Advanced Machine Learning
Summer 2019

Part 11 – Graphical Models V
15.05.2019

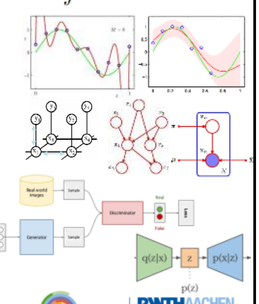
Prof. Dr. Bastian Leibe

RWTH Aachen University, Computer Vision Group
http://www.vision.rwth-aachen.de

Course Outline

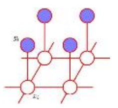
- Regression Techniques
  - Linear Regression
  - Regularization (Ridge, Lasso)
  - Kernels (Kernel Ridge Regression)
- Deep Reinforcement Learning
- Probabilistic Graphical Models
  - Bayesian Networks
  - Markov Random Fields
  - Inference (exact & approximate)
- Deep Generative Models
  - Generative Adversarial Networks
  - Variational Autoencoders

$f : \mathcal{X} \to \mathbb{R}$



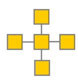
Recap: MRF Structure for Images

- Basic structure

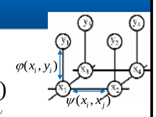


- Two components
  - Observation model
    - How likely is it that node x_i has label L_i given observation y_i?
    - This relationship is usually learned from training data.
  - Neighborhood relations
    - Simplest case: 4-neighborhood
    - Serve as smoothing terms.

    $\Rightarrow$ Discourage neighboring pixels to have different labels.
    - This can either be learned or be set to fixed "penalties".



Recap: Energy Formulation

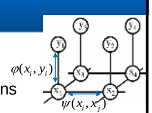


- Energy function

$$E(x,y) = \underbrace{\sum_i \varphi(x_i, y_i)}_{\text{Single-node potentials}} + \underbrace{\sum_{i,j} \psi(x_i, x_j)}_{\text{Pairwise potentials}}$$

- Single-node (unary) potentials φ
  - Encode local information about the given pixel/patch.
  - How likely is a pixel/patch to belong to a certain class (e.g. foreground/background)?
- Pairwise potentials ψ
  - Encode neighborhood information.
  - How different is a pixel/patch's label from that of its neighbor? (e.g. based on intensity/color/texture difference, edges)

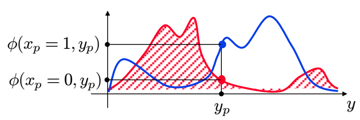
Recap: How to Set the Potentials?



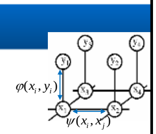
- Unary potentials
  - E.g. color model, modeled with a Mixture of Gaussians

$$\phi(x_i, y_i; \theta_\phi) = \log \sum_k \theta_\phi(x_i, k) p(k|x_i) \mathcal{N}(y_i; \bar{y}_k, \Sigma_k)$$

$\Rightarrow$ Learn color distributions for each label



Recap: How to Set the Potentials?



- Pairwise potentials
  - Potts Model

$$\psi(x_i, x_j; \theta_\psi) = \theta_\psi \delta(x_i \neq x_j)$$

    - Simplest discontinuity preserving model.
    - Discontinuities between any pair of labels are penalized equally.
    - Useful when labels are unordered or number of labels is small.
  - Extension: "contrast sensitive Potts model"

$$\psi(x_i, x_j, g_{ij}(y); \theta_\psi) = \theta_\psi g_{ij}(y) \delta(x_i \neq x_j)$$

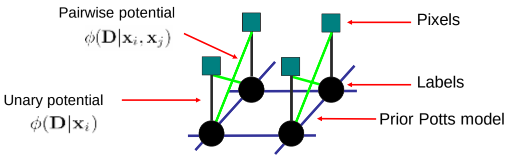
where

$$g_{ij}(y) = e^{-\beta \|y_i - y_j\|^2} \qquad \beta = 2 \cdot avg\left(\|y_i - y_j\|^2\right)$$

    - Discourages label changes except in places where there is also a large change in the observations.

Extension: Conditional Random Fields (CRF)

- Idea: Model conditional instead of joint probability



- Energy formulation

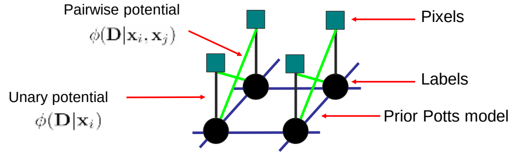
$$E(\mathbf{x}) = \sum_{i \in S} \left(\phi(\mathbf{D}|\mathbf{x}_i) + \sum_{j \in N_i} \left(\phi(\mathbf{D}|\mathbf{x}_i, \mathbf{x}_j) + \psi(\mathbf{x}_i, \mathbf{x}_j) \right) \right) + \text{const}$$

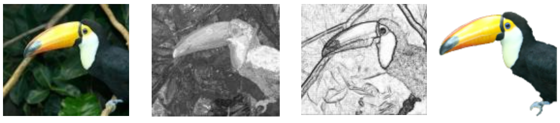
Unary likelihood | Contrast Term | Uniform Prior (Potts Model)

Slide credit: Phil Torr

Example: CRF for Image Segmentation

- CRF structure



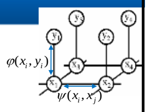


Data (D) | Unary likelihood | Pair-wise Terms | **MAP** Solution

Slide credit: Phil Torr

Energy Minimization

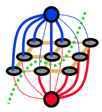
- Goal:
  - Infer the optimal labeling of the MRF.



- Many inference algorithms are available, e.g.
  - Simulated annealing — *What you saw in the movie.*
  - Iterated conditional modes (ICM) — *Too simple.*
  - Belief propagation — *Lecture 9*
  - Graph cuts — *Today*
  - Variational methods
  - Monte Carlo sampling } *For more complex problems*
- Recently, Graph Cuts have become a popular tool
  - Only suitable for a certain class of energy functions.
  - But the solution can be obtained very fast for typical vision problems (~1MPixel/sec).

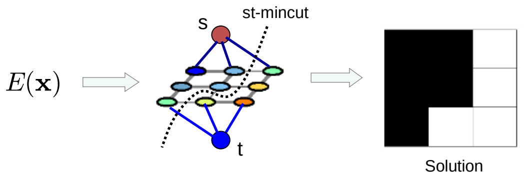
Topics of This Lecture

- Solving MRFs with Graph Cuts
  - Graph cuts for image segmentation
  - s-t mincut algorithm
  - Graph construction
  - Extension to non-binary case
  - Applications



Graph Cuts – Basic Idea

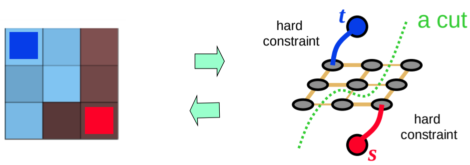
- Construct a graph such that:
  1. Any st-cut corresponds to an assignment of $\mathbf{x}$
  2. The cost of the cut is equal to the energy of $\mathbf{x}$: $E(\mathbf{x})$



Slide credit: Pushmeet Kohli

Graph Cuts for Binary Problems

- Idea: convert MRF into source-sink graph



Minimum cost cut can be computed in polynomial time

(max-flow/min-cut algorithms)

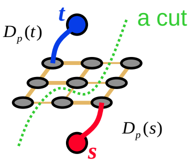
[Boykov & Jolly, ICCV'01]

Simple Example of Energy

unary potentials — pairwise potentials

$$E(L) = \sum_p D_p(L_p) + \sum_{pq \in N} w_{pq} \cdot \delta(L_p \neq L_q)$$

t-links — n-links

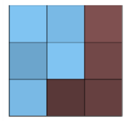


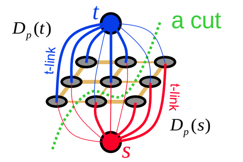
$L_p \in \{s,t\}$

(binary object segmentation)

Slide adapted from Yuri Boykov

Adding Regional Properties





Regional bias example

Suppose I^s and I^t are given "expected" intensities of object and background

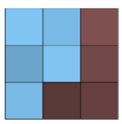
$D_p(s) \propto \exp\left(-\| I_p - I^s \|^2 / 2\sigma^2\right)$

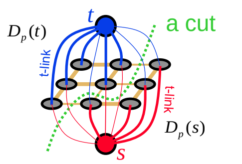
$D_p(t) \propto \exp\left(-\| I_p - I^t \|^2 / 2\sigma^2\right)$

NOTE: hard constrains are not required, in general.

Slide credit: Yuri Boykov [Boykov & Jolly, ICCV'01]

Adding Regional Properties





"expected" intensities of object and background I^s and I^t can be re-estimated

$D_p(s) \propto \exp\left(-\| I_p - I^s \|^2 / 2\sigma^2\right)$

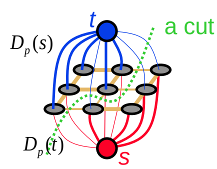
$D_p(t) \propto \exp\left(-\| I_p - I^t \|^2 / 2\sigma^2\right)$

EM-style optimization

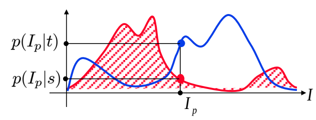
Slide credit: Yuri Boykov [Boykov & Jolly, ICCV'01]

Adding Regional Properties

- More generally, unary potentials can be based on any intensity/color models of object and background.



$$D_p(L_p) = -\log p(I_p|L_p)$$

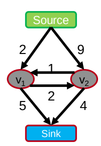


Object and background color distributions

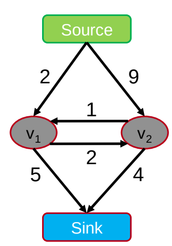
Slide credit: Yuri Boykov [Boykov & Jolly, ICCV'01]

Topics of This Lecture

- Solving MRFs with Graph Cuts
  - Graph cuts for image segmentation
  - s-t mincut algorithm
  - Graph construction
  - Extension to non-binary case
  - Applications



How Does it Work? The s-t-Mincut Problem



Graph (V, E, C)

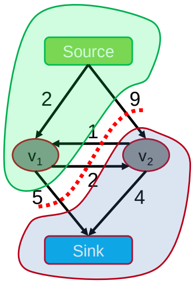
Vertices V = {v_1, v_2 ... v_n}

Edges E = {(v_1, v_2) ....}

Costs C = {$c_{(1, 2)}$ ....}

Slide credit: Pushmeet Kohli

The s-t-Mincut Problem



5 + 2 + 9 = 16

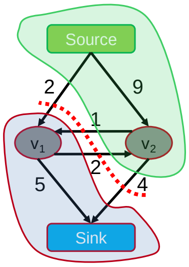
What is an st-cut?

An st-cut (S,T) divides the nodes between source and sink.

What is the cost of a st-cut?

Sum of cost of all edges going from S to T

The s-t-Mincut Problem



2 + 1 + 4 = 7

What is an st-cut?

An st-cut (S,T) divides the nodes between source and sink.

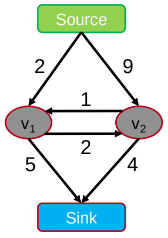
What is the cost of a st-cut?

Sum of cost of all edges going from S to T

What is the st-mincut?

st-cut with the minimum cost

How to Compute the s-t-Mincut?



Solve the dual maximum flow problem

Compute the maximum flow between Source and Sink

Constraints

Edges: Flow < Capacity

Nodes: Flow in = Flow out

Min-cut/Max-flow Theorem

In every network, the maximum flow equals the cost of the st-mincut

History of Maxflow Algorithms

Augmenting Path and Push-Relabel

year	discoverer(s)	bound
1951	Dantzig	$O(n^2mU)$
1955	Ford & Fulkerson	$O(m^2U)$
1970	Dinitz	$O(n^2m)$
1972	Edmonds & Karp	$O(m^2 \log U)$
1973	Dinitz	$O(nm \log U)$
1974	Karzanov	$O(n^3)$
1977	Cherkassky	$O(n^2m^{1/2})$
1980	Galil & Naamad	$O(nm \log^2 n)$
1983	Sleator & Tarjan	$O(nm \log n)$
1986	Goldberg & Tarjan	$O(nm \log(n^2/m))$
1987	Ahuja & Orlin	$O(nm + n^2 \log U)$
1987	Ahuja et al.	$O(nm \log(n\sqrt{\log U}/m))$
1989	Cheriyan & Hagerup	$E(nm + n^2 \log^2 n)$
1990	Cheriyan et al.	$O(n^3/ \log n)$
1990	Alon	$O(nm + n^{8/3} \log n)$
1992	King et al.	$O(nm + n^{2+\epsilon})$
1993	Phillips & Westbrook	$O(nm(\log_{m/n} n + \log^{2+\epsilon} n))$
1994	King et al.	$O(nm \log_{m/(n \log n)} n)$
1997	Goldberg & Rao	$O(m^{3/2} \log(n^2/m) \log U)$ $O(n^{2/3}m \log(n^2/m) \log U)$

n: #nodes

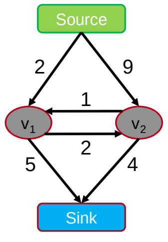
m: #edges

U: maximum edge weight

Algorithms assume non-negative edge weights

Maxflow Algorithms

Flow = 0



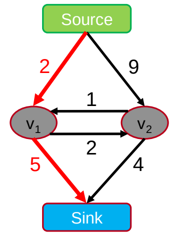
Augmenting Path Based Algorithms

1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Adjust the capacity of the used edges
4. Repeat until no path can be found

Algorithms assume non-negative capacity

Maxflow Algorithms

Flow = 0



Augmenting Path Based Algorithms

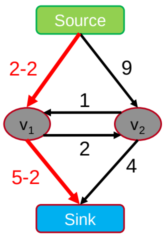
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Adjust the capacity of the used edges
4. Repeat until no path can be found

Algorithms assume non-negative capacity

Maxflow Algorithms

Flow = 0 + 2

Augmenting Path Based Algorithms



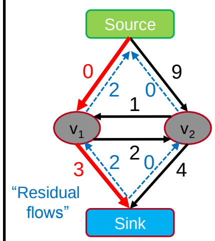
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Adjust the capacity of the used edges
4. Repeat until no path can be found

Algorithms assume non-negative capacity

Maxflow Algorithms

Flow = 2

Augmenting Path Based Algorithms



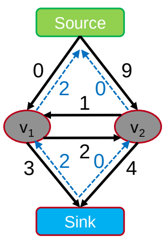
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Adjust the capacity of the used edges and record "residual flows"
4. Repeat until no path can be found

Algorithms assume non-negative capacity

Maxflow Algorithms

Flow = 2

Augmenting Path Based Algorithms



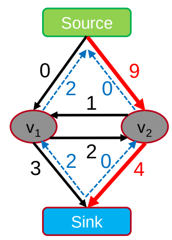
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Adjust the capacity of the used edges
4. Repeat until no path can be found

Algorithms assume non-negative capacity

Maxflow Algorithms

Flow = 2

Augmenting Path Based Algorithms



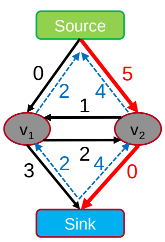
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Adjust the capacity of the used edges
4. Repeat until no path can be found

Algorithms assume non-negative capacity

Maxflow Algorithms

Flow = 2 + 4

Augmenting Path Based Algorithms



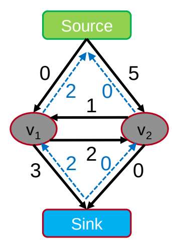
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Adjust the capacity of the used edges
4. Repeat until no path can be found

Algorithms assume non-negative capacity

Maxflow Algorithms

Flow = 6

Augmenting Path Based Algorithms



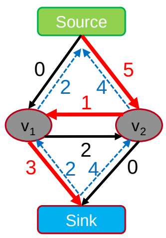
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Adjust the capacity of the used edges
4. Repeat until no path can be found

Algorithms assume non-negative capacity

Maxflow Algorithms

Flow = 6

Augmenting Path Based Algorithms



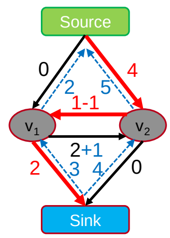
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Adjust the capacity of the used edges
4. Repeat until no path can be found

Algorithms assume non-negative capacity

Maxflow Algorithms

Flow = 6 + 1

Augmenting Path Based Algorithms



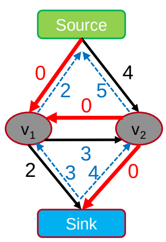
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Adjust the capacity of the used edges
4. Repeat until no path can be found

Algorithms assume non-negative capacity

Maxflow Algorithms

Flow = 7

Augmenting Path Based Algorithms



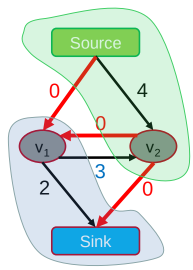
1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Adjust the capacity of the used edges
4. Repeat until no path can be found

Algorithms assume non-negative capacity

Maxflow Algorithms

Flow = 7

Augmenting Path Based Algorithms

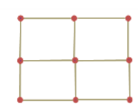


1. Find path from source to sink with positive capacity
2. Push maximum possible flow through this path
3. Adjust the capacity of the used edges
4. Repeat until no path can be found

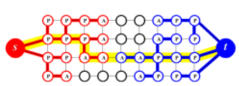
Algorithms assume non-negative capacity

Applications: Maxflow in Computer Vision

- Specialized algorithms for vision problems
  - Grid graphs
  - Low connectivity (m ~ O(n))



- Dual search tree augmenting path algorithm [Boykov and Kolmogorov PAMI 2004]
  - Finds approximate shortest augmenting paths efficiently.
  - High worst-case time complexity.
  - Empirically outperforms other algorithms on vision problems.
  - Efficient code available on the web http://pub.ist.ac.at/~vnk/software.html



When Can s-t Graph Cuts Be Applied?

$$E(L) = \underbrace{\sum_p E_p(L_p)}_{\text{unary potentials, t-links}} + \underbrace{\sum_{pq\in N} E(L_p, L_q)}_{\text{pairwise potentials, n-links}} \qquad L_p \in \{s,t\}$$

- s-t graph cuts can only globally minimize binary energies that are submodular. [Boros & Hummer, 2002, Kolmogorov & Zabih, 2004]

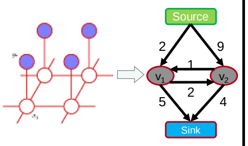
$E(L)$ can be minimized by s-t graph cuts $\Leftrightarrow$ $\boxed{E(s,s)+E(t,t) \le E(s,t)+E(t,s)}$

Submodularity ("convexity")

- Submodularity is the discrete equivalent to convexity.
  - Implies that every local energy minimum is a global minimum.
  - $\Rightarrow$ Solution will be globally optimal.

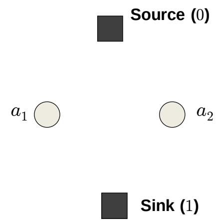
Topics of This Lecture

- Solving MRFs with Graph Cuts
  - Graph cuts for image segmentation
  - s-t mincut algorithm
  - Graph construction
  - Extension to non-binary case
  - Applications



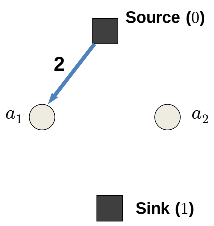
Example: Graph Construction

$E(a_1, a_2)$



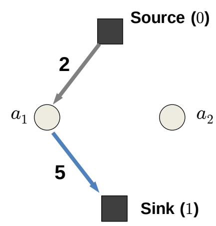
Example: Graph Construction

$E(a_1, a_2) = 2a_1$



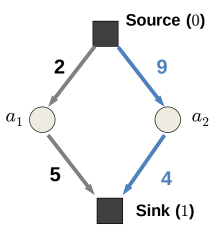
Example: Graph Construction

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



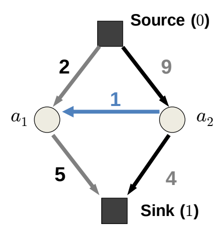
Example: Graph Construction

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



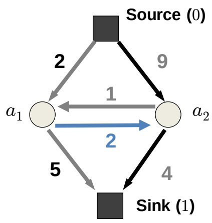
Example: Graph Construction

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



Example: Graph Construction

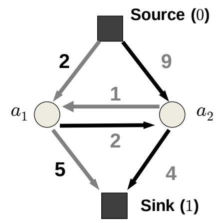
$$E(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + a_1\bar{a}_2 + 2\bar{a}_1 a_2$$



Slide credit: Pushmeet Kohli

Example: Graph Construction

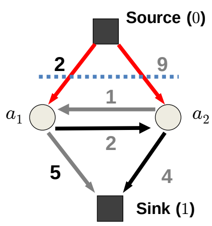
$$E(a_1,a_2) = 2a_1 + 5\bar{a}_1 + 9a_2 + 4\bar{a}_2 + a_1\bar{a}_2 + 2\bar{a}_1 a_2$$



Slide credit: Pushmeet Kohli

Example: Graph Construction

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



Cost of cut = 11

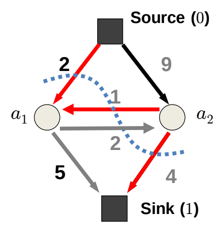
$a_1 = 1 \quad a_2 = 1$

$E\,(1,1) = 11$

Slide credit: Pushmeet Kohli

Example: Graph Construction

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



Cost of cut = 7

$a_1 = 1 \quad a_2 = 0$

$E\,(1,0) = 7$

Slide credit: Pushmeet Kohli

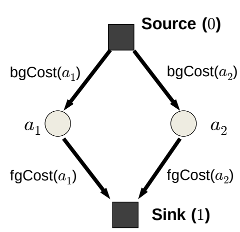
How Does the Code Look Like?

```
Graph *g;
For all pixels p

      /* Add a node to the graph */
      nodeID(p) = g->add_node();

      /* Set cost of terminal edges */
      set_weights(nodeID(p), fgCost(p), bgCost(p));
end

for all adjacent pixels p,q
      add_weights(nodeID(p), nodeID(q), cost);
end

g->compute_maxflow();

label_p = g->is_connected_to_source(nodeID(p));

// is the label of pixel p (0 or 1)
```

Slide credit: Pushmeet Kohli

How Does the Code Look Like?

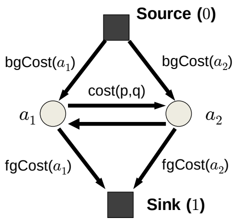
```
Graph *g;
For all pixels p

      /* Add a node to the graph */
      nodeID(p) = g->add_node();

      /* Set cost of terminal edges */
      set_weights(nodeID(p), fgCost(p), bgCost(p));
end

for all adjacent pixels p,q
      add_weights(nodeID(p), nodeID(q), cost);
end

g->compute_maxflow();

label_p = g->is_connected_to_source(nodeID(p));

// is the label of pixel p (0 or 1)
```



Slide credit: Pushmeet Kohli

How Does the Code Look Like?

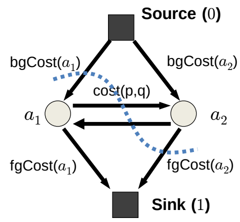
```
Graph *g;
For all pixels p
    /* Add a node to the graph */
    nodeID(p) = g->add_node();
    /* Set cost of terminal edges */
    set_weights(nodeID(p), fgCost(p), bgCost(p));
end
for all adjacent pixels p,q
    add_weights(nodeID(p), nodeID(q), cost);
end
g->compute_maxflow();
label_p = g->is_connected_to_source(nodeID(p));
// is the label of pixel p (0 or 1)
```



Slide credit: Pushmeet Kohli

How Does the Code Look Like?

```
Graph *g;
For all pixels p
    /* Add a node to the graph */
    nodeID(p) = g->add_node();
    /* Set cost of terminal edges */
    set_weights(nodeID(p), fgCost(p), bgCost(p));
end
for all adjacent pixels p,q
    add_weights(nodeID(p), nodeID(q), cost);
end
g->compute_maxflow();
label_p = g->is_connected_to_source(nodeID(p));
// is the label of pixel p (0 or 1)
```

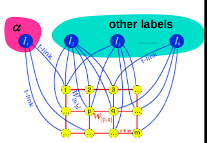


a_1 = bg a_2 = fg

Slide credit: Pushmeet Kohli

Topics of This Lecture

- Solving MRFs with Graph Cuts
  - Graph cuts for image segmentation
  - s-t mincut algorithm
  - Graph construction
  - Extension to non-binary case
  - Applications

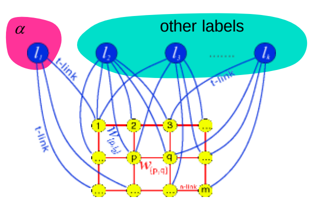


Dealing with Non-Binary Cases

- Limitation to binary energies is often a nuisance.
  ⇒ E.g. binary segmentation only…
- We would like to solve also multi-label problems.
  - The bad news: Problem is NP-hard with 3 or more labels!
- There exist some approximation algorithms which extend graph cuts to the multi-label case:
  - α-Expansion
  - $\alpha\beta$-Swap
- They are no longer guaranteed to return the globally optimal result.
  - But α-Expansion has a guaranteed approximation quality (2-approx) and converges in a few iterations.

α-Expansion Move

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



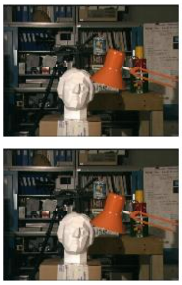
Slide credit: Yuri Boykov

α-Expansion Algorithm

1. Start with any initial solution
2. For each label "α" in any (e.g. random) order:
   1. Compute optimal α-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.

Slide credit: Yuri Boykov

Example: Stereo Vision



Original pair of "stereo" images

Depth map

Slide credit: Yuri Boykov

α-Expansion Moves

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

initial solution
-expansion
-expansion
-expansion
-expansion
-expansion
-expansion
-expansion

For each move, we choose the expansion that gives the largest decrease in the energy: $\Rightarrow$ binary optimization problem

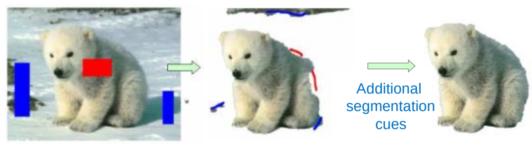
Slide credit: Yuri Boykov

Topics of This Lecture

- Solving MRFs with Graph Cuts
  - Graph cuts for image segmentation
  - s-t mincut algorithm
  - Extension to non-binary case
  - Applications

GraphCut Applications: "GrabCut"

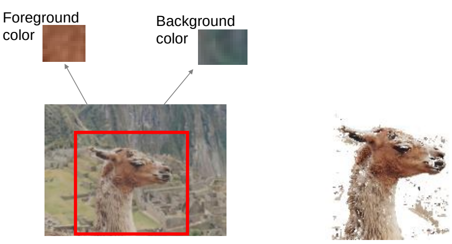
- Interactive Image Segmentation [Boykov & Jolly, ICCV'01]
  - Rough region cues sufficient
  - Segmentation boundary can be extracted from edges
- Procedure
  - User marks foreground and background regions with a brush.
  - This is used to create an initial segmentation which can then be corrected by additional brush strokes.



User segmentation cues

Slide credit: Matthieu Bray

GrabCut: Data Model



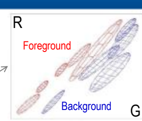
Global optimum of the energy

- Obtained from interactive user input
  - User marks foreground and background regions with a brush
  - Alternatively, user can specify a bounding box

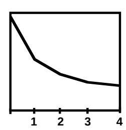
Slide credit: Carsten Rother

Iterated Graph Cuts

Result



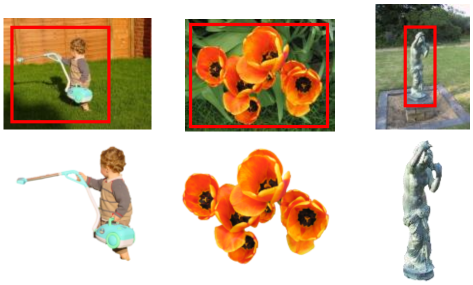
Color model
(Mixture of Gaussians)



Energy after each iteration

Slide credit: Carsten Rother

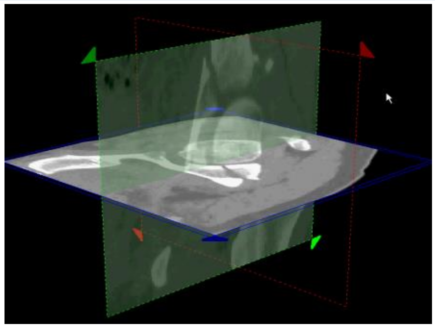
GrabCut: Example Results



This is included in all MS Office versions since 2010!

Image source: Carsten Rother

Applications: Interactive 3D Segmentation



Slide credit: Yuri Boykov

[Y. Boykov, V. Kolmogorov, ICCV'03]

References and Further Reading

- A gentle introduction to Graph Cuts can be found in the following paper:
  - Y. Boykov, O. Veksler, Graph Cuts in Vision and Graphics: Theories and Applications. In *Handbook of Mathematical Models in Computer Vision*, edited by N. Paragios, Y. Chen and O. Faugeras, Springer, 2006.
- Try the GraphCut implementation at http://pub.ist.ac.at/~vnk/software.html