

$p \colon D^n \to \mathbb{R}$ is a *random field* if $p(x) > 0 \; \forall x$, $\sum_x p(x) = 1$.

- Non-negativity is important for existence of conditional probabilities and other good reasons. Practically not a limitation.

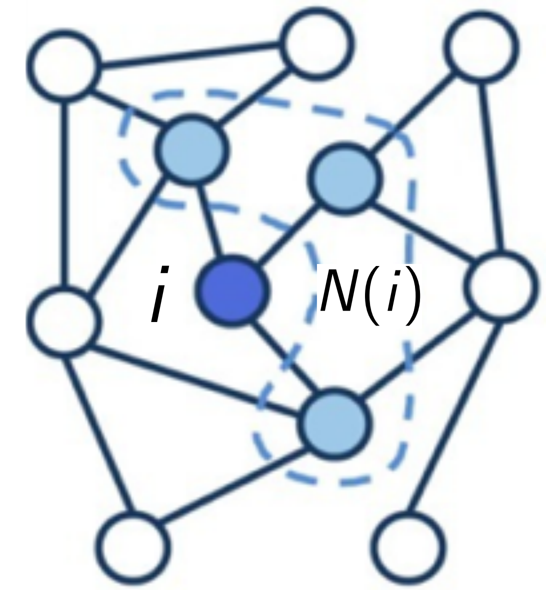
Definition

p is a *Markov random field* if it satisfies one or more conditional independence (Markov) properties.

(Book: Lauritezen S.L.,"Graphical Models", 1996)

Undirected Graphical Model

- Undirected Graphical Model
  - Graph $G = (V, E)$
  - Set of nodes V: random variables x_i, $i \in V$
  - Set of edges E



- Local Markov Property w.r.t. G:
  - Given neighbors of x_i, it is independent of the rest:

$$p(x_i \mid x_{V \setminus i}) = p(x_i \mid x_{N(i)}), \forall i \in \mathcal{V}$$

- Pairwise Markov Property w.r.t. G:
  - Absent edge (i, j) in G iff x_i and x_j are conditionally independent given the rest of variables.

Theorem (Lauritzen 96)

Local and Pairwise Markov Properties are equivalent.

Definition

MRF w.r.t. graph G is a random field satisfying Markov property w.r.t. G

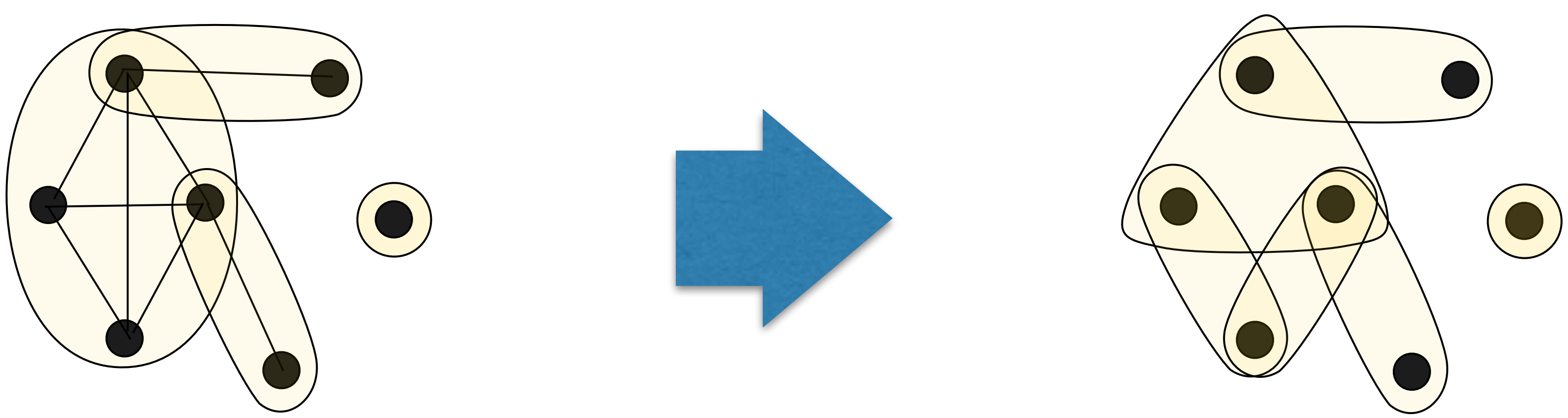
MRF factorization

- Conditional independencies help to structure and simplify the distribution

Theorem (Hammersley-Clifford,1971)

MRF p w.r.t. graph G factors over cliques of G: $p(x) = \prod_{c \in C} f_c(x_c)$,

- C is the set of cliques – maximal fully connected subgraphs



- Only factorization matters for representation tractability and inference

Definition

p is a *Gibbs Random field* if it factors as $p(x) = \prod_{c \subset S} f_c(x_c)$,

- A generalization, $S \subset 2^V$
- Here we do not need c to be a clique in some graph

Maximum *a posteriori*

- Given the model $p(x) = \prod_{c \in S} f_c(x_c)$ find the most probable state:

$$\max_x p(x)$$

- Joint maximization in all variables
- Take negative logarithm:

$$\min_x \sum_{c \in S} - \log f_c(x_c) = \min_x E(x)$$

- Partially separable minimization problem (Energy minimization)
- Converted to optimization domain (ILP, maximum cut, submodular function minimization, relaxations)
- Gibbs distribution: $p(x) = exp(-E(x))$ – physics origins

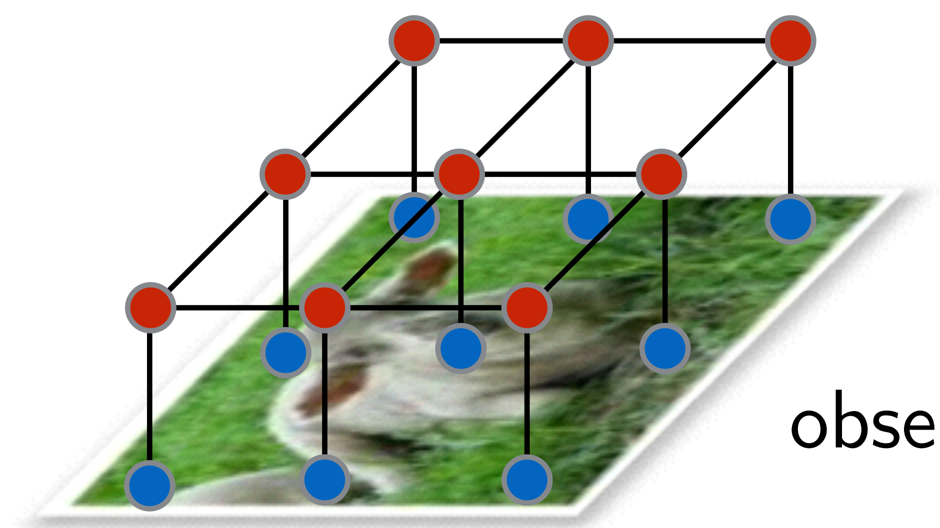
Conditional Random Field

- $x_i, \ i \in V$ - hidden random variables (segmentation)
- $y_j, \ j \in V'$ - observed random variables (Image)

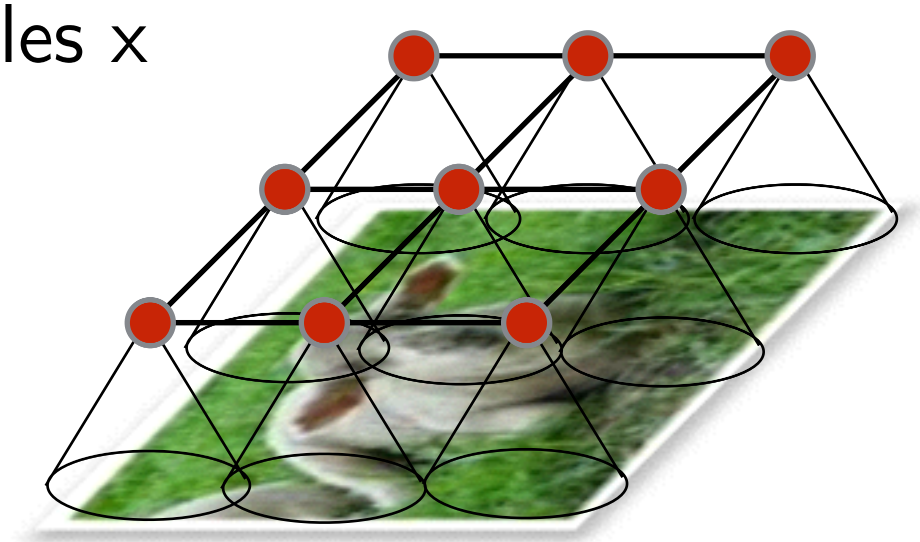
Definition (Lafferty *et al.* 01)

$p(x \mid y)$ is a conditional random field if it satisfies Markov properties w.r.t. x given y.

MRF p(x,y)



CRF p(x|y)



hidden variables x

observed variables y

Generative: $p(y) = \sum_x p(x, y)$

can be learned unsupervised

Discriminative, no model of p(y)

more flexible for recognition

Recognition is the same: $\mathrm{argmin}_x \, p(x, y) = \mathrm{argmin}_x \, p(x \mid y)$

Energy Minimization

Energy Minimization

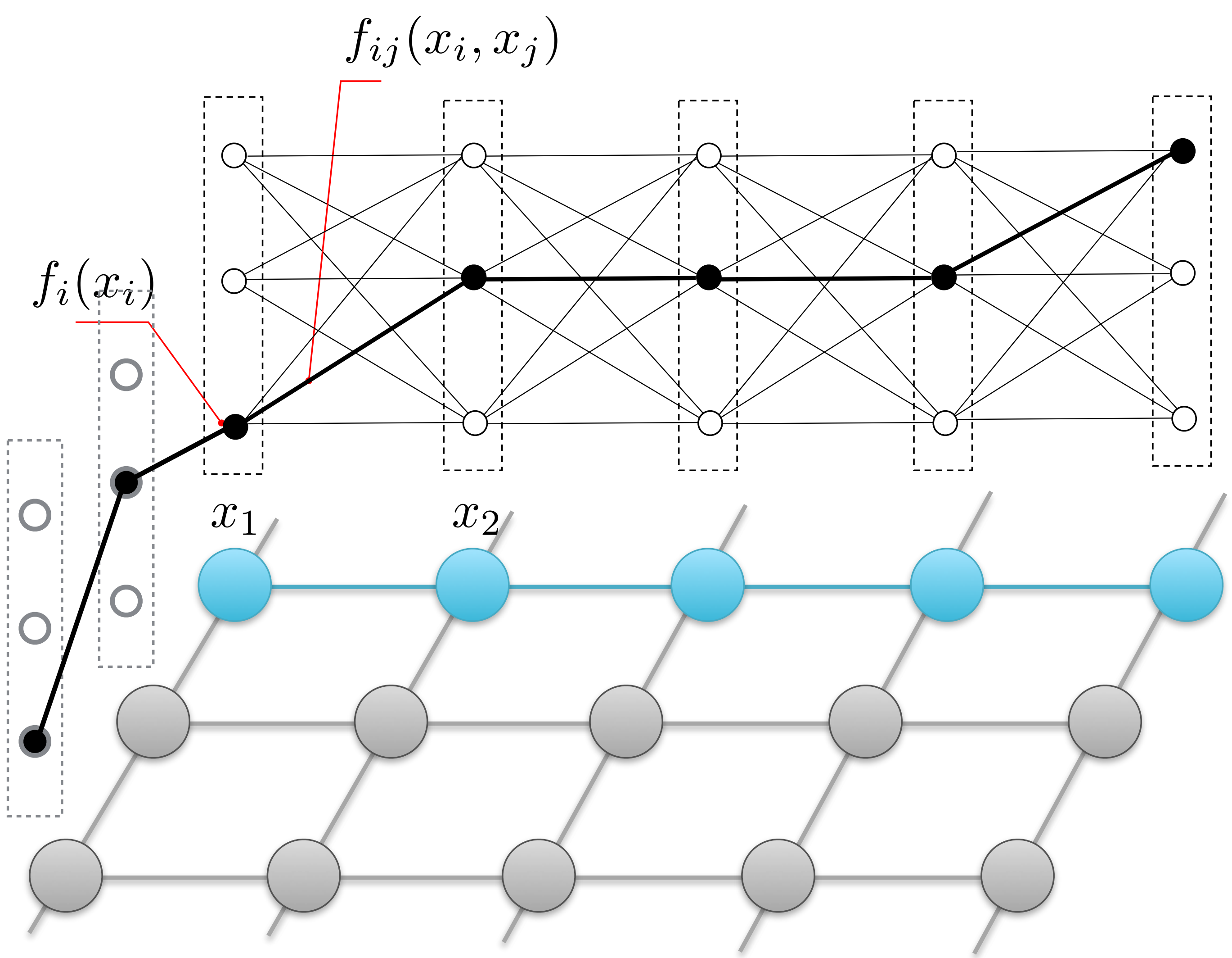
$$\min_x \sum_{i \in \mathcal{V}} f_i(x_i) + \sum_{ij \in \mathcal{E}} f_{ij}(x_i, x_j)$$

$(\mathcal{V}, \mathcal{E})$ - graph

$\mathcal{V}$ - set of nodes

$\mathcal{E}$ - set of edges

$x = (x_i \mid i \in \mathcal{V})$ - labeling



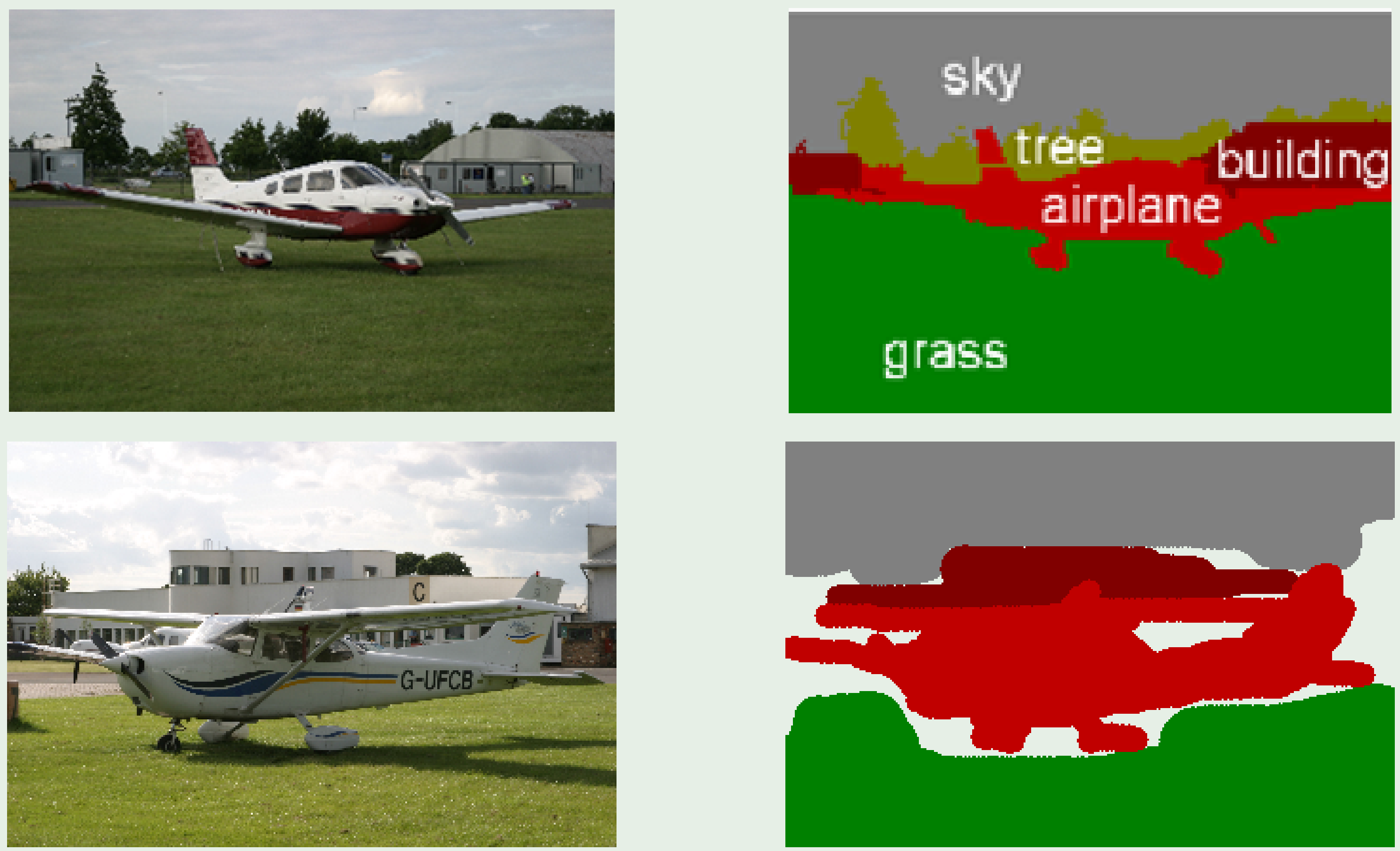
- NP-hard (includes MAX-CUT, vertex packing, etc.)
- exp-APX-complete (approximation-preserving reduction from WSAT)
- Two large groups of methods:
  - minimum cut (graph cuts)
  - LP relaxation / message passing
- There are much more

Example: segmentation

Example: Potts Model for Object Class Segmentation

- $\mathcal{V}$ - set of pixels; $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ neighboring pixels;
- $\mathcal{X}_s = \{1, \dots K\}$ – class label;
- $E_f(x) = \sum_{s \in \mathcal{V}} f_s(x_s) + \sum_{st \in \mathcal{E}} \lambda_{st} [\![x_s \neq x_t]\!]$.

Image Ground Truth



(MSRC object class segmentation)

Minimum Cut

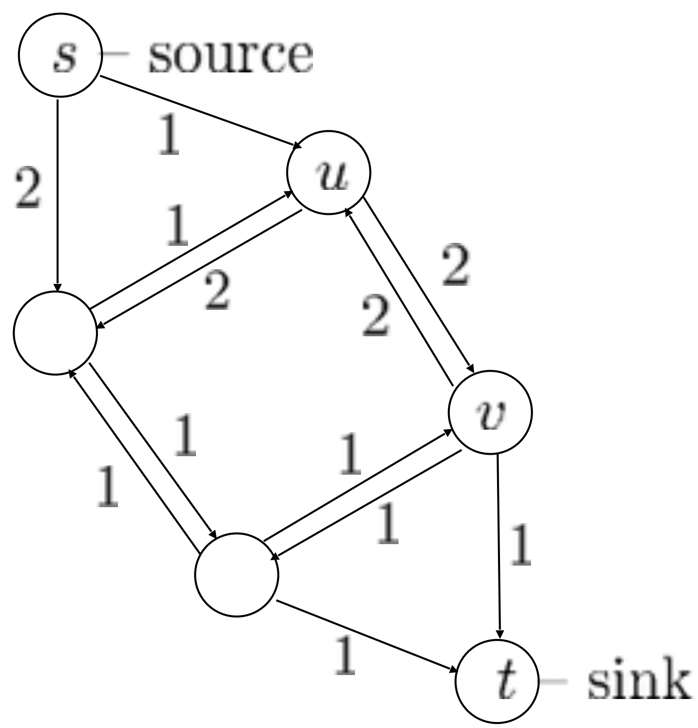
- Minimum s-t cut problem

Capacitated network

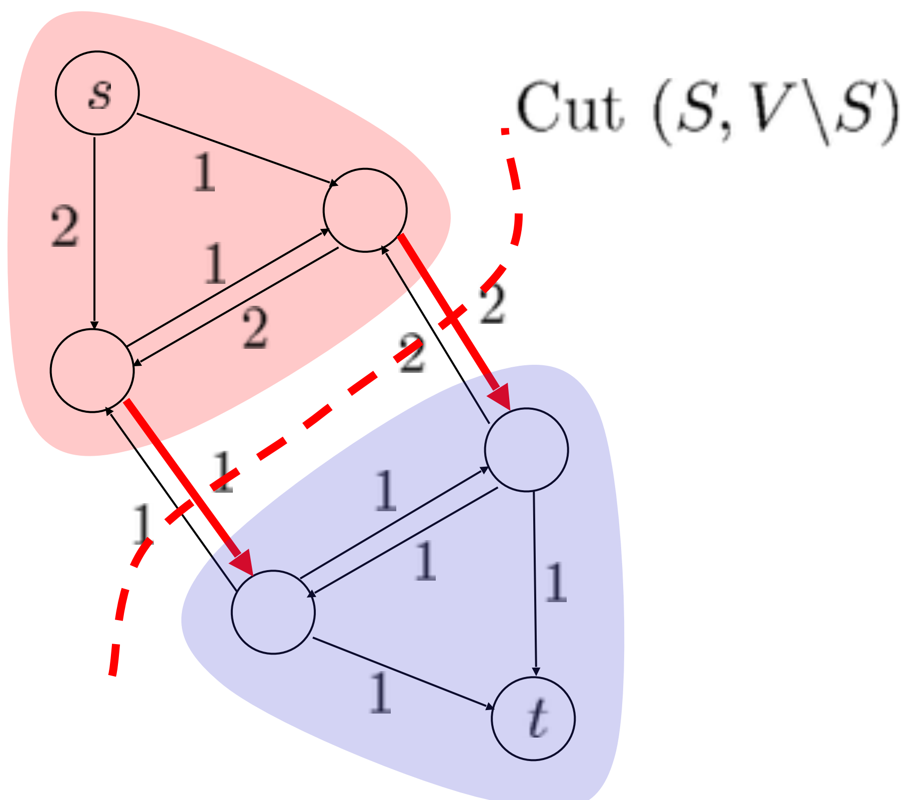
$G = (V, E, c),$

$c(u, v) \geq 0$ – arc capacities

Cut cost: $\sum_{\substack{(u,v)\in E \\ u\in S \\ v\notin S}} c(u, v) \rightarrow \min_{\substack{S \\ s\in S \\ t\notin S}}$



Source set S



Sink set $T = V\backslash S$

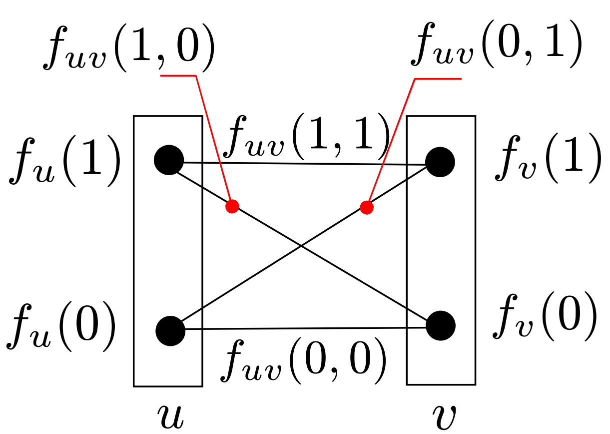
- Problem history: 30+ years
- Active research for better algorithms:
  - theoretical (Orlin'12: O(mn) algorithm), parallel algorithms
  - practical, esp. in computer vision

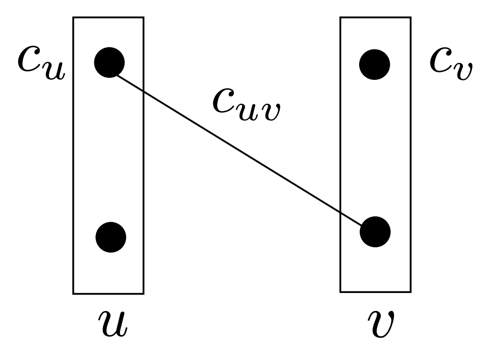
Minimum Cut

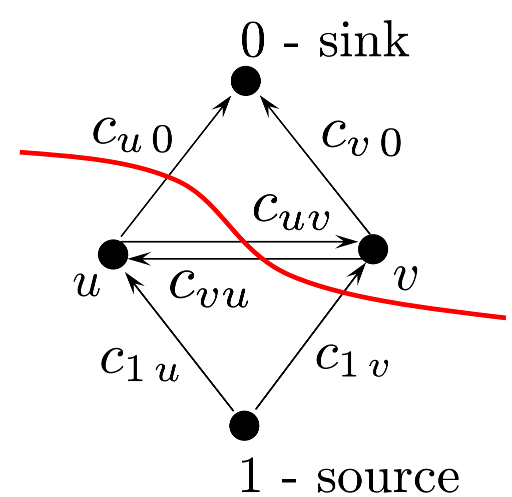
- Let $x_i \in \{0, 1\}$
- Energy minimization: $\min_x \sum_{i \in \mathcal{V}} f_i(x_i) + \sum_{ij \in \mathcal{E}} f_{ij}(x_i, x_j)$
- Expand as polynomial:

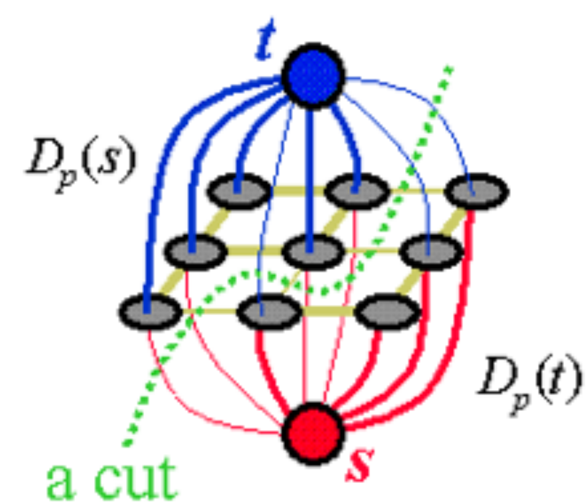
$$
\begin{aligned}
f_i(x_i) = f_i(1)x_i + f_i(0)(1 - x_i) \qquad &= c_0 + c_i x_i; \\
f_{ij}(x_i, x_j) = \dots \qquad &= c_0' + c_i' x_i + c_j'' x_j + c_{ij} x_i (1 - x_j).
\end{aligned}
$$

- Minimum cut: $\min_{S \subset V} \sum_{ij \in (S, V \setminus S)} c_{ij}$





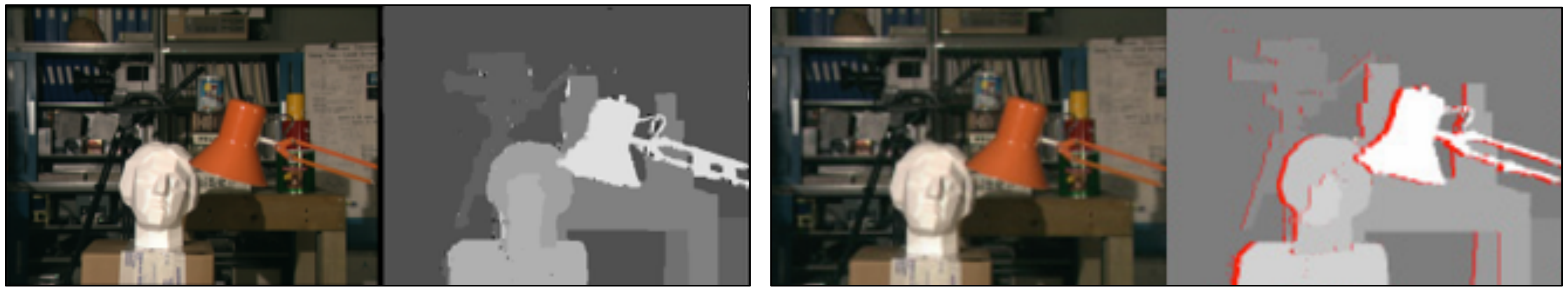




- Solvable in polynomial time if c_uv >=0

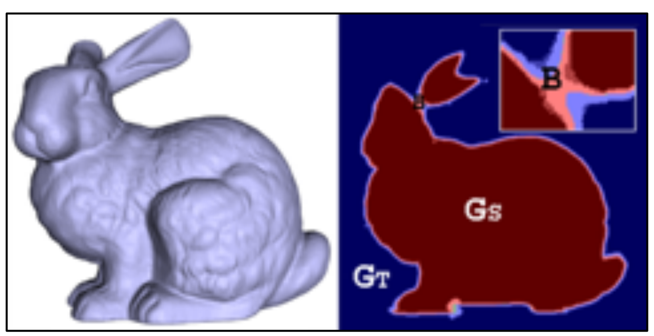
Applications of min-cut

- Exemplar applications

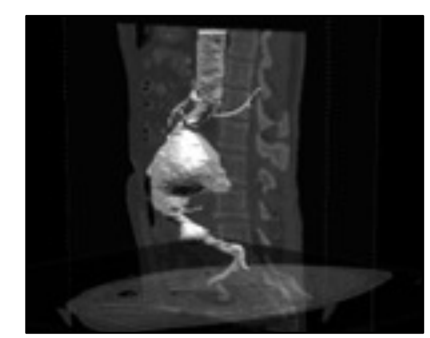


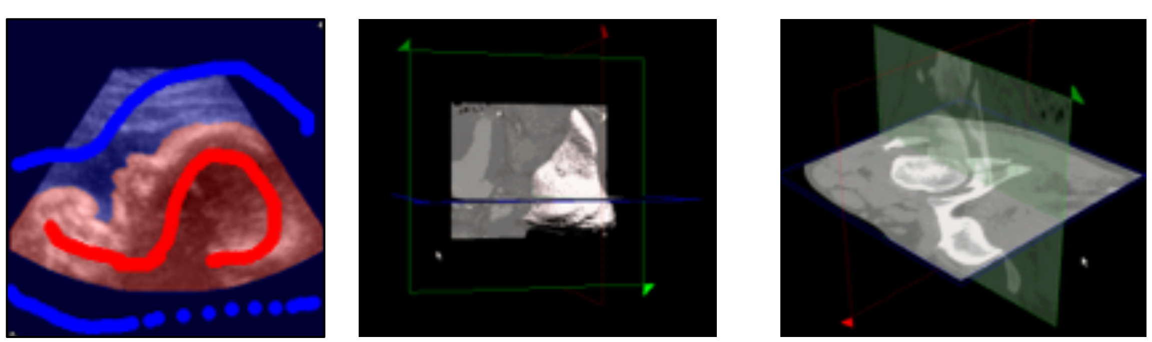
Stereo
Boykov et al. 1998
Kolmogorov and Zabih 2001

Multiview Reconstruction
Lempitsky et al. 2006
Boykov and Lempitsky 2006

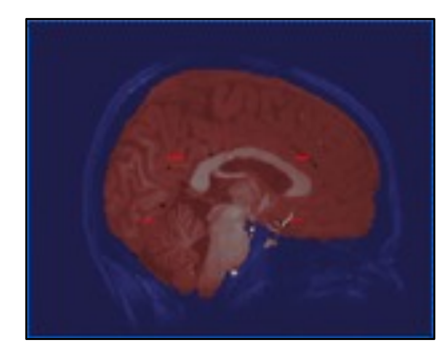


Surface Fitting
Lempitsky and Boykov 2007





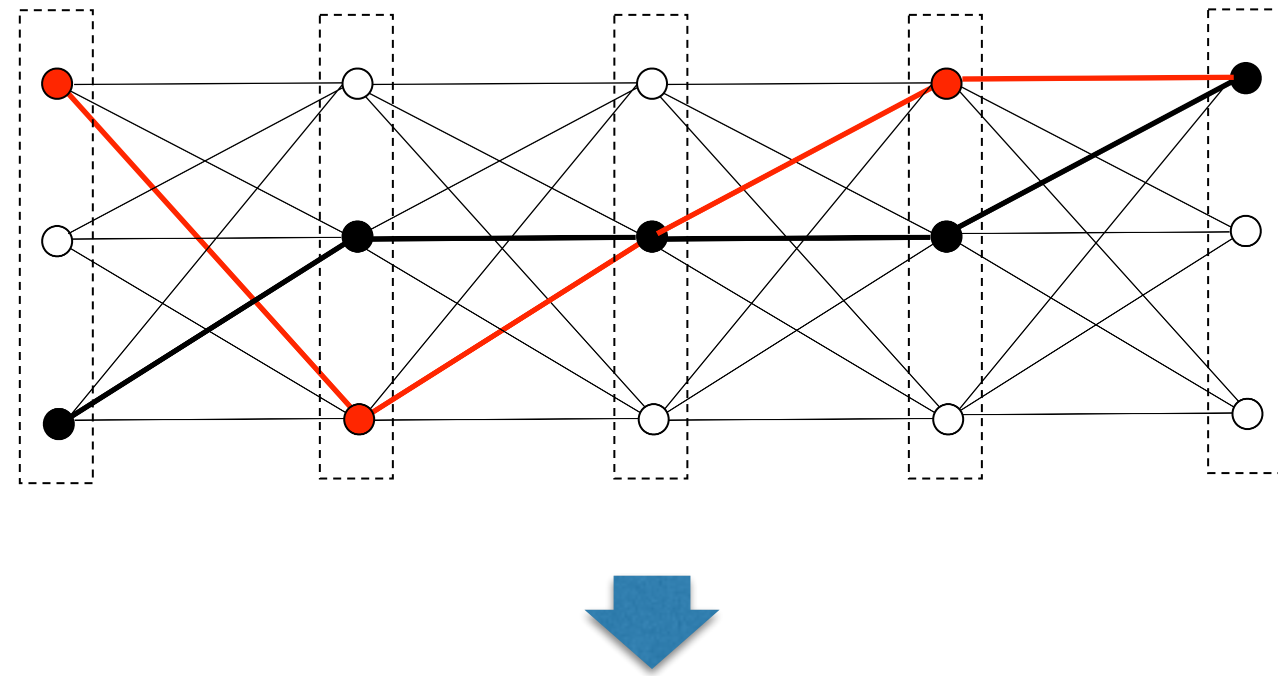
3D Segmentation
Boykov and Joly 2001
Boykov and Funka-Lea 2006
Boykov and Kolmogorov 2003



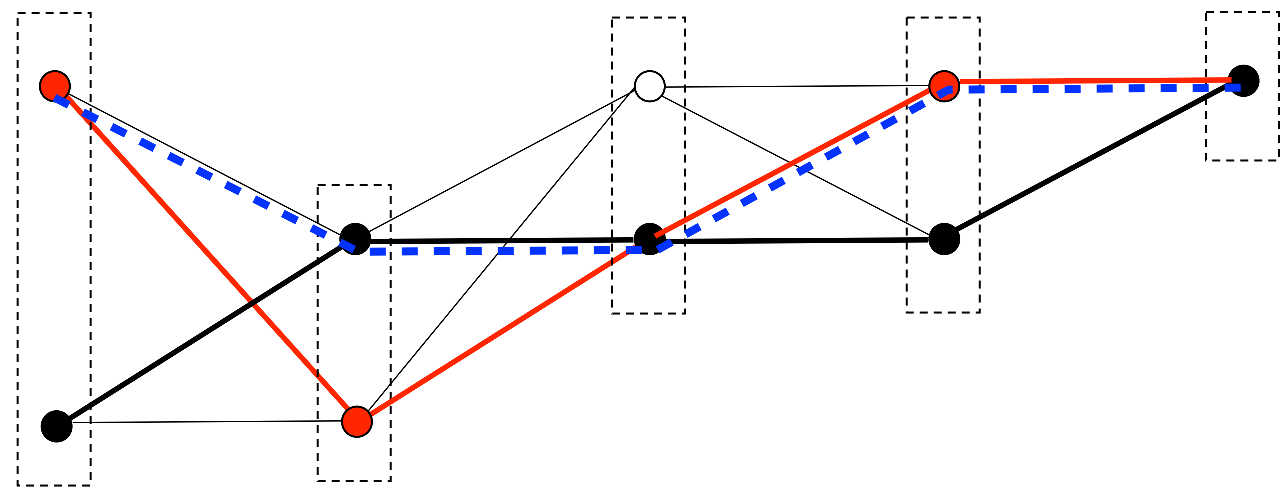
Optimized Crossover

Current best solution x

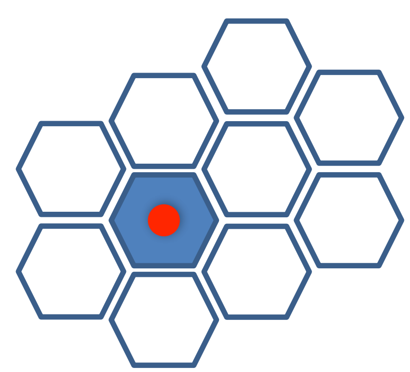
Proposal solution y



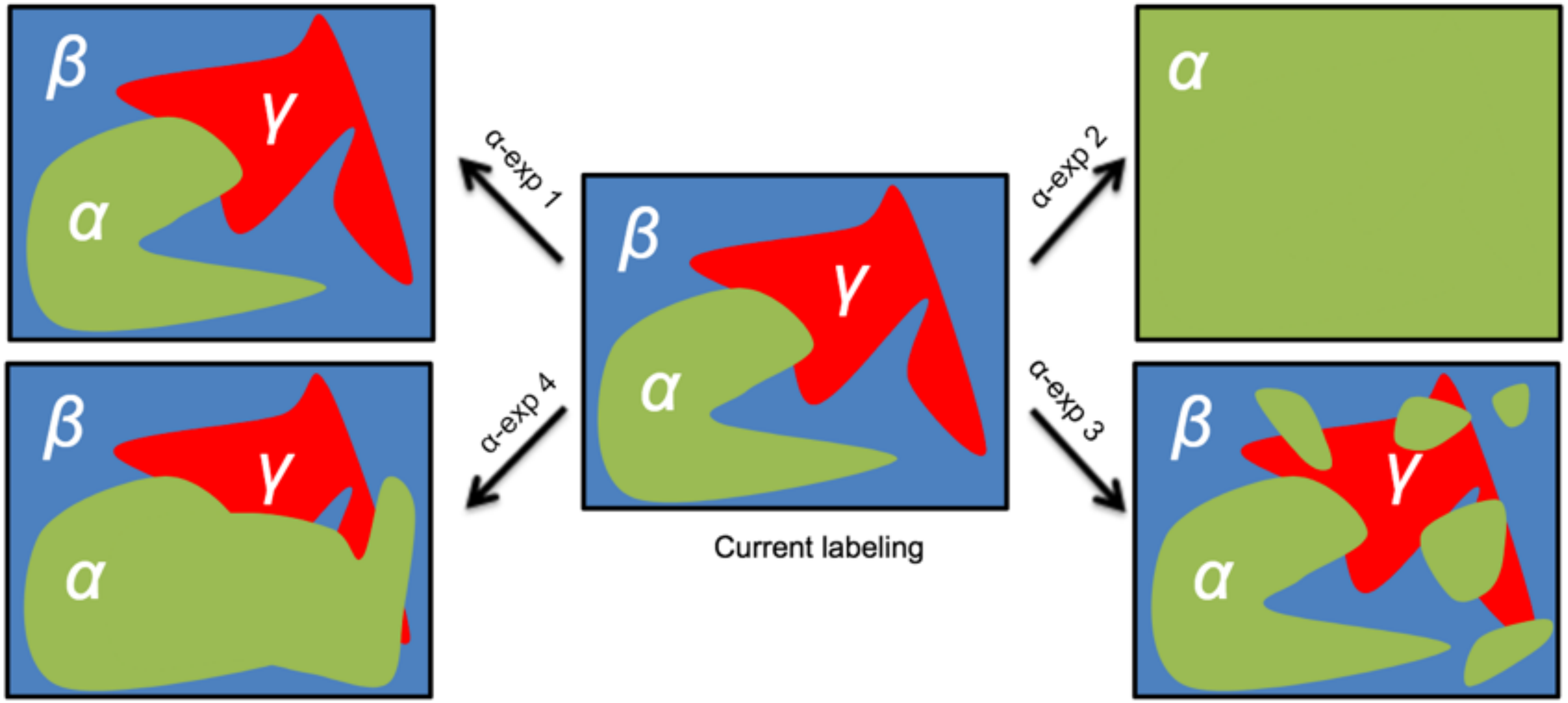
Crossover (fusion problem) x y



Local Search in some combinatorial locality



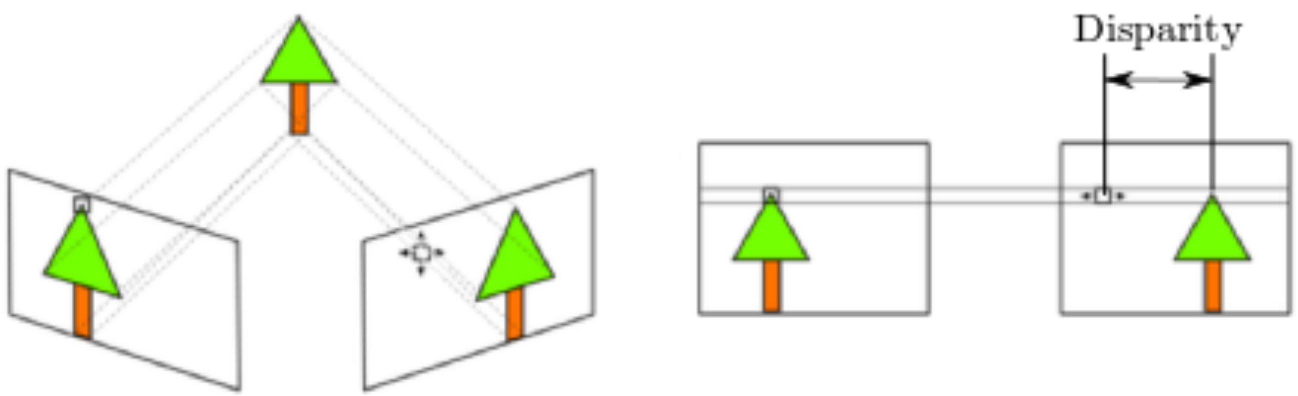
Expansion Move

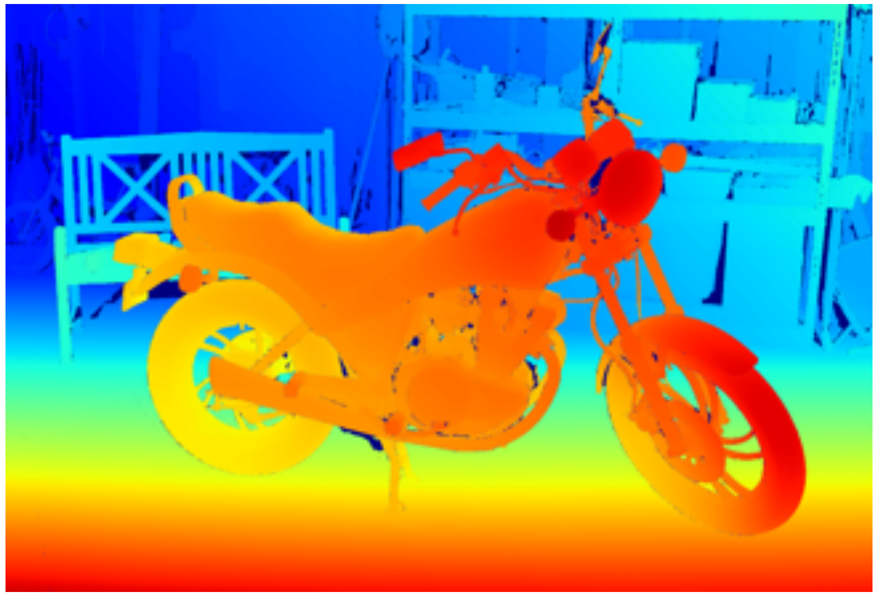


Example: Stereo Reconstruction

- Input
  - Two images from a calibrated camera pair
  - Rectified: epipolar lines correspond to image rows

Input Pair

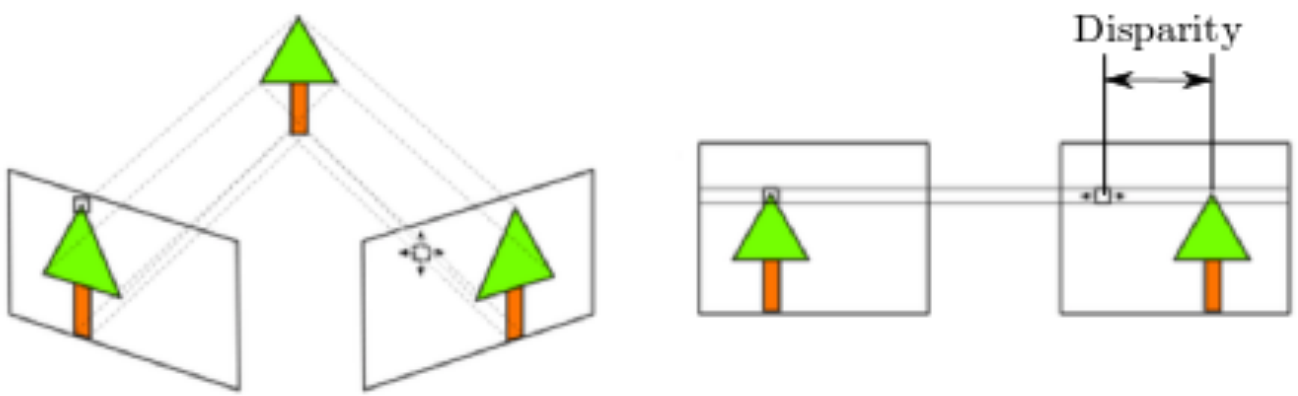




Disparity Map (GT)

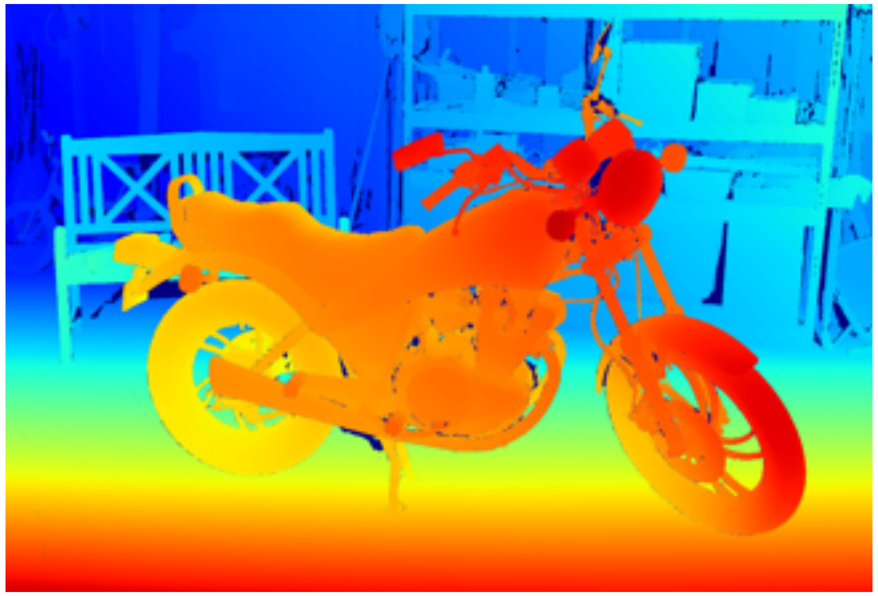
Example: Stereo Reconstruction

- Input
  - Two images from a calibrated camera pair
  - Rectified: epipolar lines correspond to image rows



Input Pair

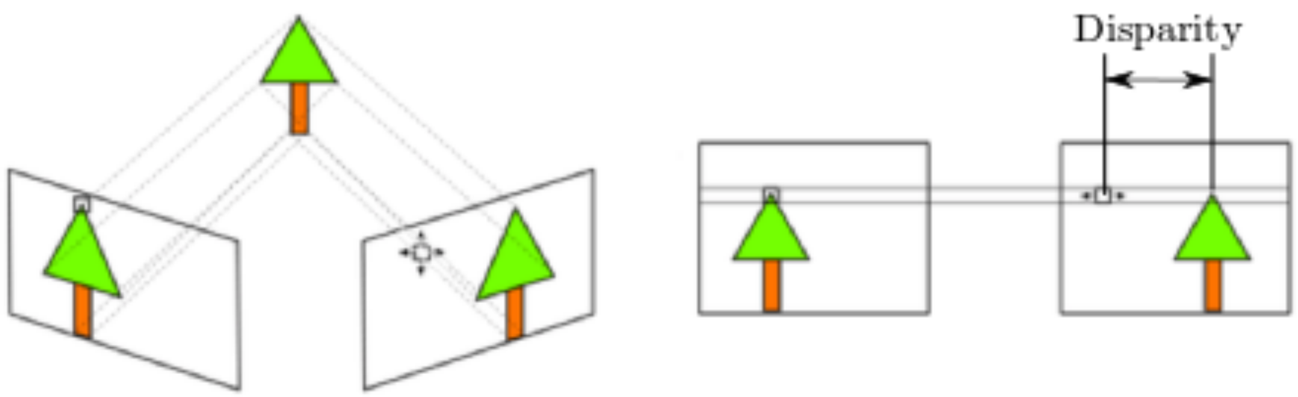
- Problem



Disparity Map (GT)

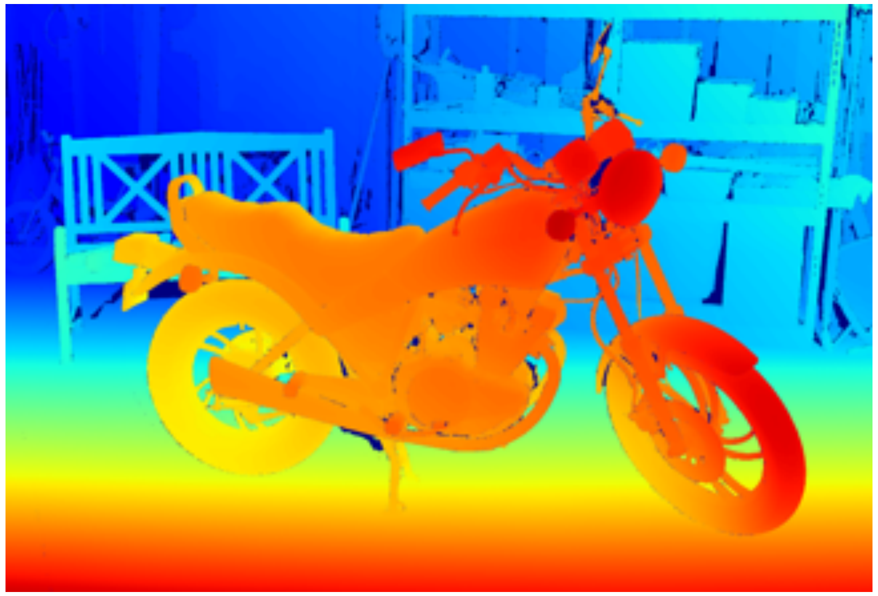
Example: Stereo Reconstruction

- Input
  - Two images from a calibrated camera pair
  - Rectified: epipolar lines correspond to image rows



- Problem
  - For each pixel in the left image find the corresponding pixel in the right image

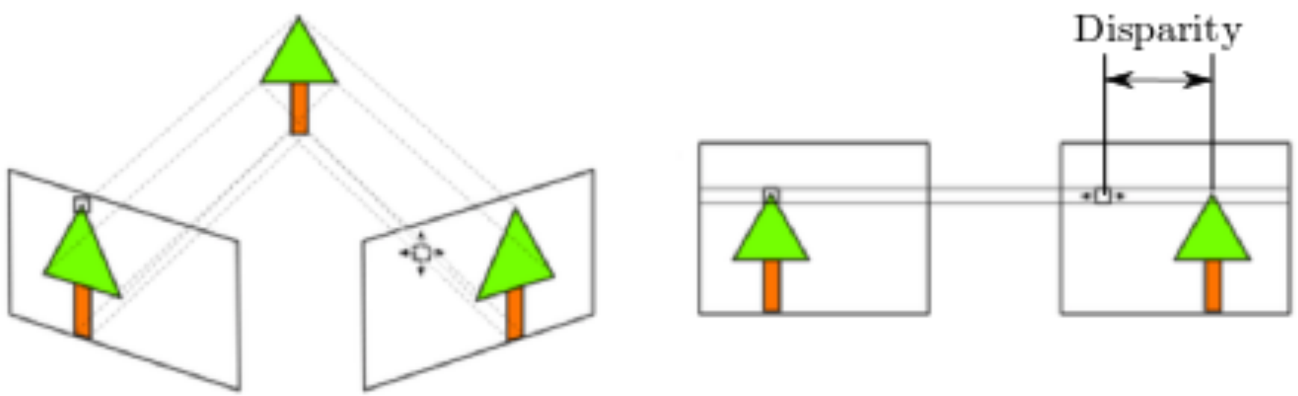
Input Pair



Disparity Map (GT)

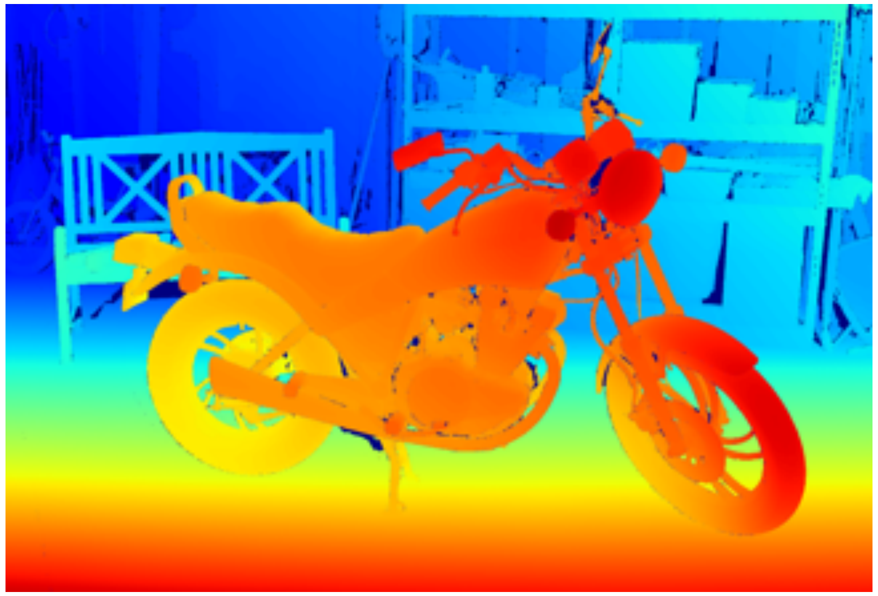
Example: Stereo Reconstruction

- Input
  - Two images from a calibrated camera pair
  - Rectified: epipolar lines correspond to image rows



- Problem
  - For each pixel in the left image find the corresponding pixel in the right image

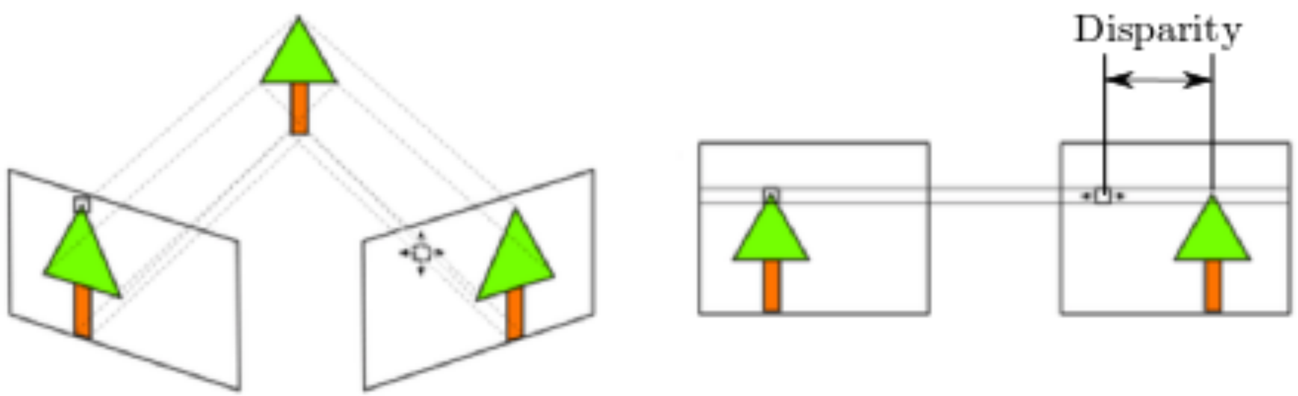
Input Pair



Disparity Map (GT)

Example: Stereo Reconstruction

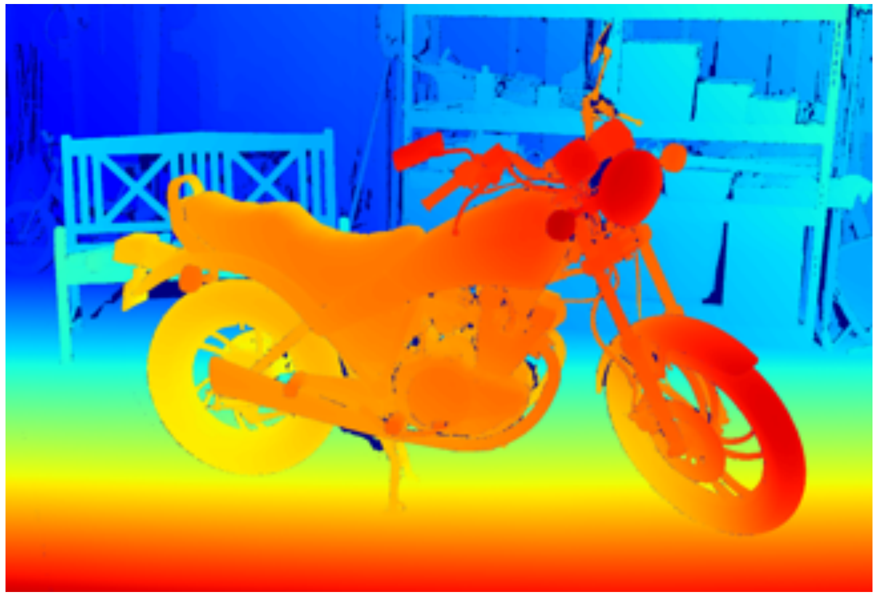
- Input
  - Two images from a calibrated camera pair
  - Rectified: epipolar lines correspond to image rows



- Problem
  - For each pixel in the left image find the corresponding pixel in the right image

- Output

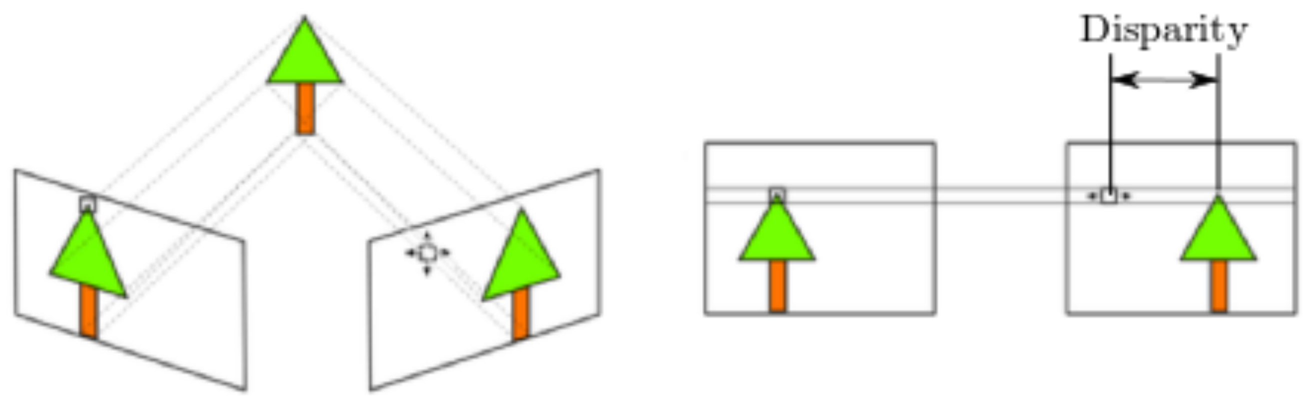
Input Pair



Disparity Map (GT)

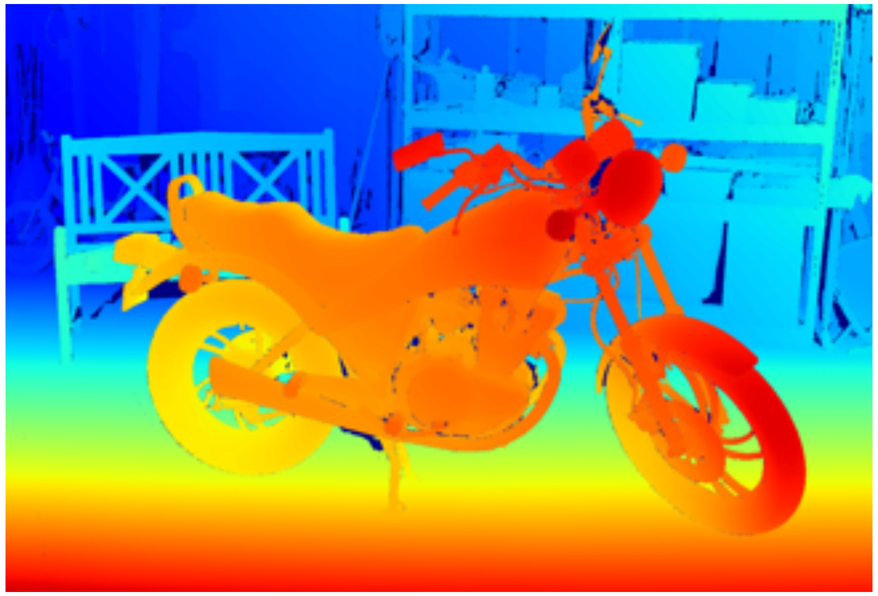
Example: Stereo Reconstruction

- Input
  - Two images from a calibrated camera pair
  - Rectified: epipolar lines correspond to image rows



Input Pair

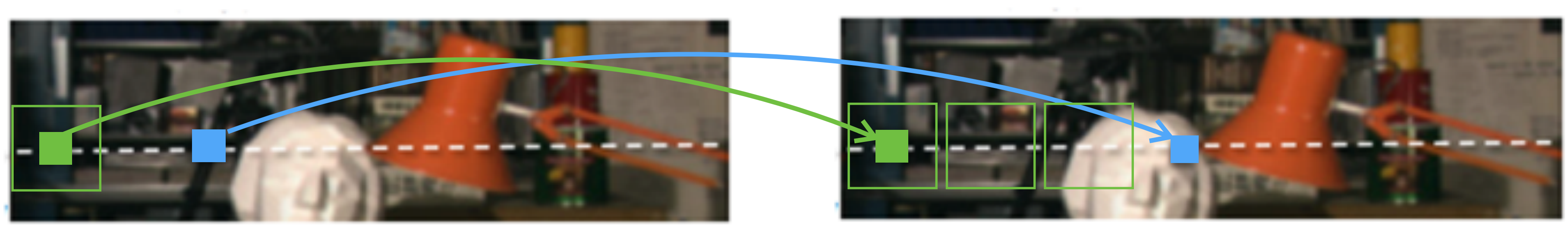
- Problem
  - For each pixel in the left image find the corresponding pixel in the right image

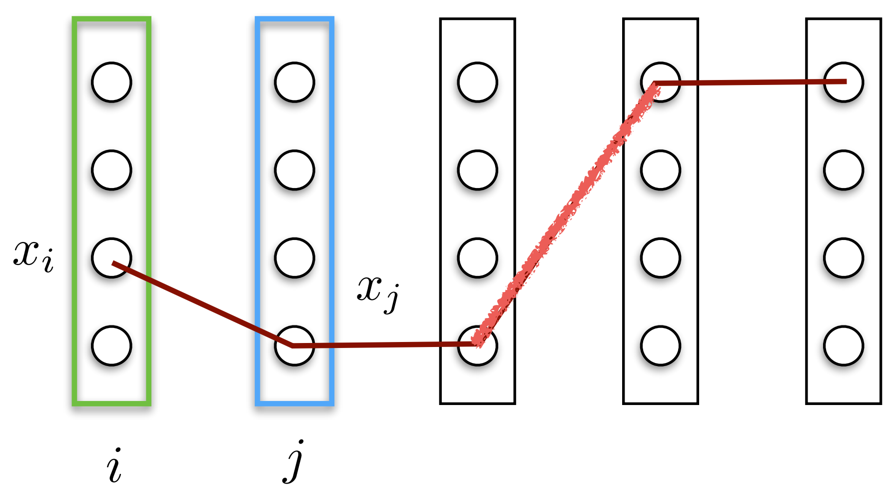


Disparity Map (GT)

- Output
  - Dense depth (disparity) map

Example: Scan-line Stereo





i - pixel

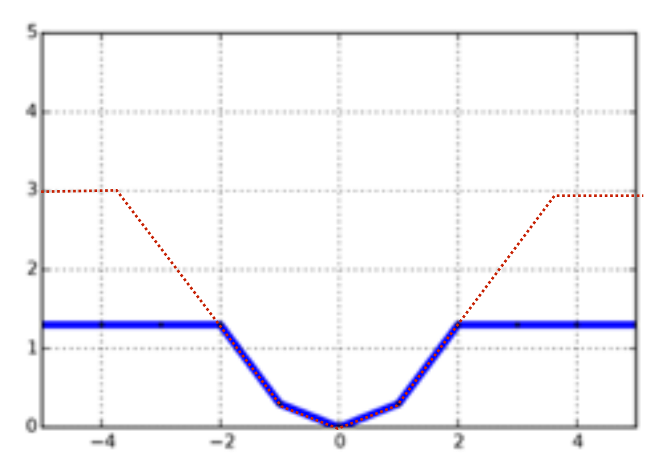
x_i - chosen disparity label

$x = (x_i \mid i \in \mathcal{V})$ - labeling

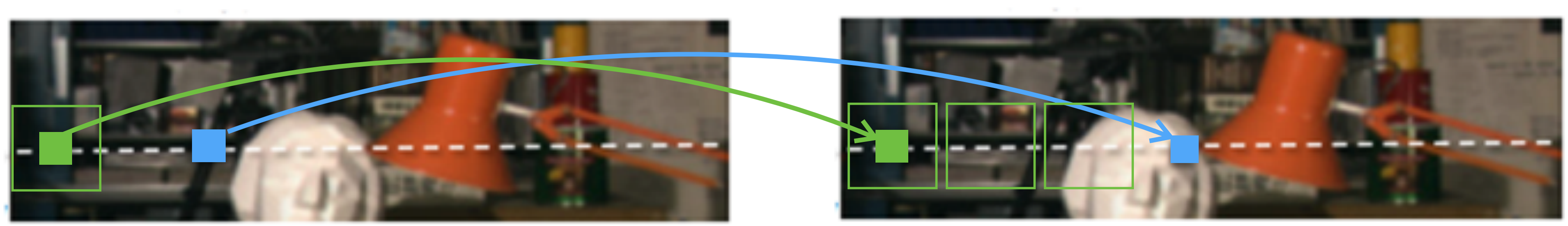
$f_i(x_i)$ - matching cost

$f_{ij}(x_i, x_j)$ - smoothness cost

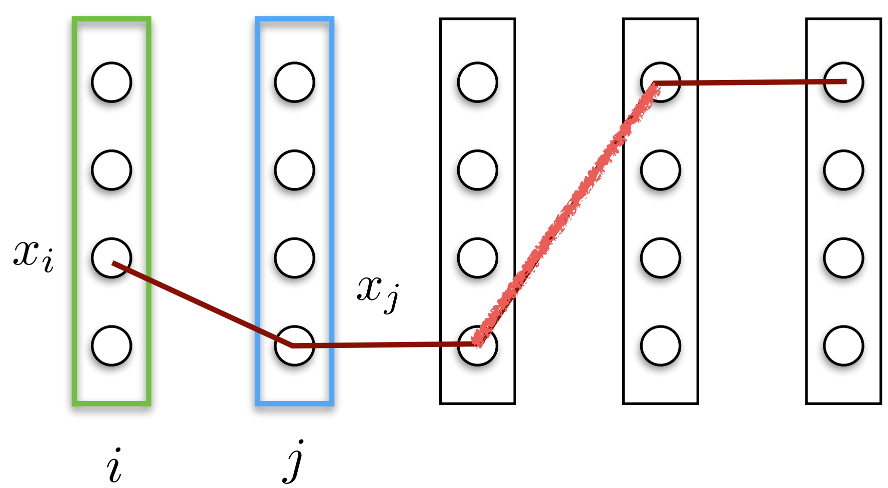
$$\min_x \sum_{i \in \mathcal{V}} f_i(x_i) + \sum_{ij \in \mathcal{E}} f_{ij}(x_i, x_j)$$



Example: Scan-line Stereo



- Trellis graph



i - pixel

x_i - chosen disparity label

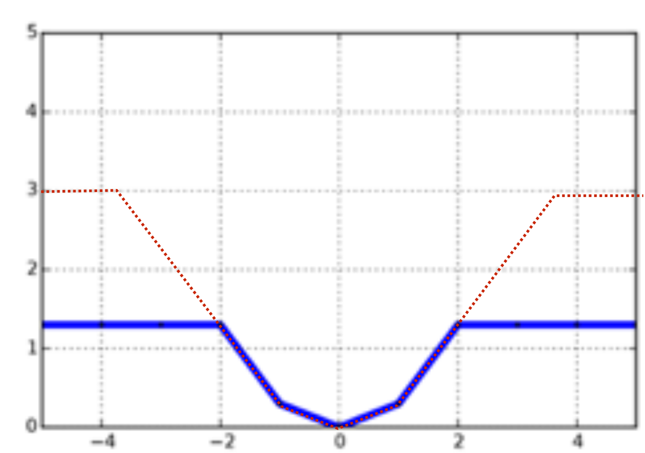
$x = (x_i \mid i \in \mathcal{V})$ - labeling

$f_i(x_i)$ - matching cost

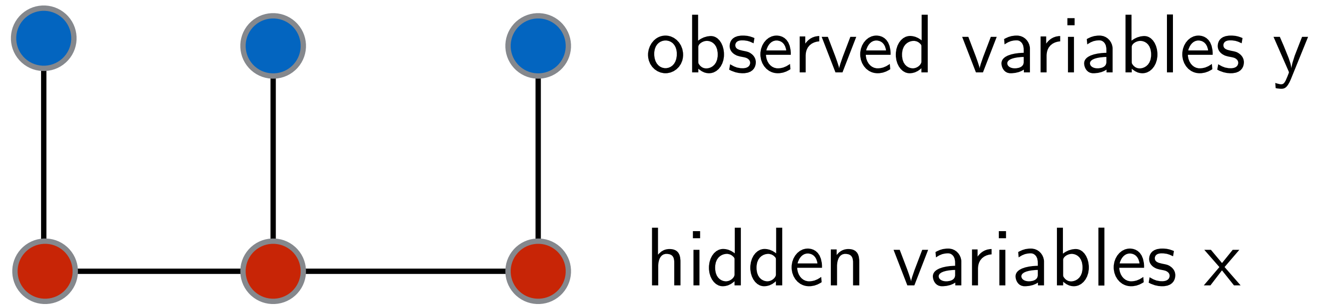
$f_{ij}(x_i, x_j)$ - smoothness cost

- Energy minimization

$$\min_x \sum_{i \in \mathcal{V}} f_i(x_i) + \sum_{ij \in \mathcal{E}} f_{ij}(x_i, x_j)$$



Hidden Markov Model



$$p(x, y) = p(x_1) \prod_{i=2}^{n} p(x_i \mid x_{i-1}) \prod_{i=1}^{n} p(y_i \mid x_i)$$

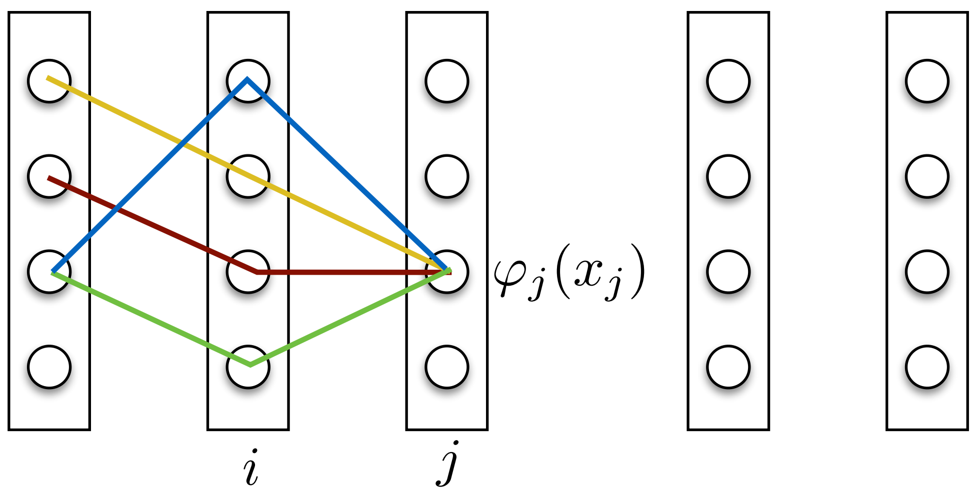
- Conditionally independent model: given x_i, y_i is independent of everything else.
- Recognition (MAP):

$$\operatorname*{argmax}_x p(x, y) = \operatorname*{argmin}_x \sum_i f_i(x_i) + \sum_{i=2}^{n} f_{i-1,i}(x_{i-1}, x_i)$$
$$f_i(x_i) = -\log p(y_i \mid x_i)$$

Viterbi Algorithm

- Problem:

$$\min_{x} \sum_{i \in \mathcal{V}} f_i(x_i) + \sum_{ij \in \mathcal{E}} f_{ij}(x_i, x_j)$$



- Use distributivity:

$$\min(a + c, b + c) = min(a, b) + c$$

$$\min_{x_n} \left(\cdots + \min_{x_2} \left(f_{2,3}(x_2, x_3) + f_2(x_2) + \min_{x_1} \left(f_{1,2}(x_1, x_2) + f_1(x_1) \right) \right) \right)$$

- Recurrent update:

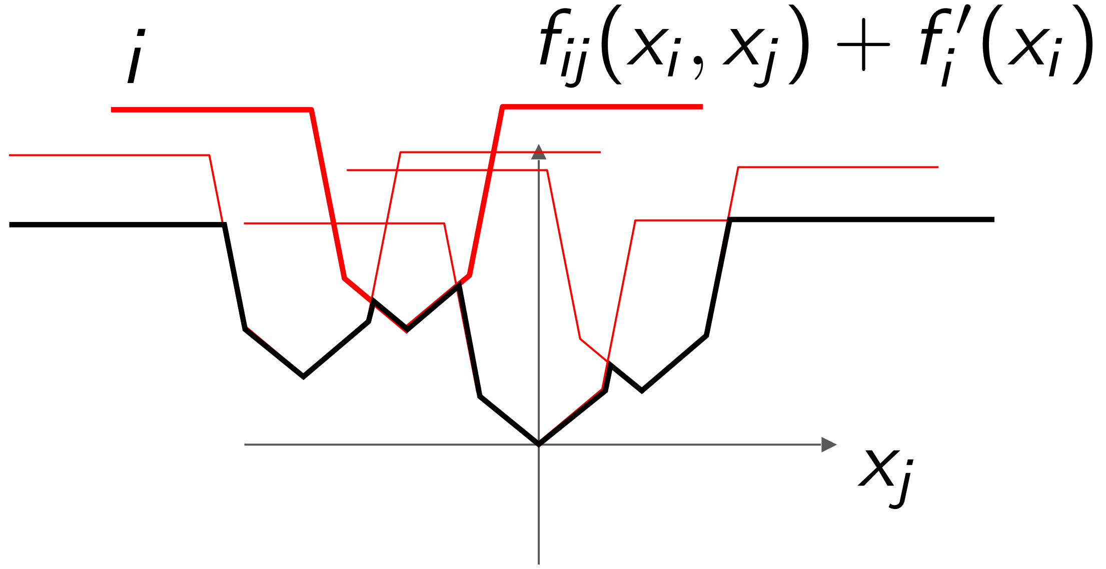
$$\varphi_j(x_j) = \min_{x_i} \left(f_{ij}(x_i, x_j) + f_i(x_i) + \varphi_i(x_i) \right)$$

- Shortest path from the left
- Core of all message passing algorithms

Distance Transforms

- Recurrent update (message passing):

$$\boxed{\varphi_j(x_j) = \min_{x_i}(\varphi_i(x_i) + f_i(x_i) + f_{ij}(x_i, x_j))}$$



- Lower envelope (distance transform)

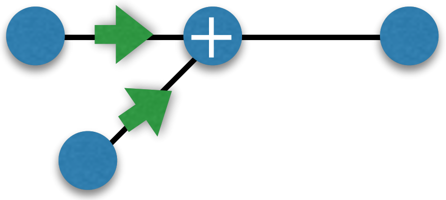
$f_{ij}(x_i, x_j) = w_{ij}\rho(x_i - x_j)$

$O(nL^2)$ - naive approach, n variables, L labels

$O(nL)$ - efficient sequential algorithms [Hirata'96, Meijster'02] [Felzenszwalb&H.'06]

$O(n \log L)$ - efficient parallel algorithms, using L processors [Goodrich'86, Chen'02]

- Extends to trees

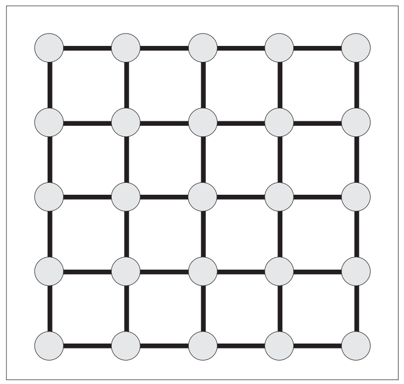


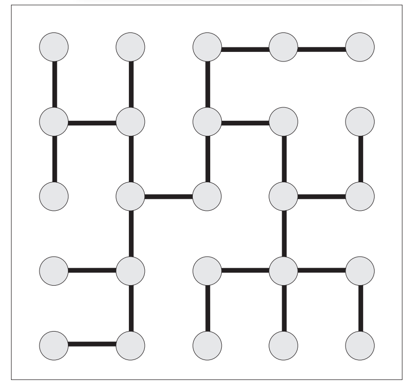
Many Heuristics for Stereo

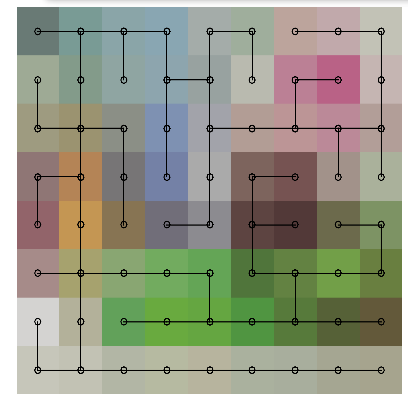
Veksler-05

Psota et al. ICCV-15

full graph

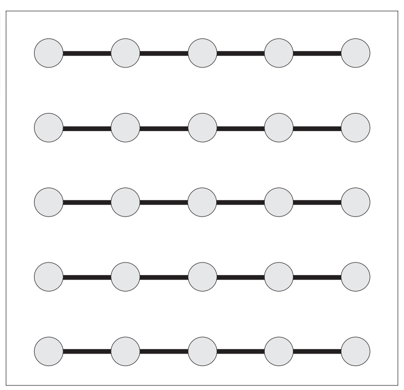


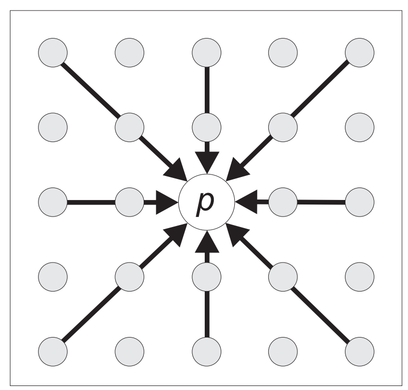




+ connect similar colors first
+ learned potentials

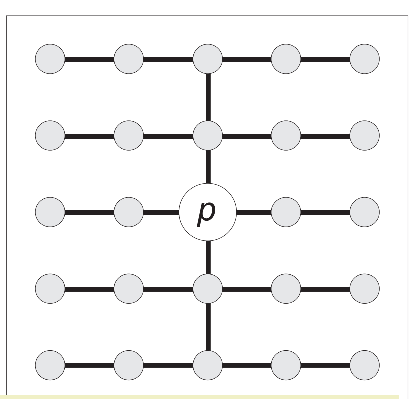
scanline DP

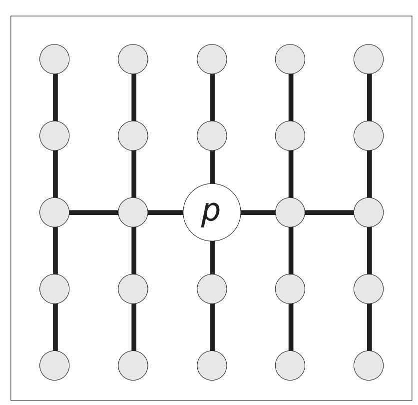


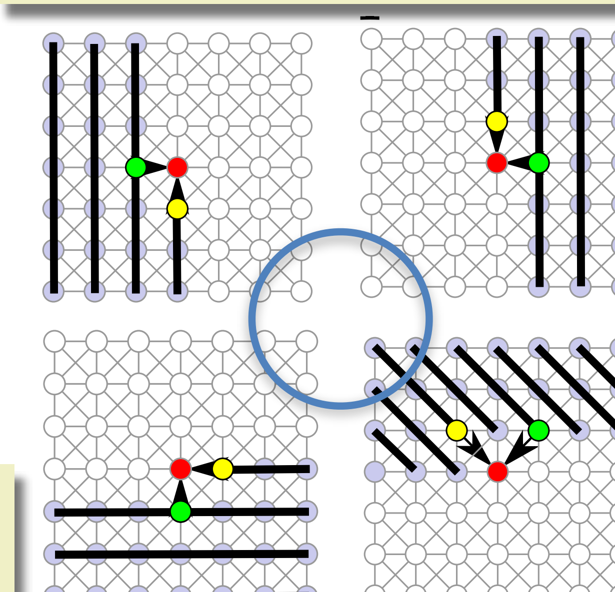


Hirschmüller-05 (SGM)
+ own tree for each pixel
+ reuse messages in DP

Facciolo et al.-15 (MGM)
+ combine more messages





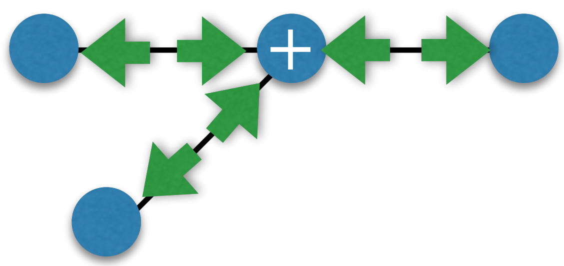


Bleyer & Gelautz-VISSAP-08
+ own tree for each pixel
+ larger coverage

Yang-15
+ with cost volume modification

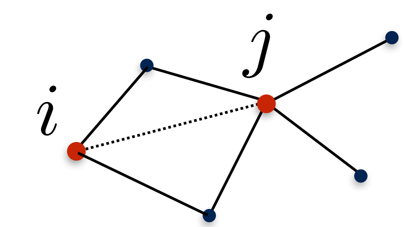
Max-Product BP, Tree-Reweighted[1]

- Can Run Message passing in parallel



$O(n)$ time, $O(n)$ processors

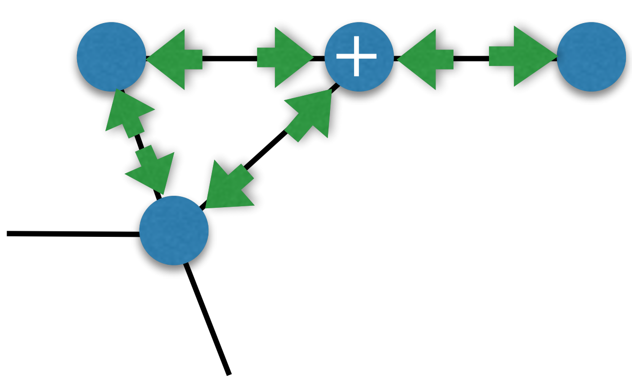
c.f. all shortest paths in a graph



$$d(i,j) := \min_k (d(i,k) + d(k,j))$$

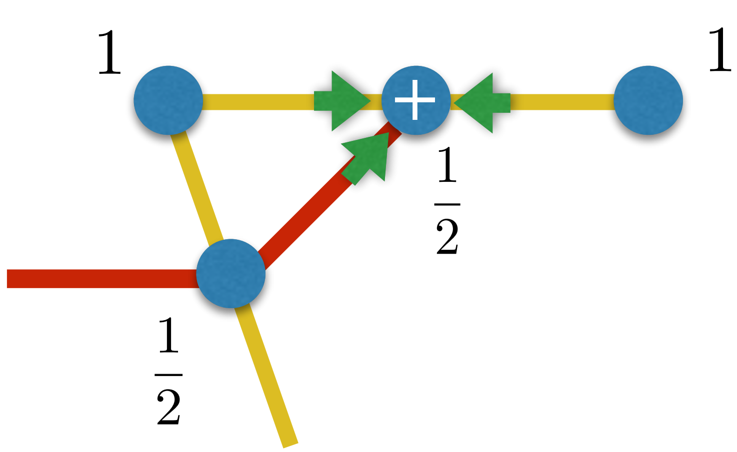
(Floyd–Warshall alg.)

- Can apply on graphs with loops (loopy BP)



  - Over-counting
  - May oscillate
  - May diverge (unbounded)

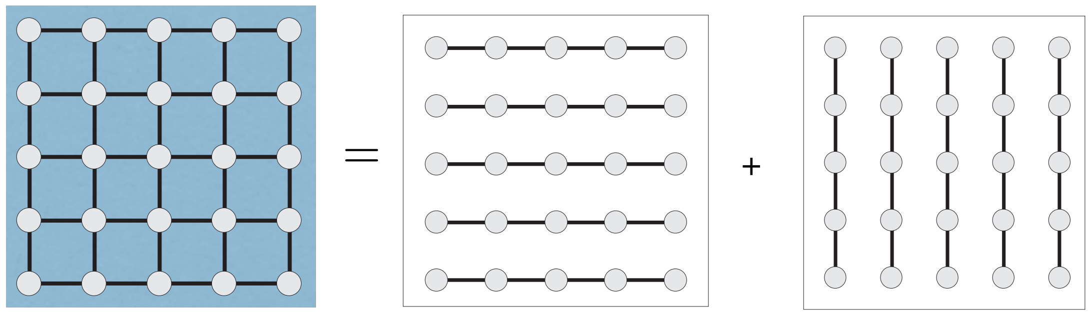
- Tree-Reweighted [Wainwright'05]



  - Decomposition into trees
  - Connection to LP relaxation and its dual
  - Parallel algorithm may still oscillate

Dual Decomposition

$$f(x) = \sum_{i \in \mathcal{V}} f_i(x_i) + \sum_{ij \in \mathcal{E}} f_{ij}(x_i, x_j)$$



Sum of chain subproblems: $f = f^1 + f^2$

$$\min_x (f^1(x) + f^2(x))$$

$$= \min_{x^1 = x^2} (f^1(x^1) + f^2(x^2))$$

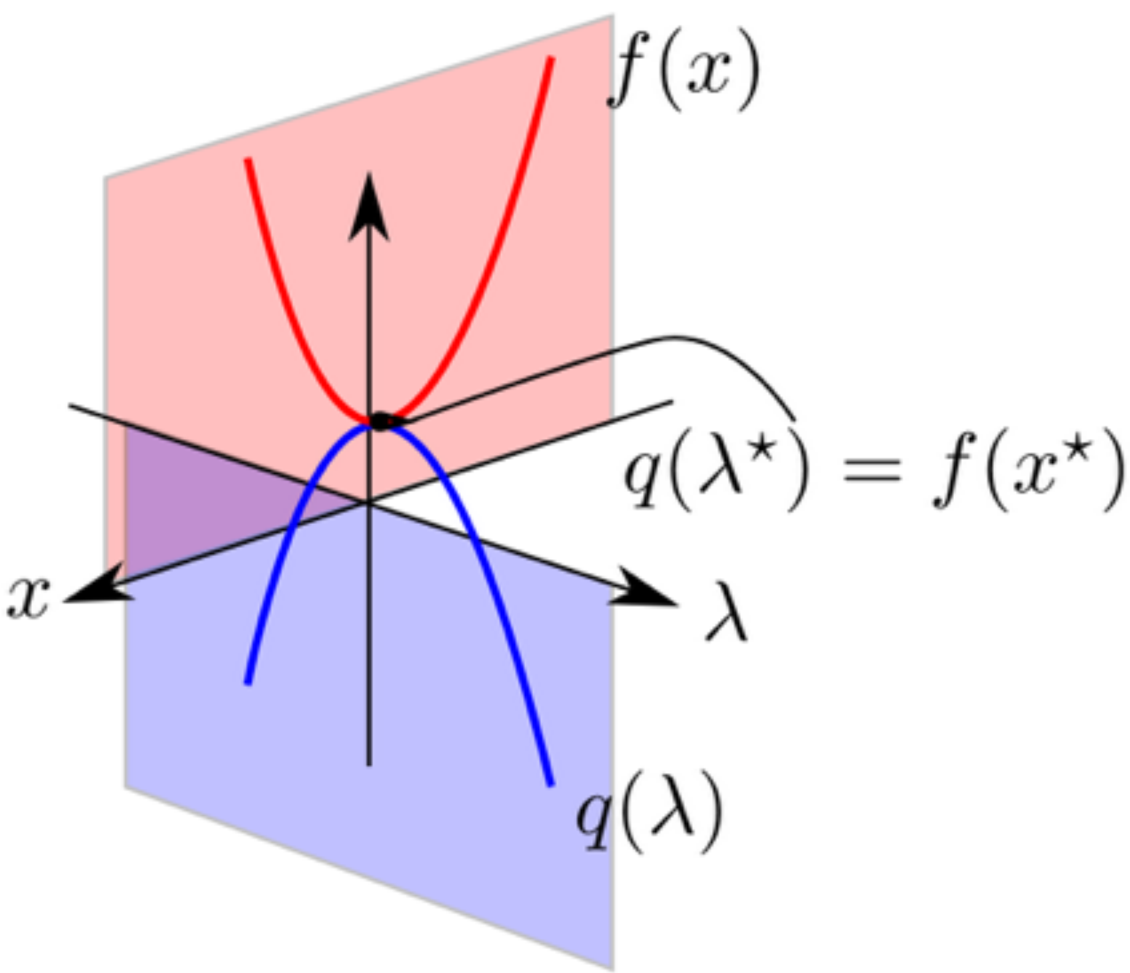
$$= \min_{x^1, x^2} \max_{\varphi} \langle \varphi, x^1 - x^2 \rangle + f^1(x^1) + f^2(x^2)$$

$$\geq \max_{\varphi} [\min_{x^1} (f^1 + \varphi)(x^1) + \min_{x^2} (f^2 - \varphi)(x^2)]$$

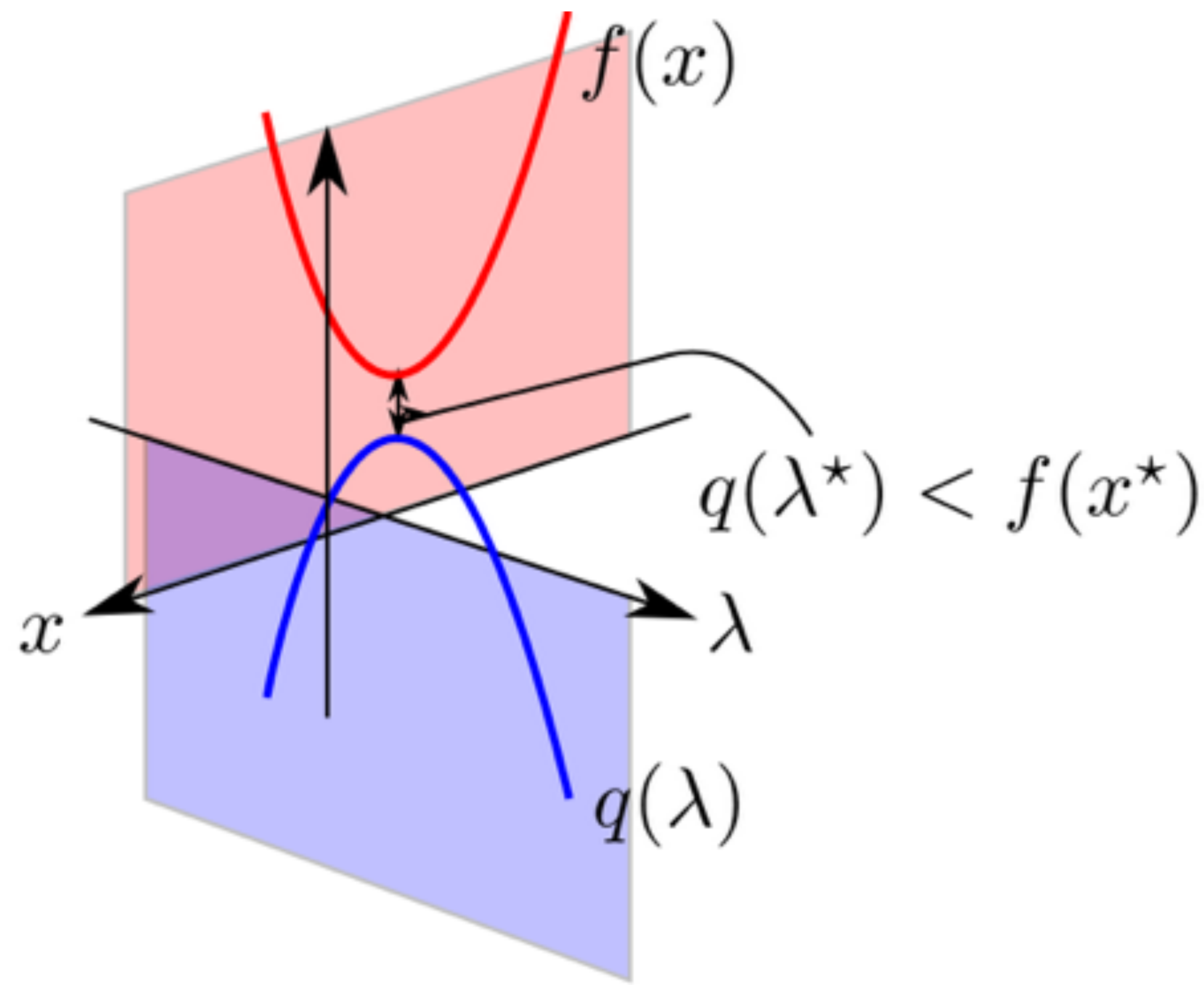
- Convex
- Dual to the Schlesinger's LP relaxation[1]

[1] Shlezinger, (1976) "Syntactic analysis of two-dimensional visual signals in the presence of noise," Cybernetics and Systems Analysis

Basic idea of duality



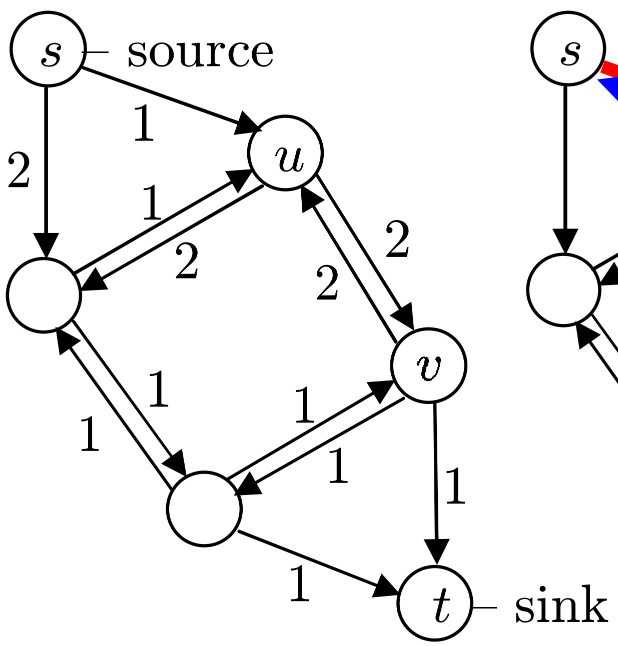
strong duality

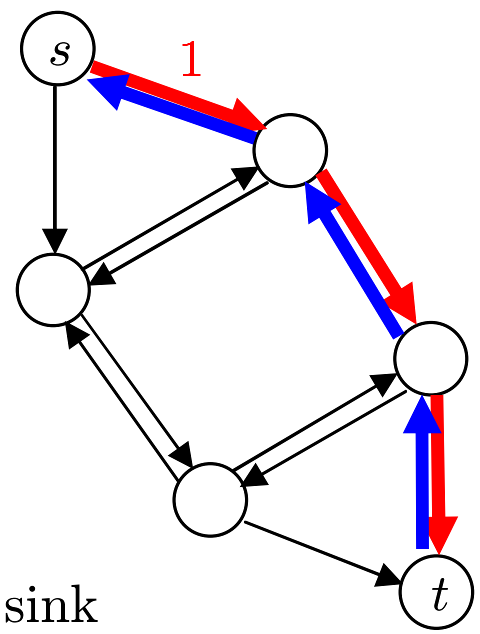


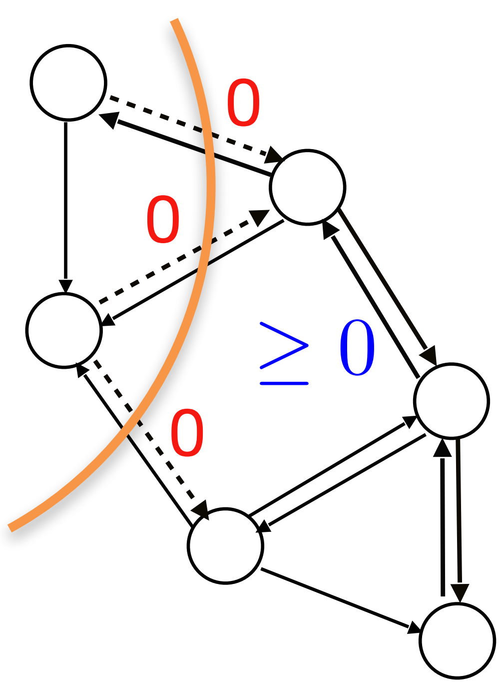
weak duality

Equivalent Transforms

Minimum Cut / Maximum Flow







Linear Programming

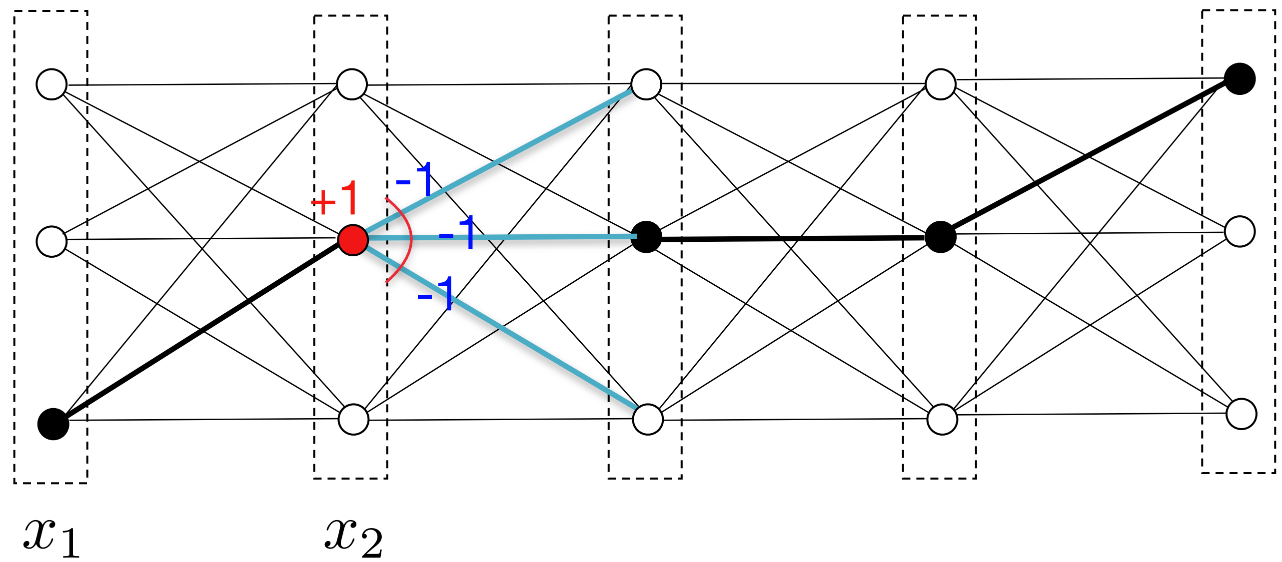
$$\min c^{\mathsf{T}} x - d$$
$$Ax = b$$
$$x \geq 0$$

$$\left[\begin{array}{ccc|c} c_1 & c_2 & c_3 & d \\ \hline a_{11} & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \end{array}\right]$$

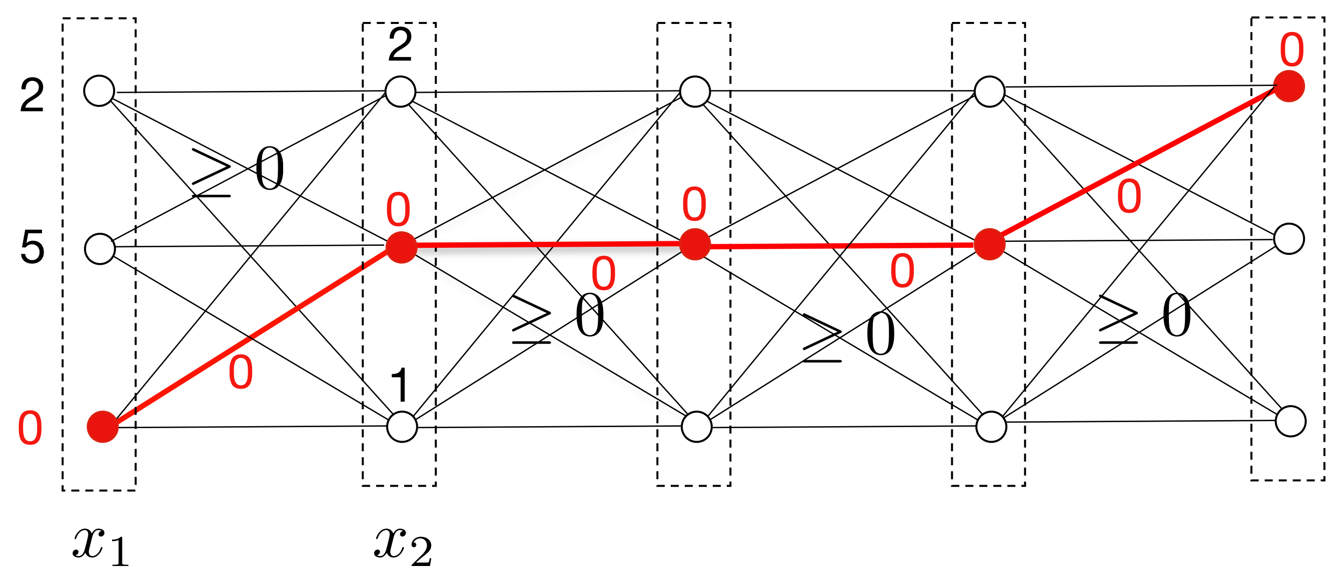
$$\left[\begin{array}{ccc|c} 0 & 0 & \tilde{c}_3 & d \\ \hline 1 & 0 & \tilde{a}_{13} & \tilde{b}_1 \\ 0 & 1 & \tilde{a}_{23} & \tilde{b}_2 \end{array}\right]$$

Equivalent Transforms

Elementary Equivalent transform:

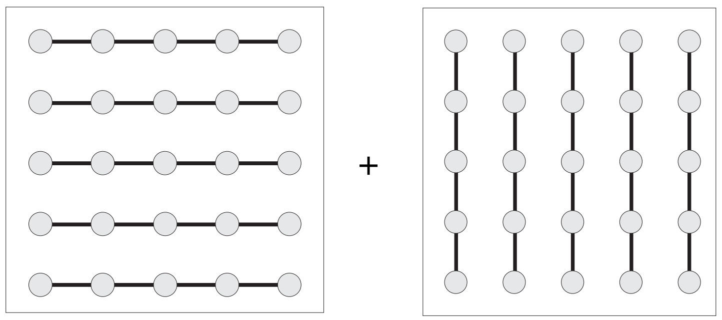


Want to achieve the state:



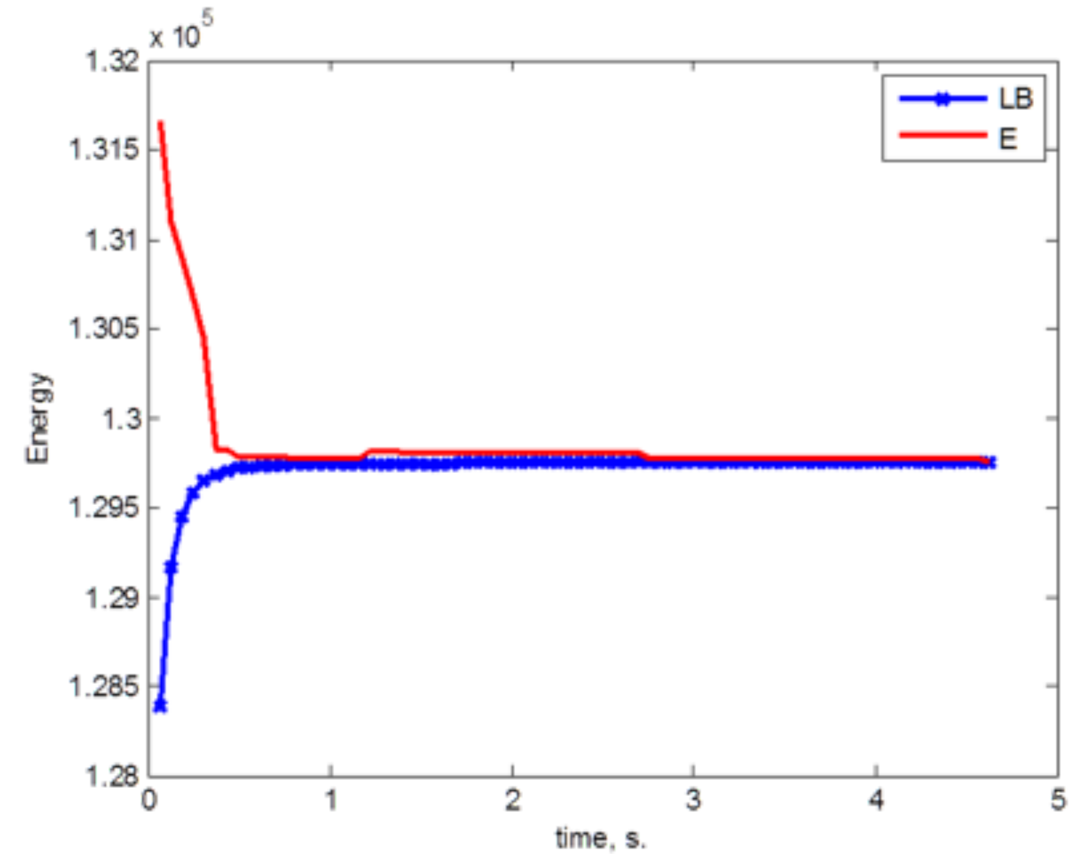
Equivalent Transformation Method, Primal-Dual

Dual Decomposition: Primal Solutions



$$\max_{\varphi}[\min_{x^1}(f^1 + \varphi)(x^1) + \min_{x^2}(f^2 - \varphi)(x^2)]$$

φ - Lagrange multiplier to $x^1 = x^2$

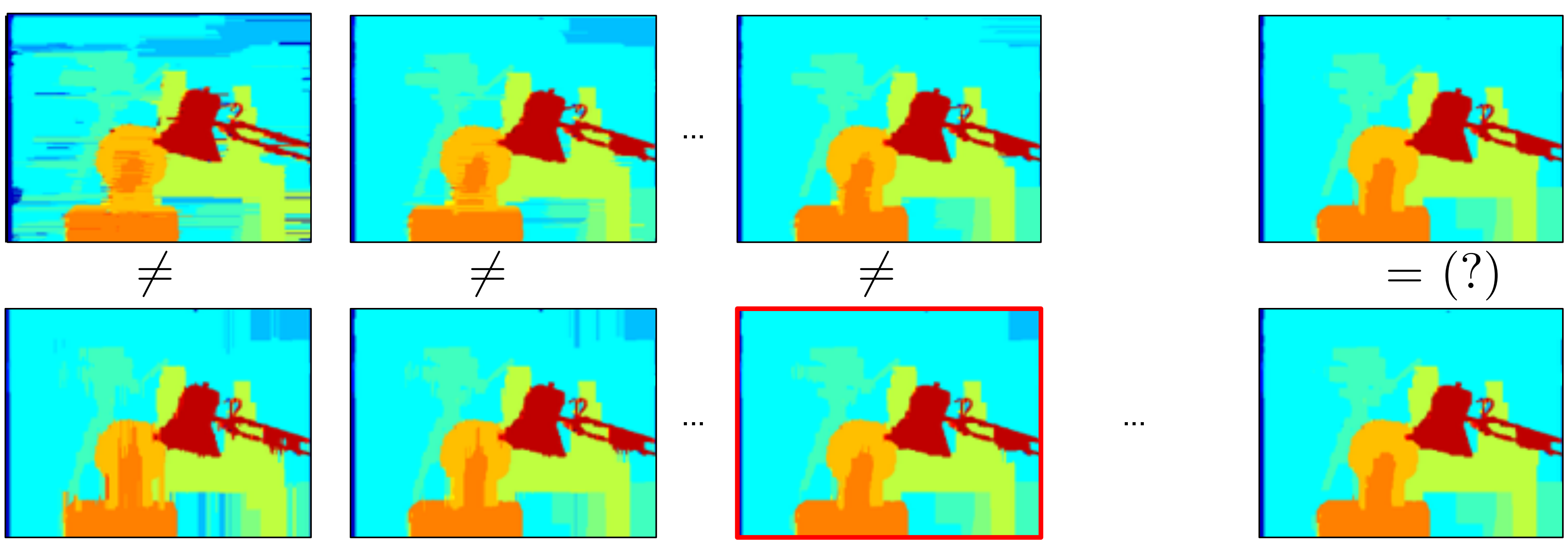


1 2 5 500 Iterations

x^1

$\neq$ $\neq$ $\neq$ $= (?)$

x^2



Stereo Matching - Real-Time Reconstruction

Cost Vol.	Discrete	Cont. Ref.	Total
27 ms	73 ms (4 iterations)	39 ms	**139 ms**

Table: Runtime analysis of the individual components of our stereo matching method (640×480, 128 labels).

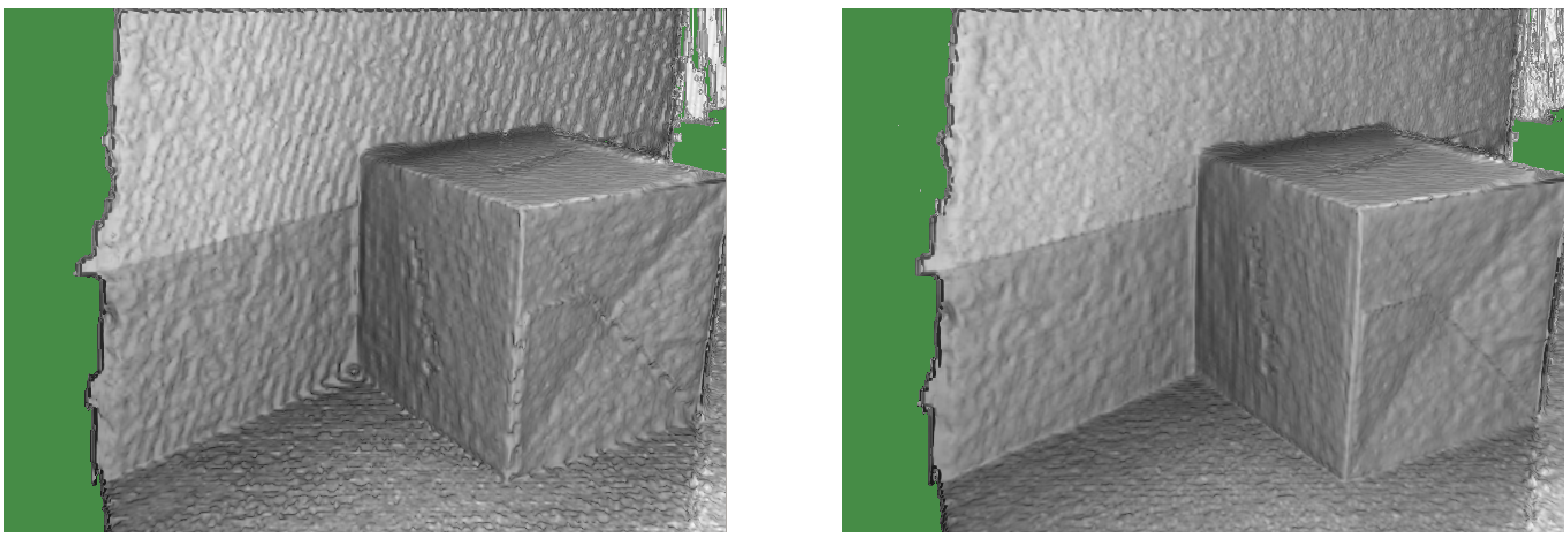
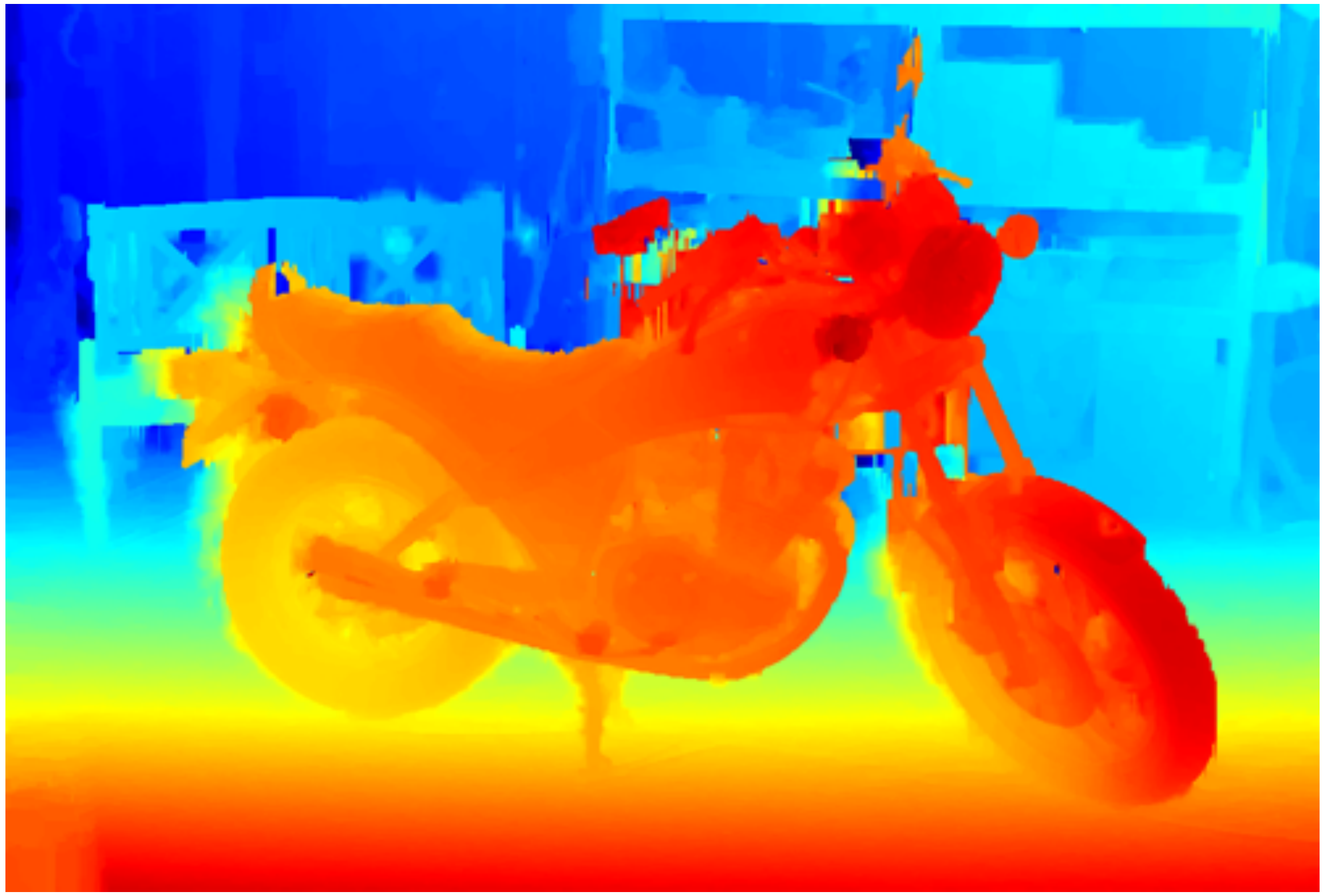
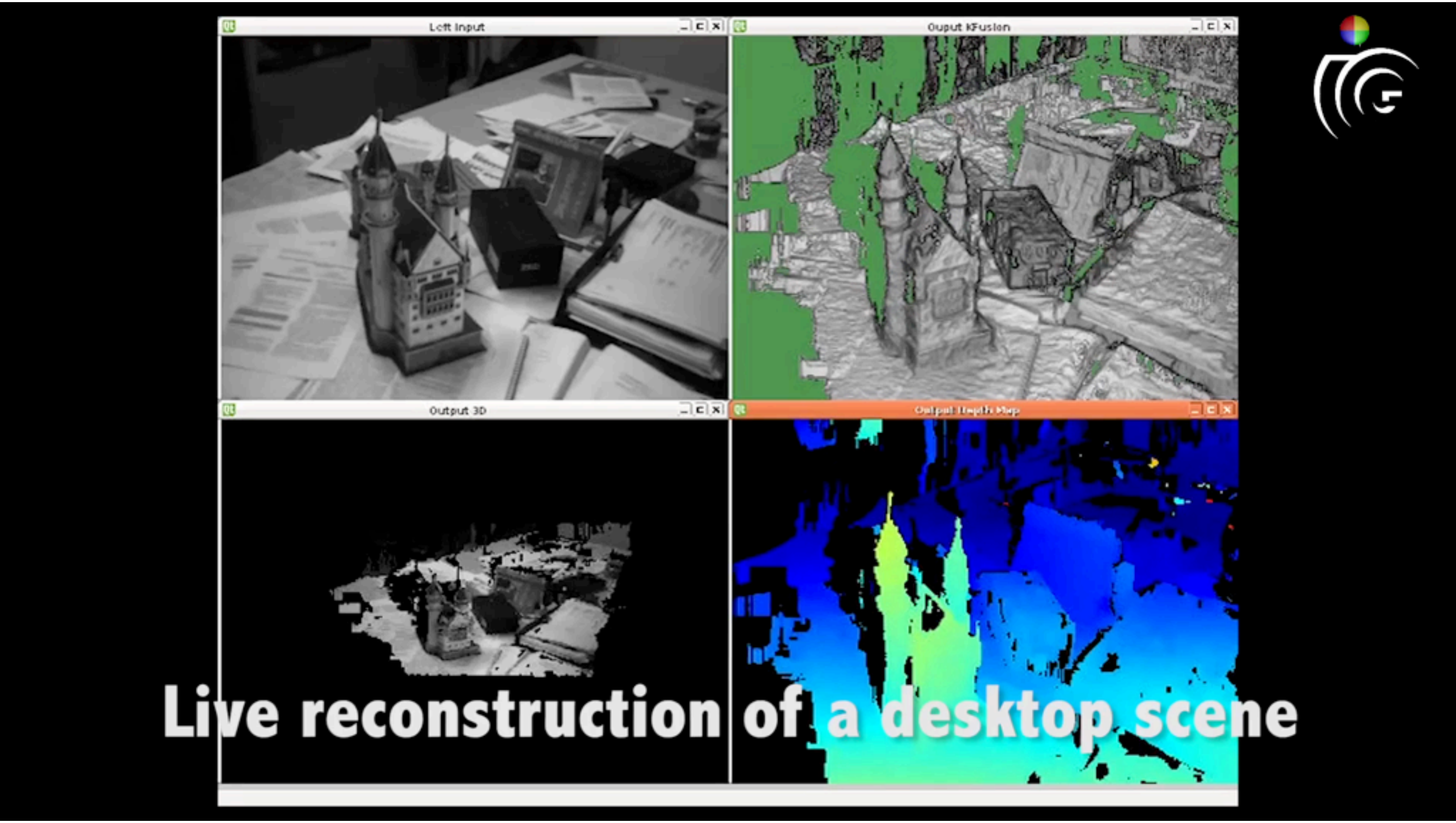


Figure: Influence of continuous refinement on the reconstruction quality of KinectFusion.

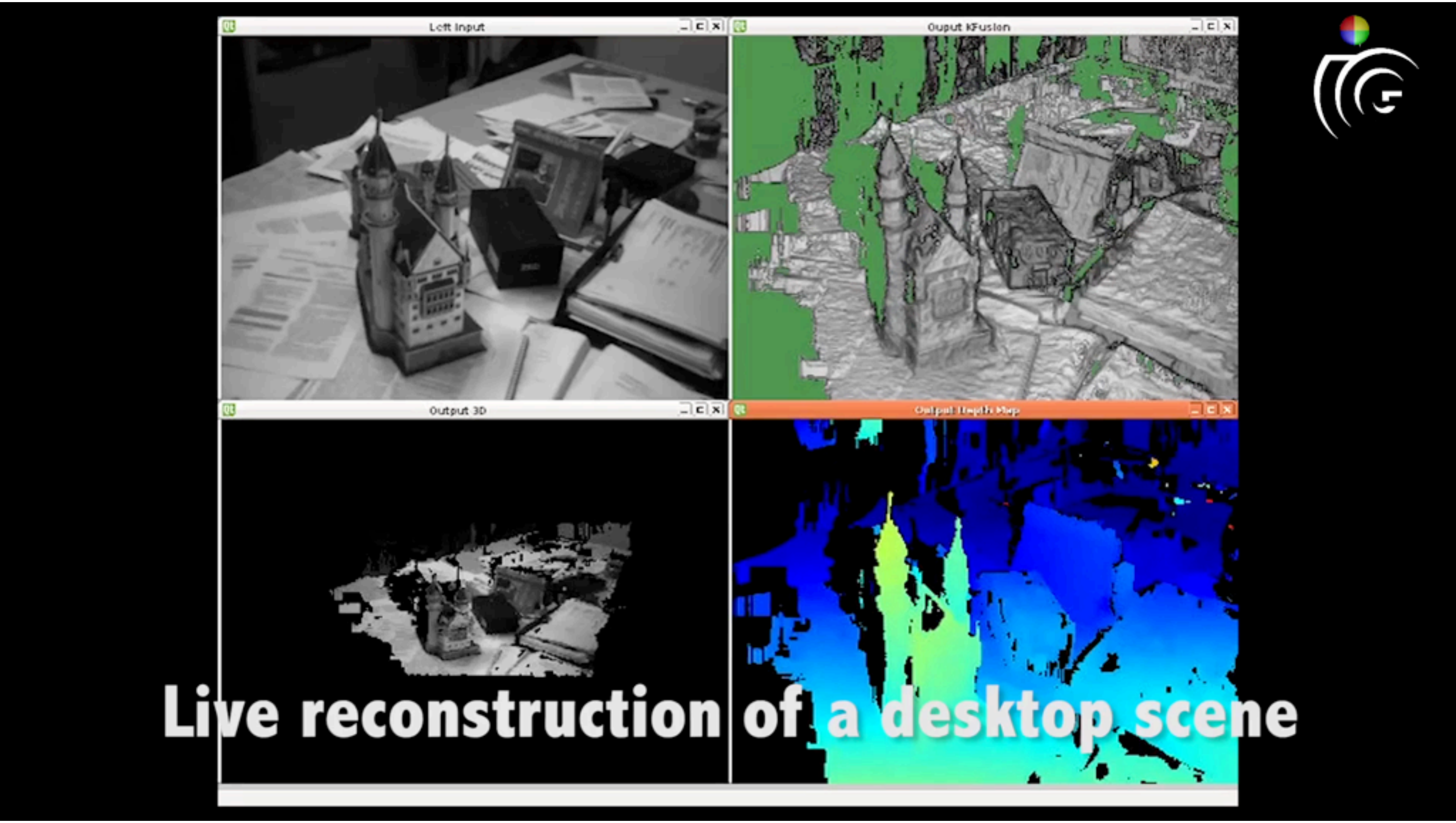
Stereo Matching



Stereo Matching: Real-time fusion



Stereo Matching: Real-time fusion



GrabCut:
Joint Segmentation and Parameter Estimation

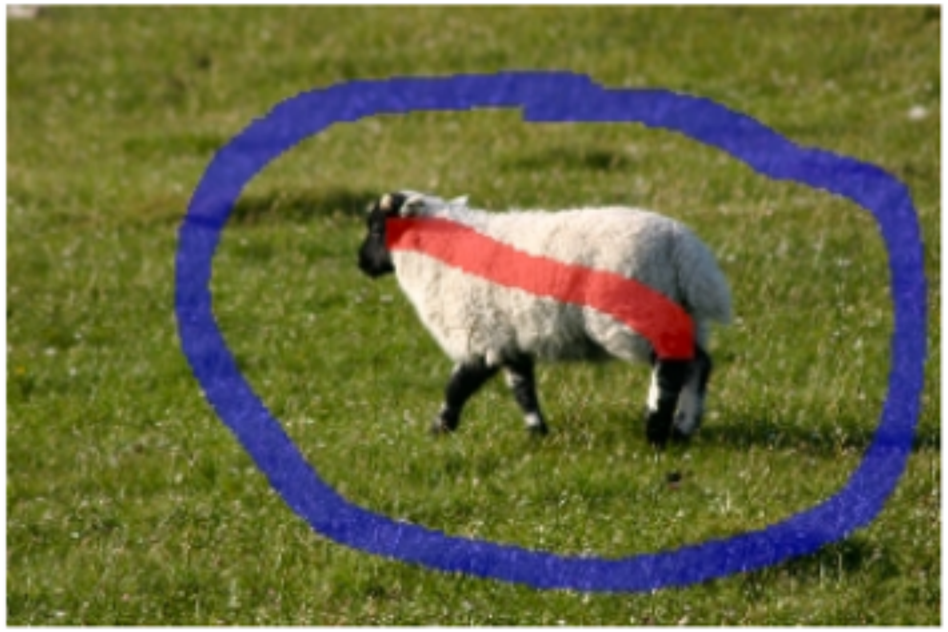
Based on the work by Rother, Kolmogorov, Blake:
"GrabCut" — Interactive Foreground Extraction using Iterated Graph Cuts

Task 1: Joint Segmentation and Parameter Estimation

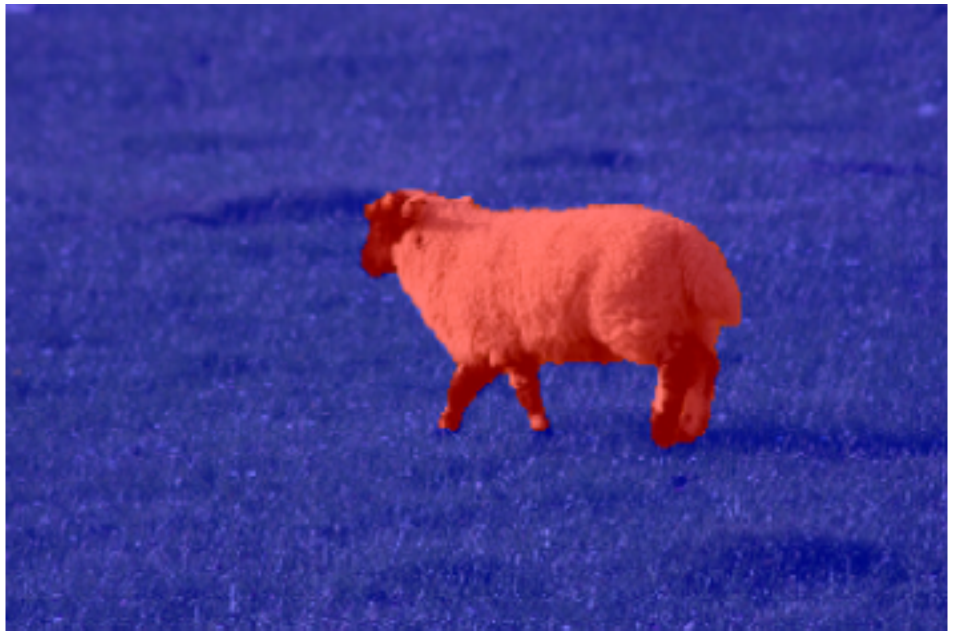
- Input:

Image

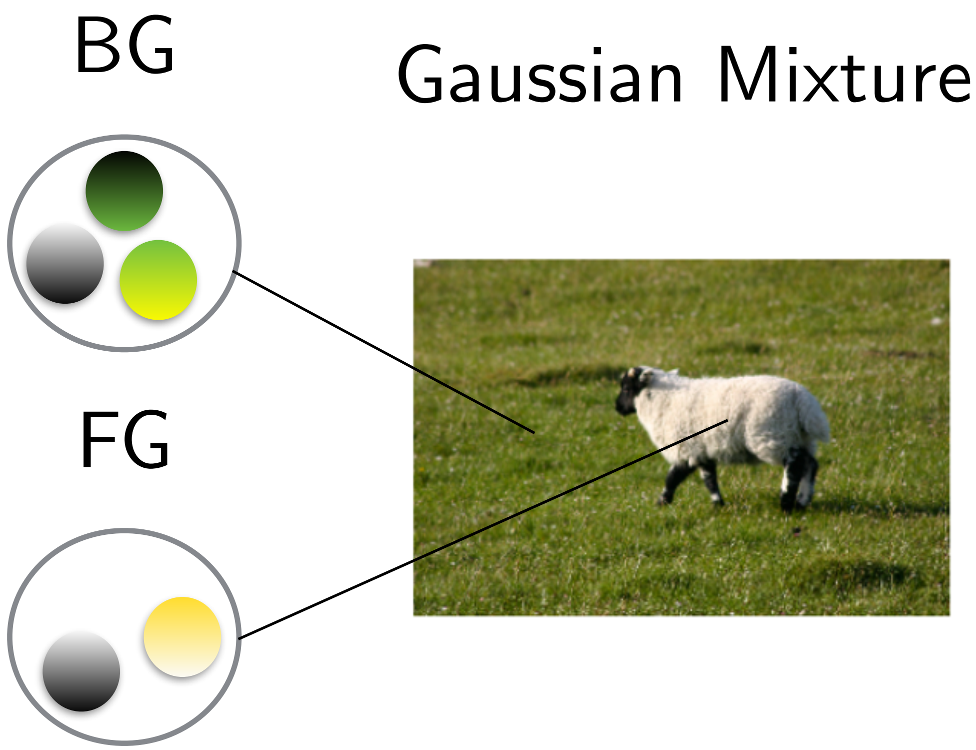
FG / BG brush

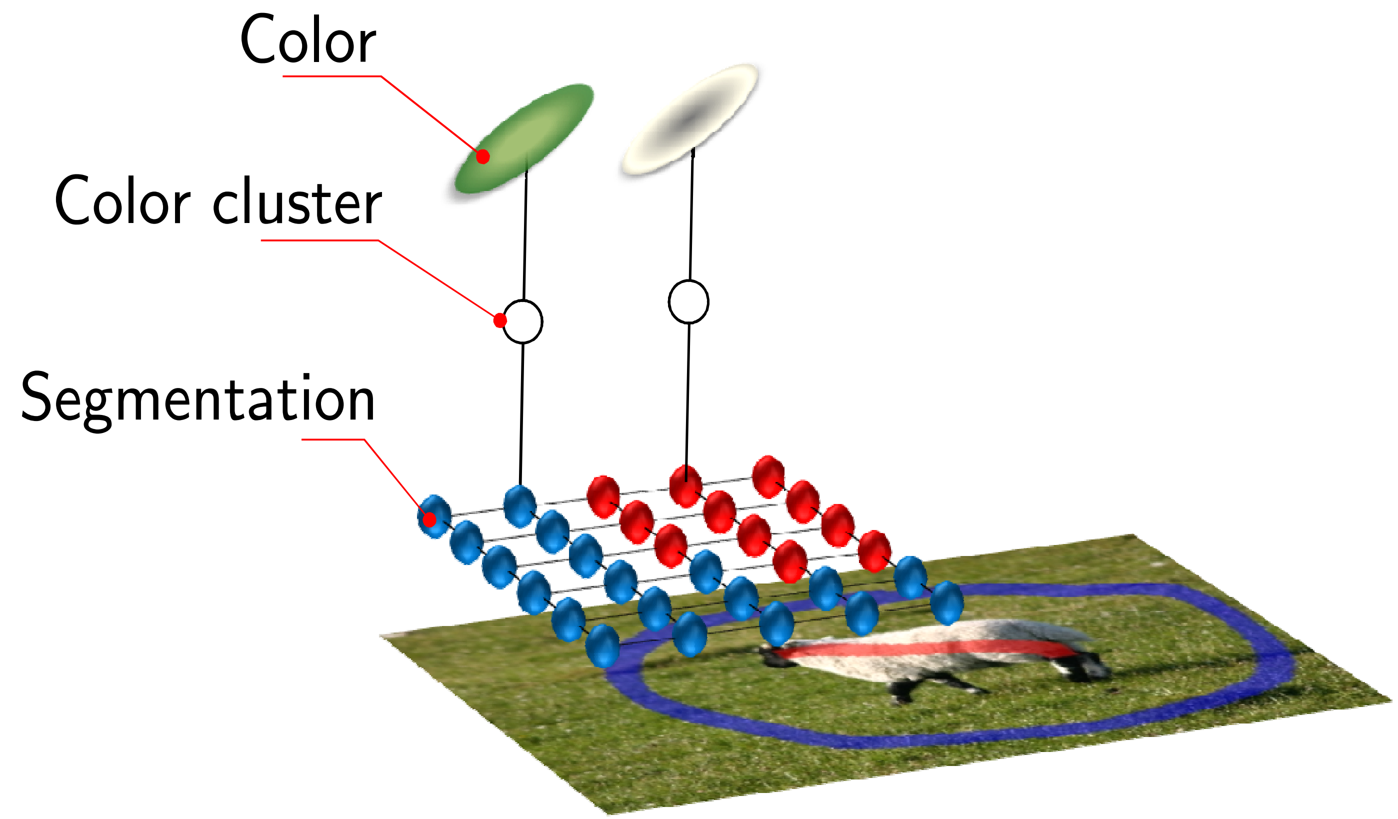


- Output:
  - Complete segmentation



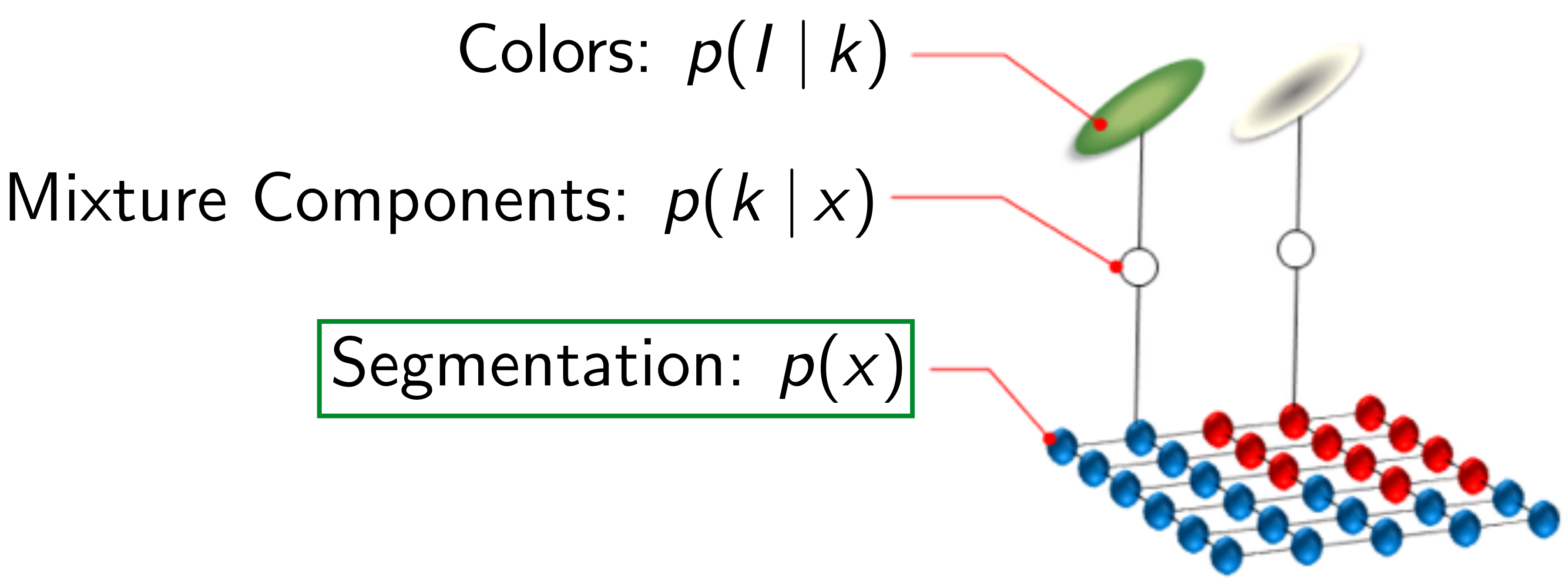
Task 1, model





- Markov random field (generative) model:
- Segmentation $x \colon \Omega \to \{0, 1\}$
  - Model: $p(x)$ - neighboring pixels are more likely to take the same segment
- Color clusters: $k \colon \Omega \to \{1, \dots K\}$
  - Model: $p(k|x)$ - conditionally independent for all pixels
- Image: $I \colon \Omega \to \mathbb{R}^3$ - color drawn from a color cluster
  - Model: $p(I|k)$ - conditionally independent for all pixels

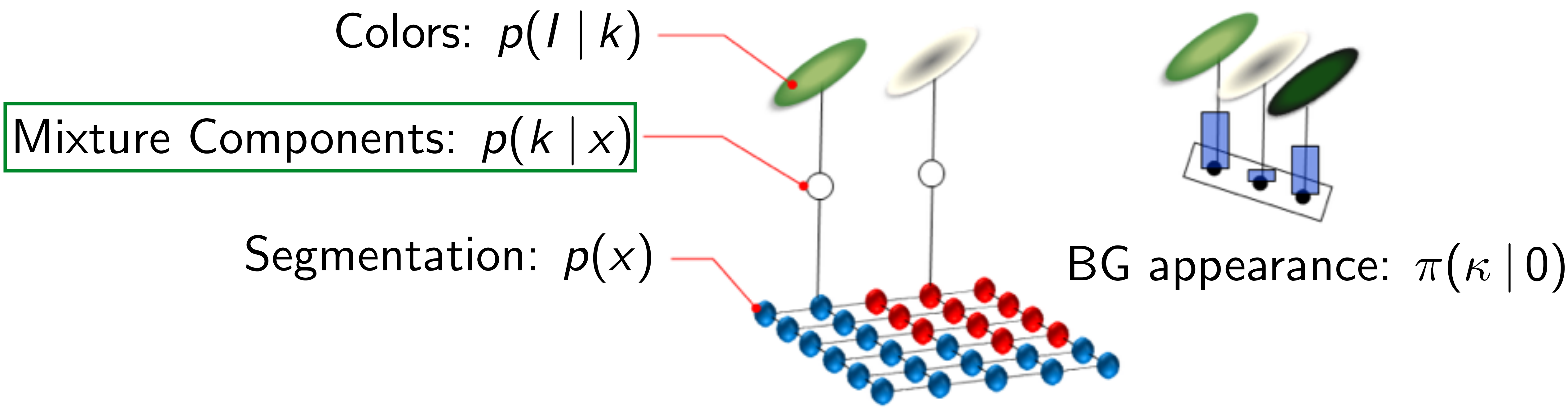
Task 1, model: segmentation



- Segmentation $x\colon \Omega \to \{0,1\}$

$$p(x) = \exp(-J(x)), \quad J(x) = \sum_{ij \in \mathcal{E}} \lambda |x_i - x_j|$$

Task 1, model: mixture components

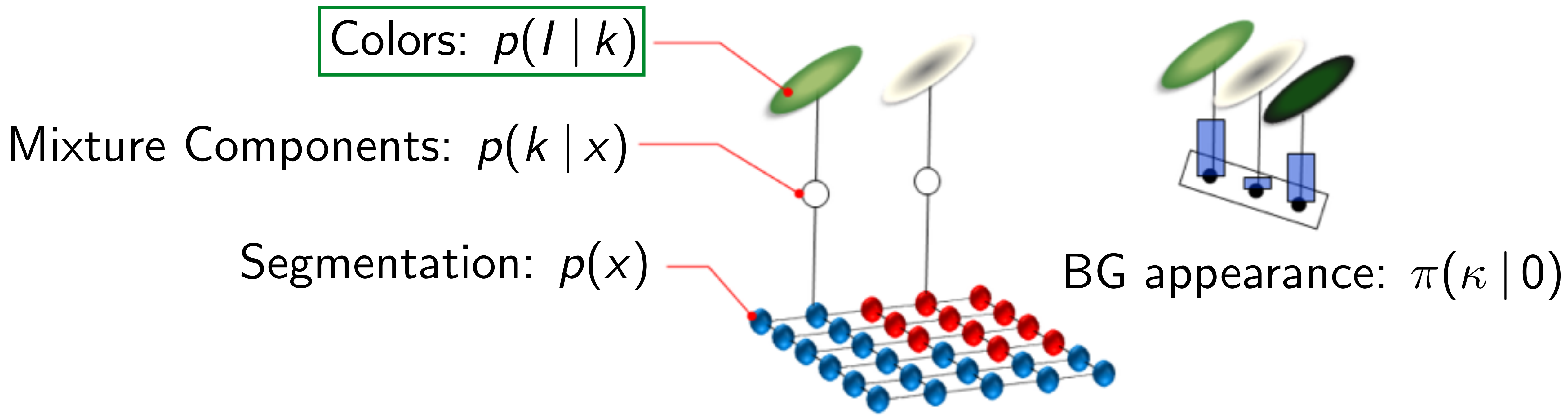


- Color clusters $k\colon \Omega \to \{1,\ldots K\}$
  - Conditional independent model

$$p(k \mid x) = \prod_{i \in \Omega} p(k_i \mid x_i)$$

- $p(k_i{=}\kappa \mid x_i{=}s) = \pi(\kappa \mid s)$ - mixture coefficients

Task 1, model: colors

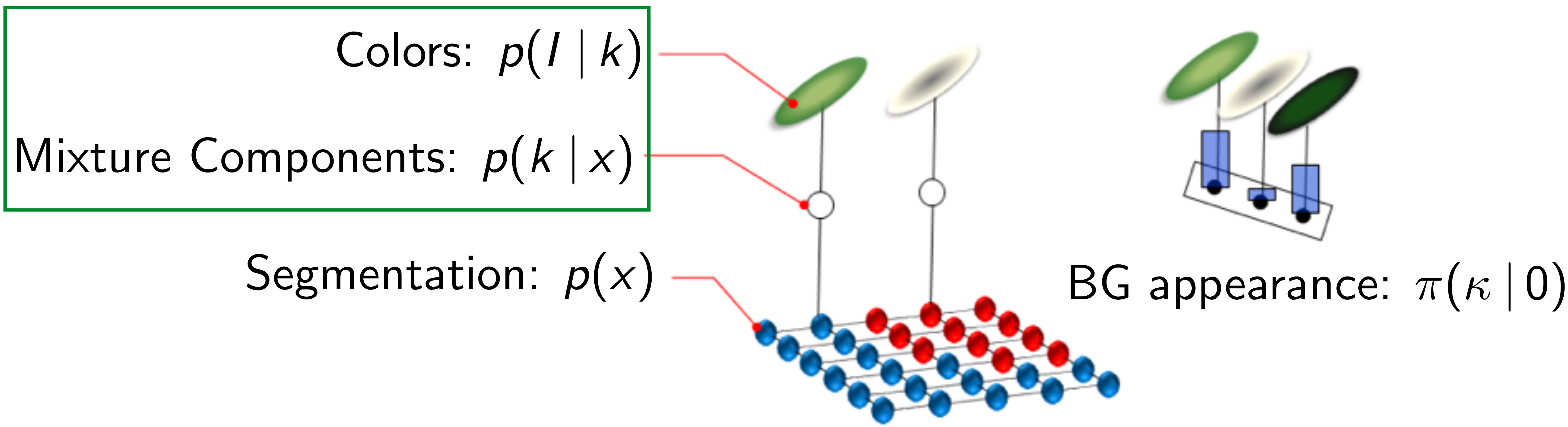


- Colors: $I : \Omega \to \mathbb{R}^3$
  - Conditional independent model

$$p(I \mid x, k) = \prod_{i \in \Omega} p(I_i \mid k_i)$$

- $p(I_i \mid k_i = \kappa) = p_{\mathcal{N}}(I_i; \mu_\kappa, \Sigma_\kappa)$
- Parameters μ_κ, Σ_κ can be learned or pre-estimated for efficiency

Task 1, Gaussian Mixture View

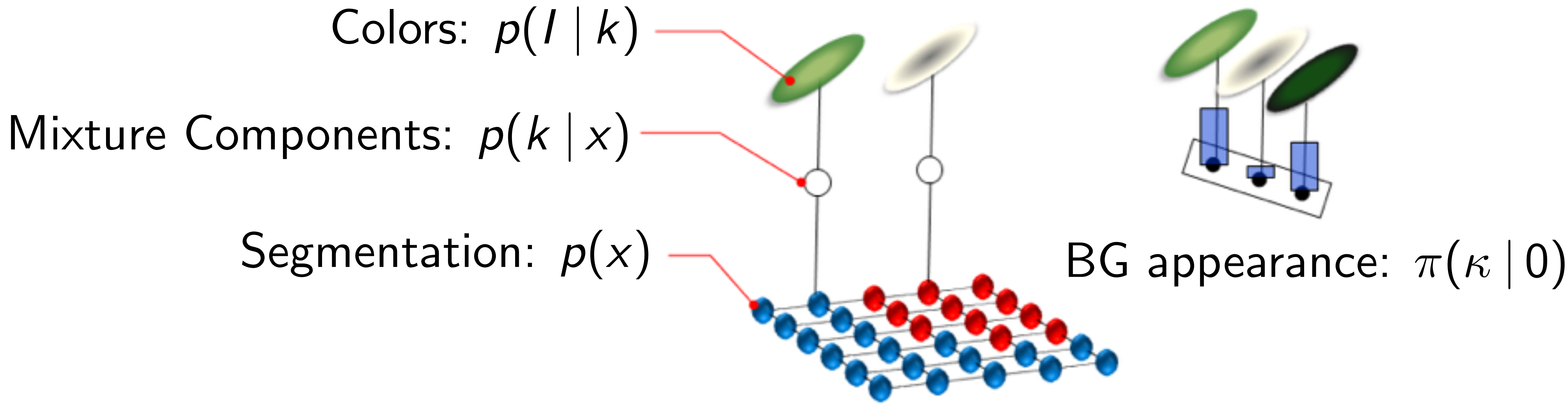


- Efficient Color model:

$$p(I_i \mid x_i) = \sum_{k_i} p(I_i, k_i \mid x_i) = \sum_{k_i} p(I_i|k_i)p(k_i \mid x_i)$$
$$= \sum_{\kappa} p_{\mathcal{N}}(I_i; \mu_\kappa, \Sigma_\kappa)\pi(\kappa \mid x_i)$$

- Gaussian mixture

Task 1, Derivation of EM Algorithm



- Maximum Likelihood:
  - Find segmentation x
  - estimate color models $\pi(\kappa \mid s)$
  - marginalize over hidden color clusters k

Task 1, Derivation of EM Algorithm

- Maximum log Likelihood (for one image):

$$\max_{x,\pi} \log \prod_{i\in\Omega} \sum_{\kappa} p_{\mathcal{N}}(I_i; \mu_\kappa, \Sigma_\kappa)\pi(\kappa|x_i)p(x)$$

  - Of the form $\max \prod \sum$, apply EM lower bound:

$$\geq \sum_{i\in\Omega} \sum_{\kappa} \alpha(\kappa \mid x_i)\Big(\log\big(p_{\mathcal{N}}(\mu_\kappa, \Sigma_\kappa) + \log \pi(\kappa|x_i)\big) - \log \alpha(\kappa \mid x_i)\Big) + \log p(x)$$

- E step:

$$\alpha(\kappa \mid x_i) \propto p_{\mathcal{N}}(I_i; \mu_\kappa, \Sigma_\kappa)\pi(\kappa \mid x_i) \tag{1}$$

- M step:

$$x \in \operatorname*{argmin}_x J(x) - \sum_i \sum_\kappa \alpha(\kappa \mid x_i) \log\big(p(I_i \mid \kappa)\big)\pi(\kappa \mid x_i) \tag{2}$$

$$\pi(\kappa \mid s) \propto \sum_{i \,\mid\, x_i = s} \alpha(\kappa \mid s) \tag{3}$$

Task 1, Overall Algorithm

- EM: Iteratively reestimate
  - Segmentation $x\colon \Omega \to \{0, 1\}$ having appearance model π and probabilities of hidden components α
  - Appearance models, $\pi(\kappa \mid s)$
  - Soft cluster assignment for each pixel, $\alpha(\kappa \mid x_i)$

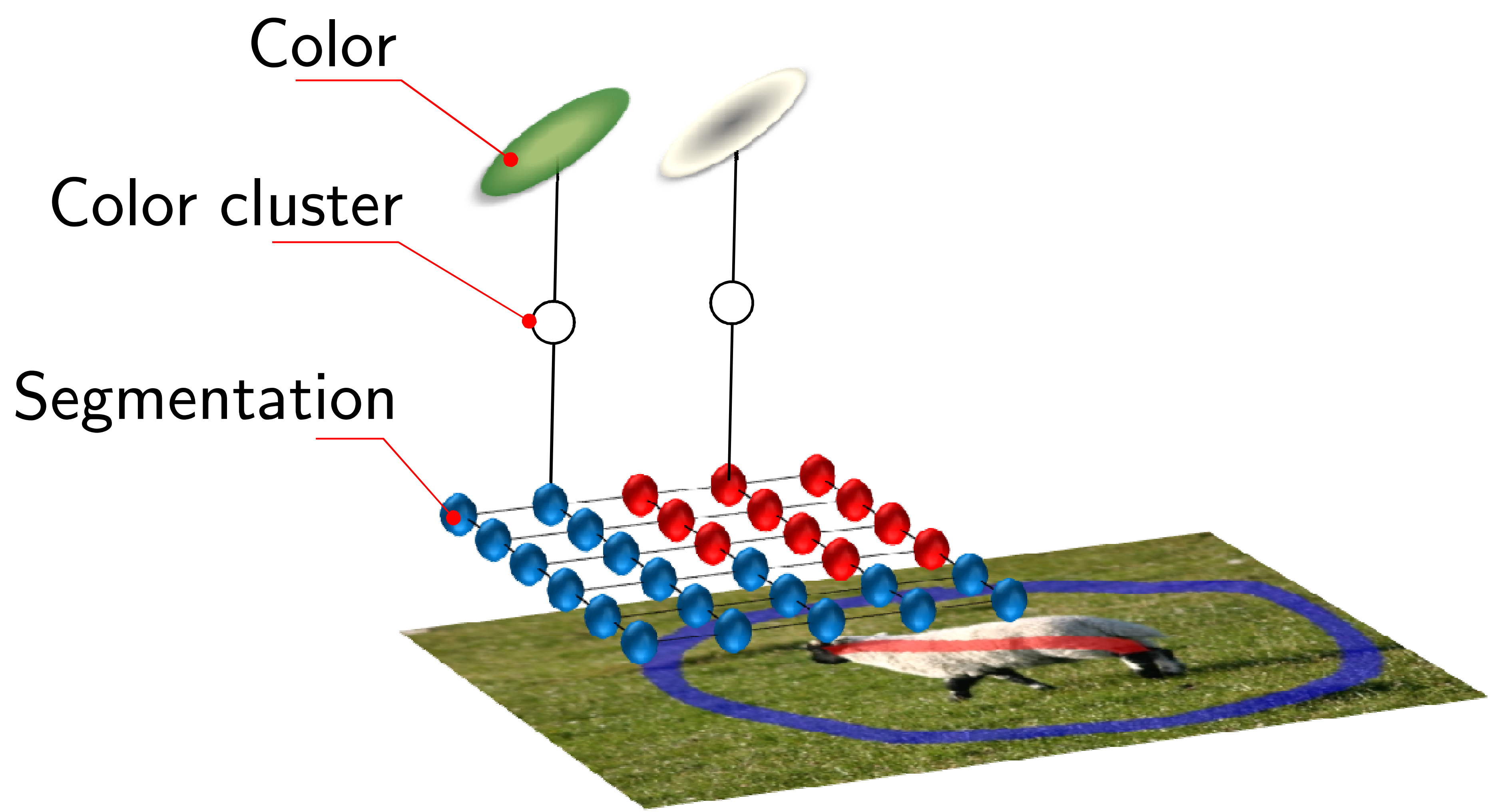
Notice the difference to the following "ad-hoc" algorithm:

- Ad-hock: Iteratively reestimate
  - Segmentation x for current appearance model π
  - Appearance models π from current segmentation

The later method may be not converging and can get stuck more easily, similarly to K-means.

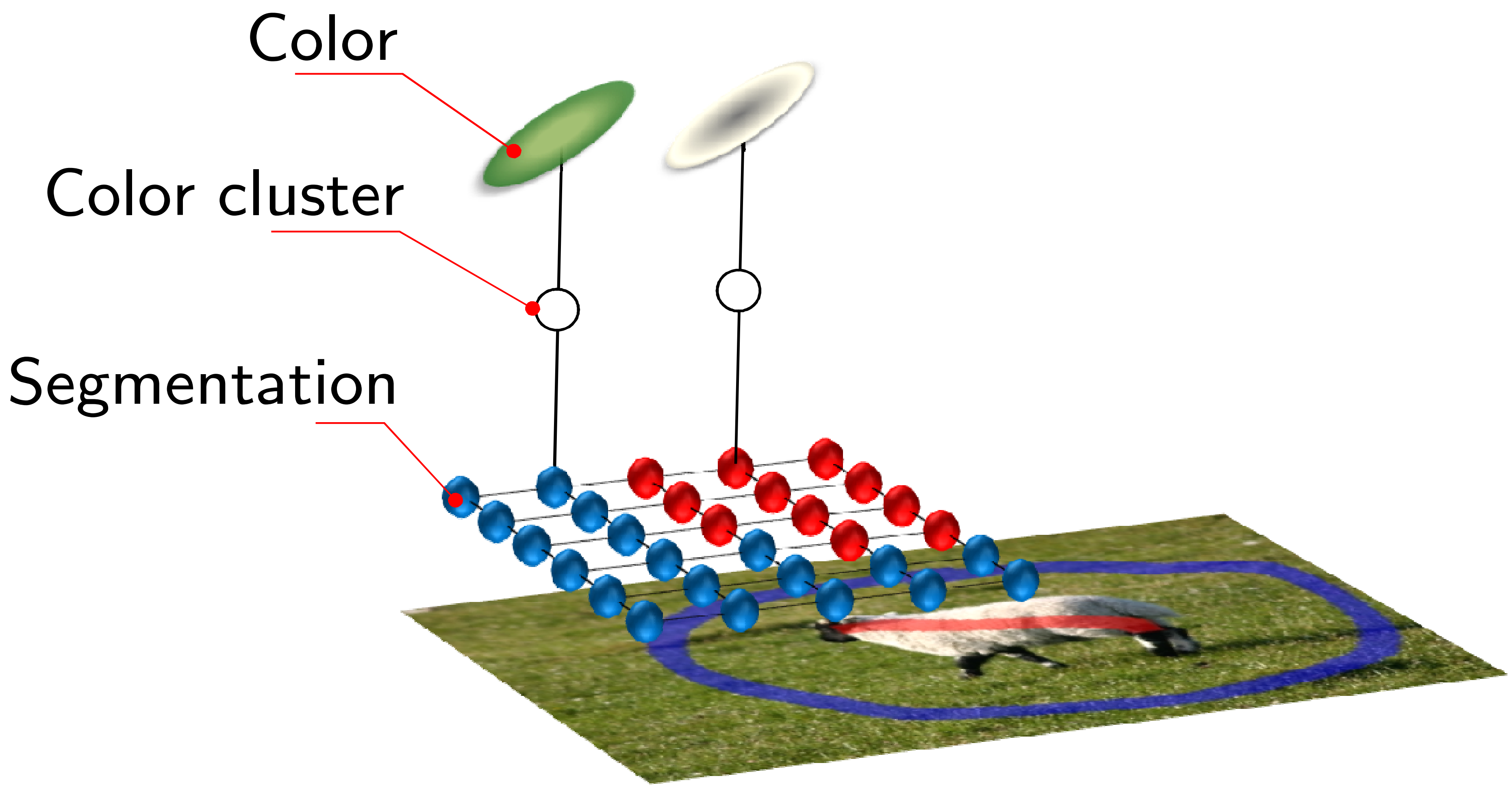
GrabCut TV version

Grabcut TV version



Generative model (bottom-up in the picture)

- Segmentation $u \colon \Omega \to \{0, 1\}$
- Assignment of pixels to color clusters $k \colon \Omega \to \{1, \ldots, K\}$
- Image $I \colon \Omega \to \mathbb{R}^3$ – color drawn from Gaussian cluster k

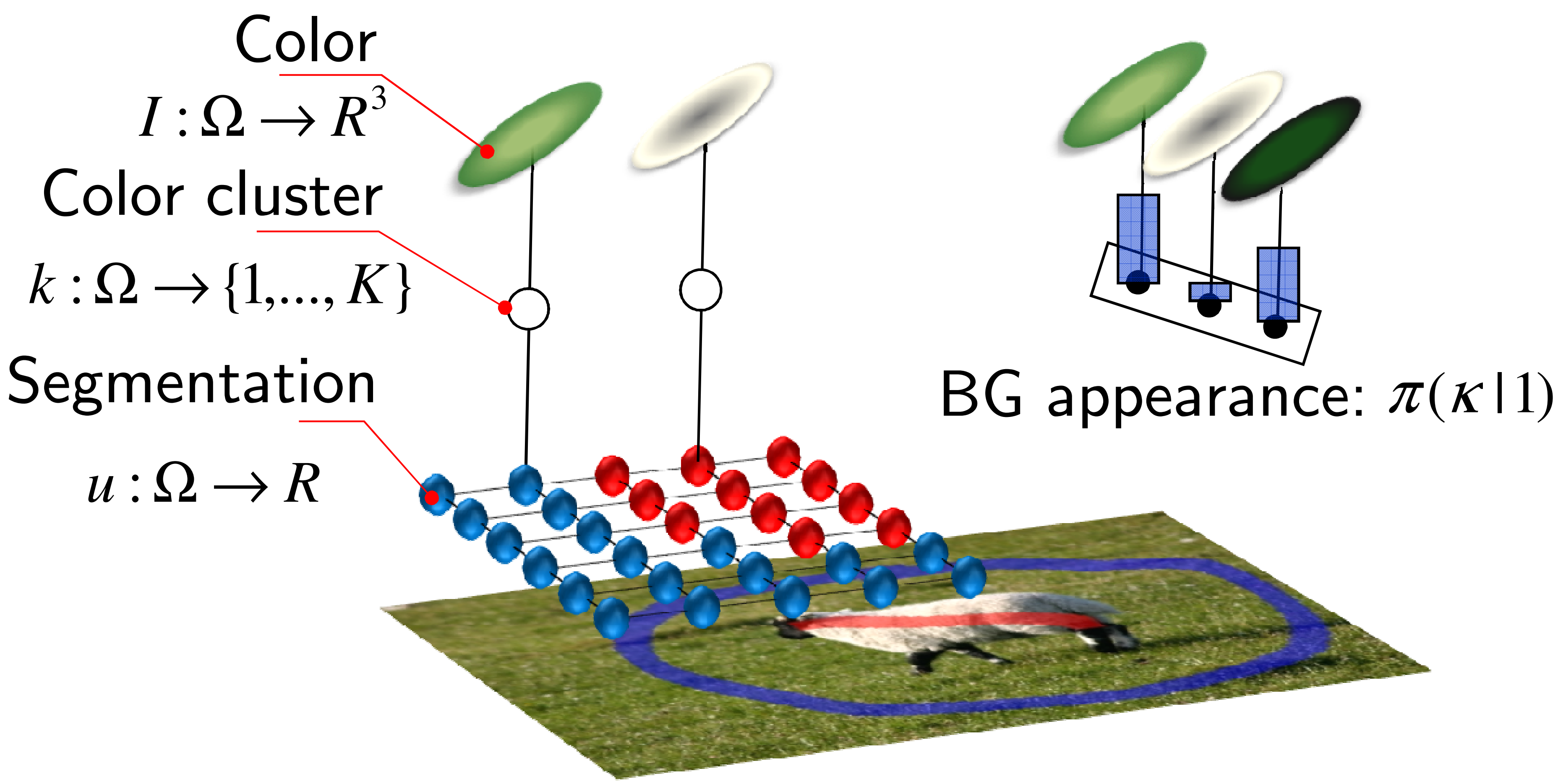


Segmentation:

- Assume TV prior (neighboring pixels are more likely to be in the same segment)

$$p(u) = exp(-J(u)), \quad J(u) = \lambda \sum_{\Omega} ||(\nabla u)(x)||_2.$$

Graphical Model



Color cluster:

- Assume conditional independence

$$p(k \mid u) = \prod_{x\in\Omega} p(k(x) \mid u(x));$$

$p(k(x){=}\kappa, u(x){=}s) \;=\; \pi(\kappa \mid s)$ – unknown appearance

Graphical Model

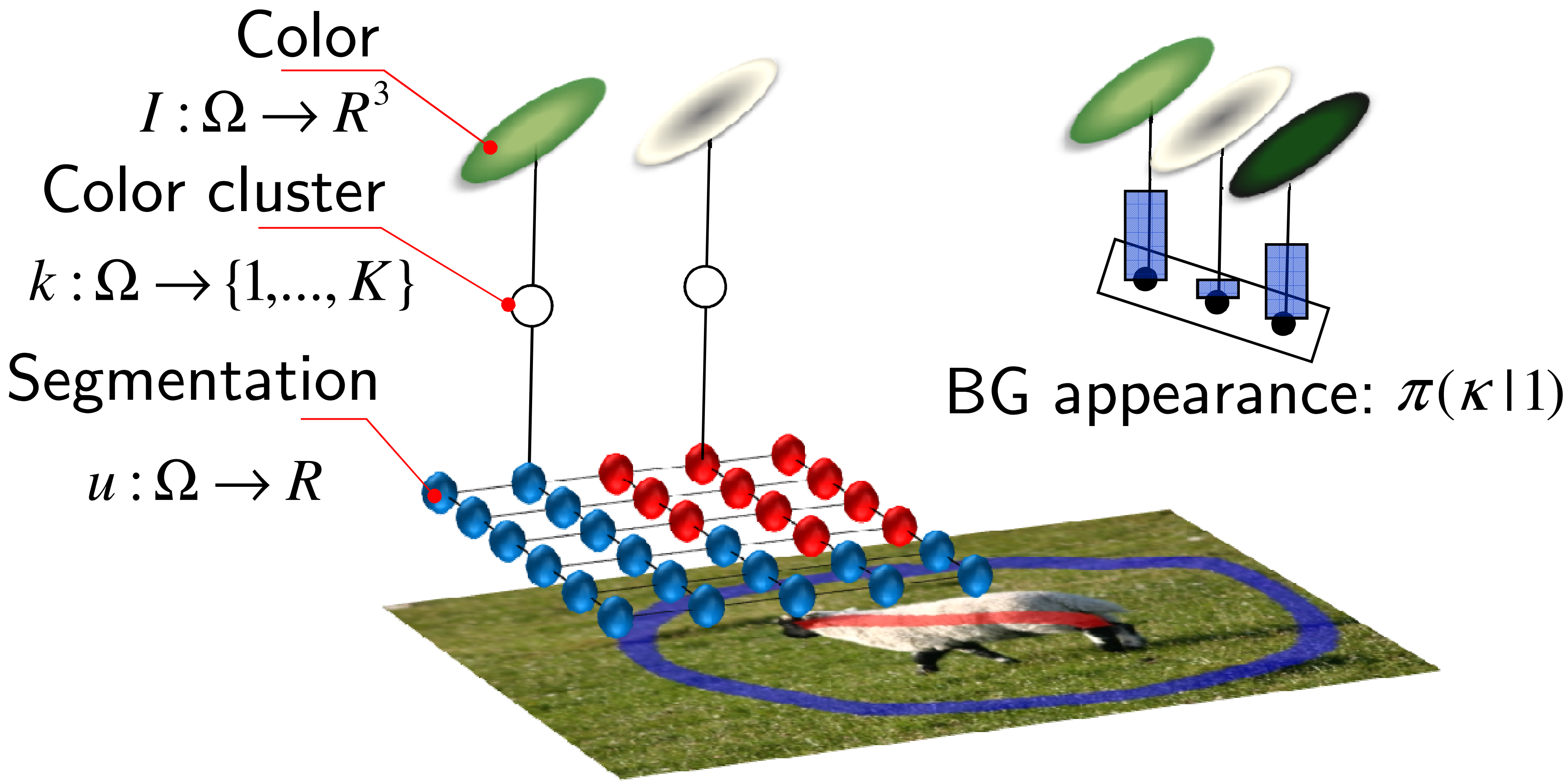


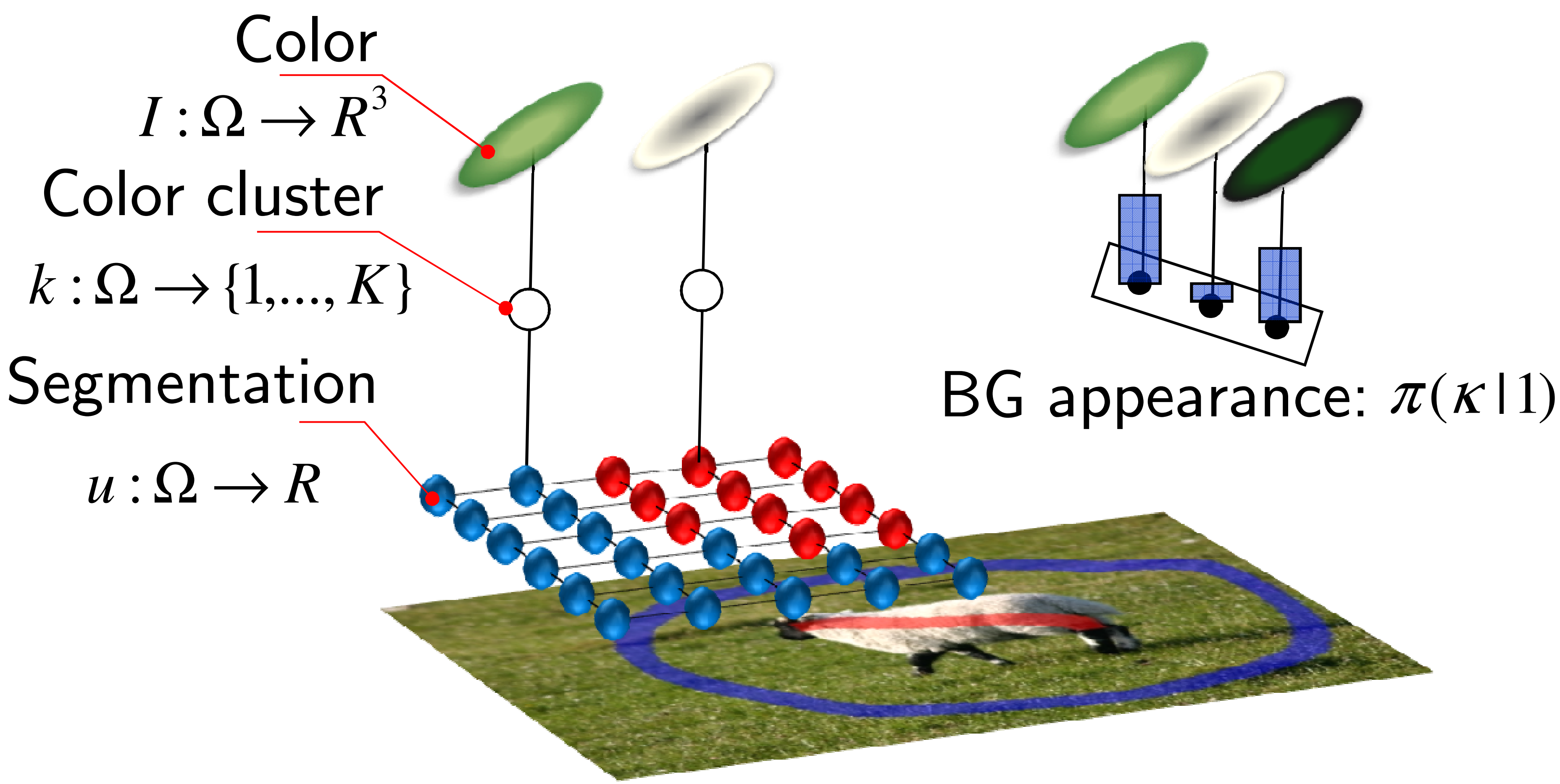
Image colors:

- assume conditional i.i.d. given cluster k,

$$p(I(x) \mid k(x)=\kappa) = G_{\Sigma_\kappa}(I(x) - \mu_\kappa);$$

parameters Σ_κ, μ_k could be learned or preestimated for efficiency

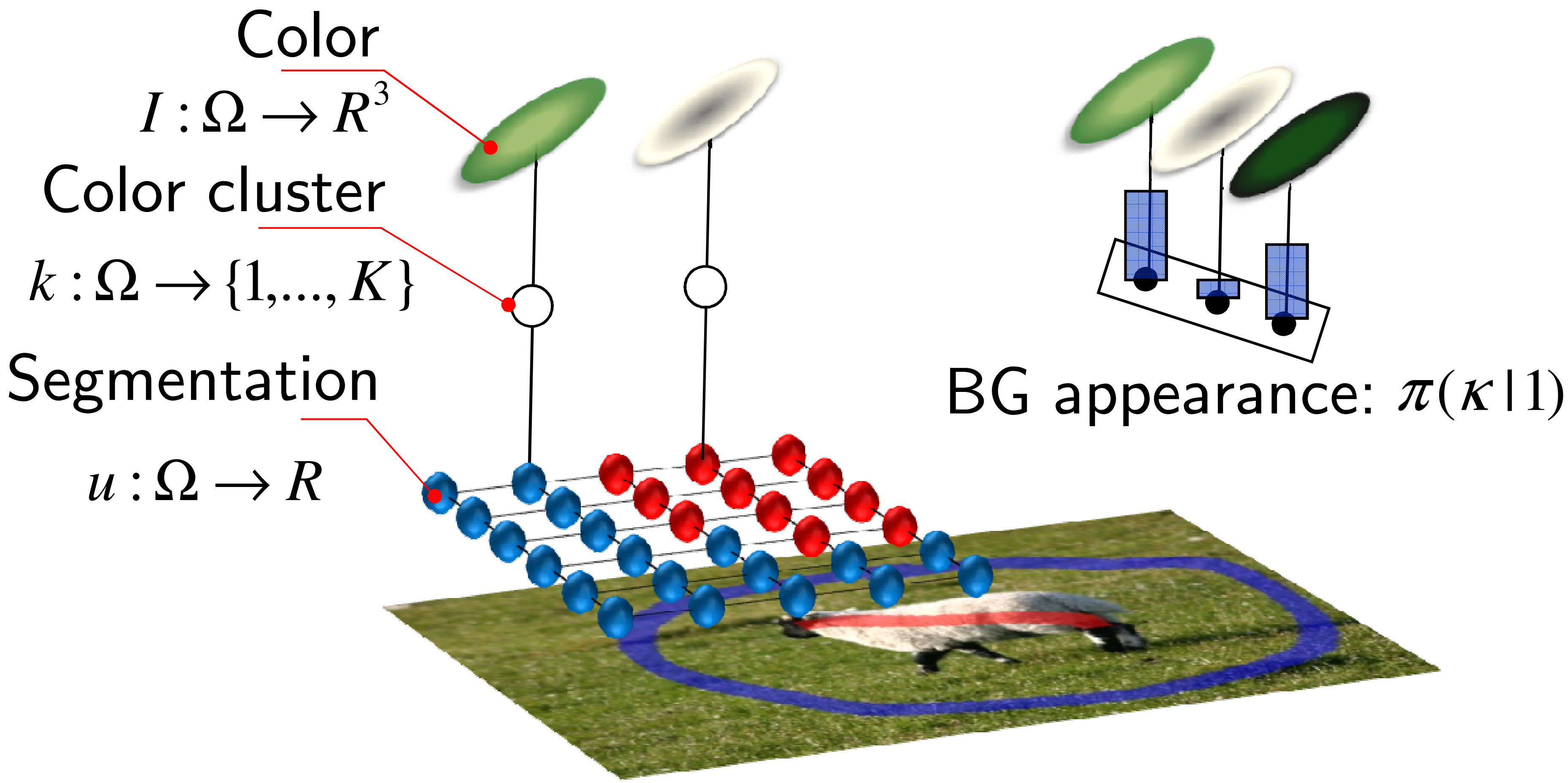
Graphical Model



- Image colors are drawn from a mixture:

$$p(I(x) \mid u(x) = s) = \sum_{\kappa} \pi(\kappa \mid s) G_{\Sigma_\kappa}(I(x) - \mu_\kappa);$$

Graphical Model



Maximum Likelihood:

- find segmentation u
- estimate color models $\pi(\kappa \mid s)$
- marginalize over latent color clusters k

Apply EM Algorithm

- Maximum likelihood:

$$\operatorname*{argmax}_{u,\pi} \sum_{k} \prod_{x \in \Omega} p(I(x) \mid k(x)) \pi(k(x) \mid u(x)) p(u)$$

- derive (blackboard)

$$= \operatorname*{argmax}_{u,\pi} \prod_{s \in \{0,1\}} \prod_{x \in \Omega \mid u(x)=s} \sum_{\kappa} p(I(x) \mid \kappa) \pi(\kappa \mid s) p(u),$$

- to allow for linearization, express log likelihood as as

$$\sum_{s \in \{0,1\}} \sum_{x \in \Omega} (1 + s - u(x)) \log \sum_{\kappa=1}^{K} p(I(x) \mid \kappa) \pi(\kappa \mid s) + \log p(u).$$

Apply EM Algorithm

- EM lower bound:

$$\sum_{s\in\{0,1\}} \sum_{x\in\Omega} \log \sum_{\kappa=1}^{K} \Big(p(I(x) \mid \kappa)\pi(\kappa \mid s)\Big)^{(1+s-u(x))} + \log p(u)$$

introduce numbers $\alpha_x(\kappa \mid s) \geq 0$ such that $\sum_\kappa \alpha_x(\kappa \mid s) = 1$,

$$\geq \sum_{s\in\{0,1\}} \sum_{x} \sum_{\kappa} \Big(\alpha_x(\kappa \mid s)(1+s-u(x)) \log[p(I(x) \mid \kappa)\pi(\kappa \mid s)]$$

$$- \log \alpha_x(k(x) \mid s)\Big) + \log p(u).$$

- Bound valid for $u\colon \Omega \to [0,1]$!

Maximization Step

- Maximization step in u (blackboard):

$$u := \operatorname*{argmax}_u \sum_x g(x)u(x) + J(u),$$

$$g(x) = \sum_{\kappa=1}^{K} \Big(\alpha_x(\kappa \mid 1) - \alpha_x(\kappa \mid 0)\Big) \log[p(I(x) \mid \kappa)\pi(\kappa \mid s)].$$

(log likelihood ratio of FG and BG models with soft assignment α)

- Maximization step in π (blackboard):

$$\pi(\kappa \mid s = 0) \propto \sum_x \alpha_x(\kappa \mid s = 0)(1 - u(x)),$$

$$\pi(\kappa \mid s = 1) \propto \sum_x \alpha_x(\kappa \mid s = 1)u(x).$$

Expectation Step

(Maximize (tighten) bound in α)

- (blackboard):

$$\alpha_x(\kappa \mid 1) \propto u(x) p(I(x) \mid \kappa) \pi(\kappa \mid 1)$$
$$\alpha_x(\kappa \mid 0) \propto (1 - u(x)) p(I(x) \mid \kappa) \pi(\kappa \mid 0),$$

Overall Algorithm

Iteratively reestimate

- Soft (hard) segmentation, $u\colon \Omega \to [0, 1]$
  (resp. $u\colon \Omega \to \{0, 1\}$)
- Appearance models, $\pi(\kappa \mid s)$
- Soft cluster assignment of each pixel, $\alpha_x(\kappa \mid s) \in [0,\ 1]$