Putting it all together – Unifying frameworks.

Filip Malmberg

Unifying frameworks

- Many of the algorithms presented in the course are closely related.
- There have been attempts to investigate the theoretical relationship between the various methods.
- In this lecture, we will look at two "unifying frameworks":
  - In part 1, we look at a theoretical framework that unifies many of the methods for seeded segmentation that we have covered in this course.
  - In part 2, we will see that two types of hard constraints (seeds and boundary constraints) used in interactive segmentation can be seen as special cases of a more general type of constraints.

Part 1: Power waterheds

Camille Couprie, Leo Grady, Laurent Najman and Hugues Talbot
Power Watersheds: A Unifying Graph-Based Optimization Framework
IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 33, No. 7, pp. 1384-1399, 2011.

Ali Kemal Sinop and Leo Grady *A Seeded Image Segmentation Framework Unifying Graph Cuts and Random Walker Which Yields A New Algorithm*
Proc. of ICCV, 2007

Recap, Random walker

Find a mapping $x : V \rightarrow [0, 1]$ that minimizes

$$\left(\sum_{e_{ij} \in E} (w_{ij} |x_i - x_j|)^2 \right)^{1/2} , \tag{1}$$

subject to $x(F) = 1$ and $x(B) = 0$. A final segmentation s is given by

$$s_i = \begin{cases} 1 & \text{if } x_i \geq \frac{1}{2} \\ 0 & \text{if } x_i < \frac{1}{2} \end{cases} . \tag{2}$$

Recap, Random walker

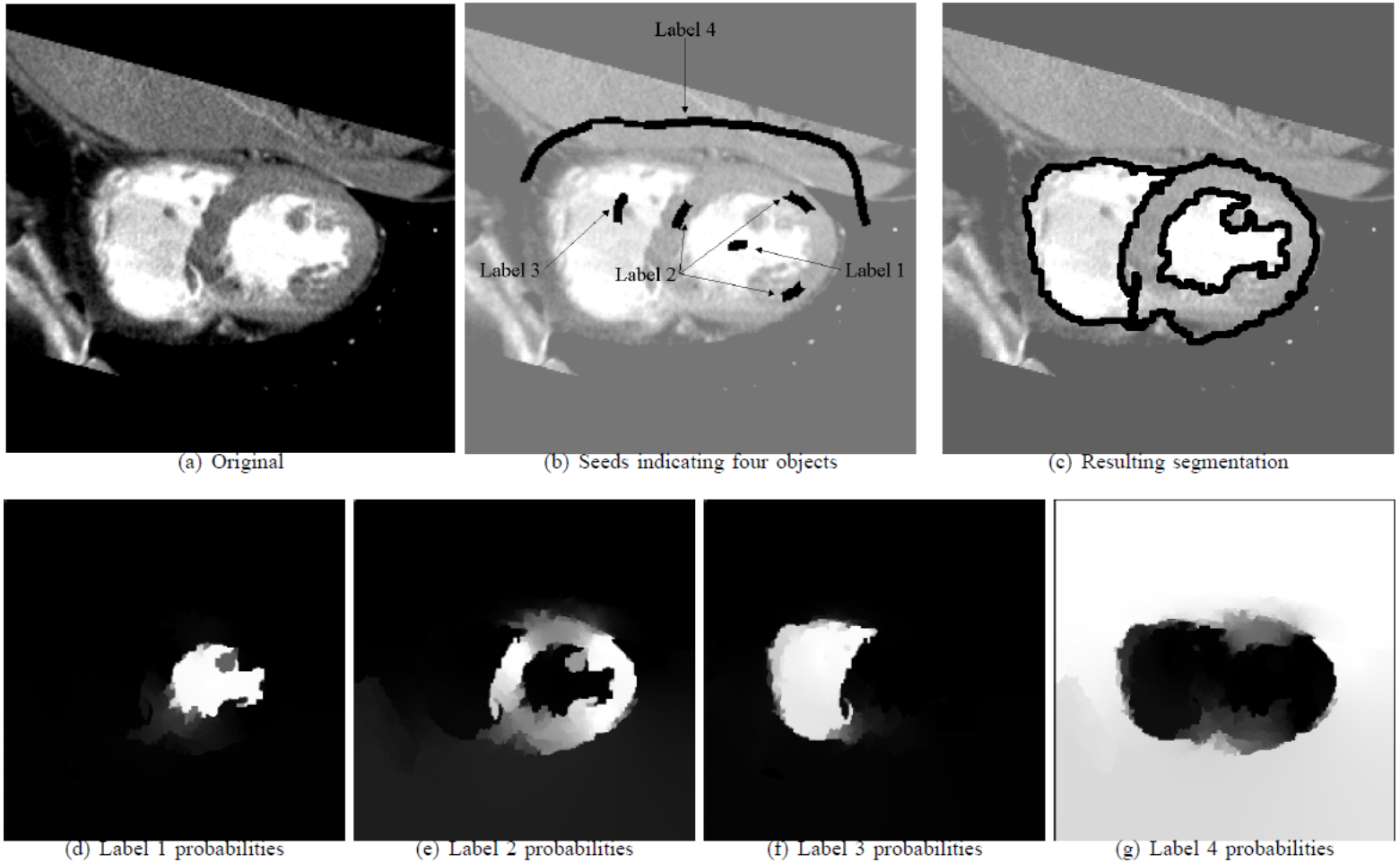


Figure 1: Seeded segmentation by random walker.

A general formulation of seeded segmentation

- In other words, the Random Walker method tries to minimize the l_2 norm of the difference in x between adjacent vertices.
- We have previously seen that the l_2 norm is a special case of a l_p norm.
- What happens if we try to extend the Random Walker method to other l_p norms?

A general formulation of seeded segmentation

Find a labeling $x : V \longrightarrow [0, 1]$ that minimizes

$$\left(\sum_{e_{ij} \in E} (w_{ij} |x_i - x_j|)^q \right)^{1/q} , \tag{3}$$

subject to $x(F) = 1$ and $x(B) = 0$. A final segmentation s is given by

$$s_i = \begin{cases} 1 & \text{if } x_i \geq \frac{1}{2} \\ 0 & \text{if } x_i < \frac{1}{2} \end{cases} . \tag{4}$$

A general formulation of seeded segmentation

In the next few slides, we will see that the general formulation includes many of the algorithms we have convered in this course as special cases!

- For $p = 1$, we get the max flow/min cut problem.
- For $p = 2$, we get the Random walker problem. (By definition)
- For $p = \infty$, we get the shortest path problem.
- We will also extend the general formulation so that it includes minimum spanning forests/watersheds.

Case $q = 1$, Minimal graph cuts

If we substitute $q = 1$ into (3), we get

$$\sum_{e_{ij} \in E} w_{ij} |x_i - x_j| \ . \tag{5}$$

It was shown in [3] that minimizing this equation subject to $x(F) = 1$ and $x(B) = 0$ is equivalent (dual) to the max flow problem. Thus, (6) can be minimized using, e.g., the Ford-Fulkerson algorithm described in lecture 4.

Case $q = \infty$, Shortest paths

If we let q approach ∞, we obtain the problem of minimzing

$$\max_{e_{ij} \in E} w_{ij} |x_i - x_j| \ , \tag{6}$$

subject to $x(F) = 1$ and $x(B) = 0$. It was shown in [3] that this is equivalent to segmentation by shortest path forests. Thus it can be solved by Dijkstra's algorithm.

Extending the framework to watersheds

To incorporate watersheds into the general framework, we separate the exponent on the weights from the exponent on the variables. We thus seek to minimize

$$\sum_{e_{ij} \in E} w_{ij}^{p} |x_i - x_j|^{q} \ . \tag{7}$$

subject to $x(F) = 1$ and $x(B) = 1$. When $p = q$, this is equivalent to the previous formulation (we can skip the root). When q is finite and $p \to \infty$, the results of the above optimization problem converges to MSF cuts (watersheds).

Unary terms

- So far, we have only considered binary terms ("interaction" between pairs of vertices).
- We can extend (7) further by including unary terms:

$$\sum_{e_{ij} \in E} w_{ij}^p |x_i - x_j|^q + \sum_{v_i \in V} w_{Fi}^p |x_i|^q + \sum_{v_i \in V} w_{Bi}^p |x_i - 1|^q \, . \qquad (8)$$

- The unary terms can be incorporated by adding "phantom" seeds V_F and V_B.

Unary terms

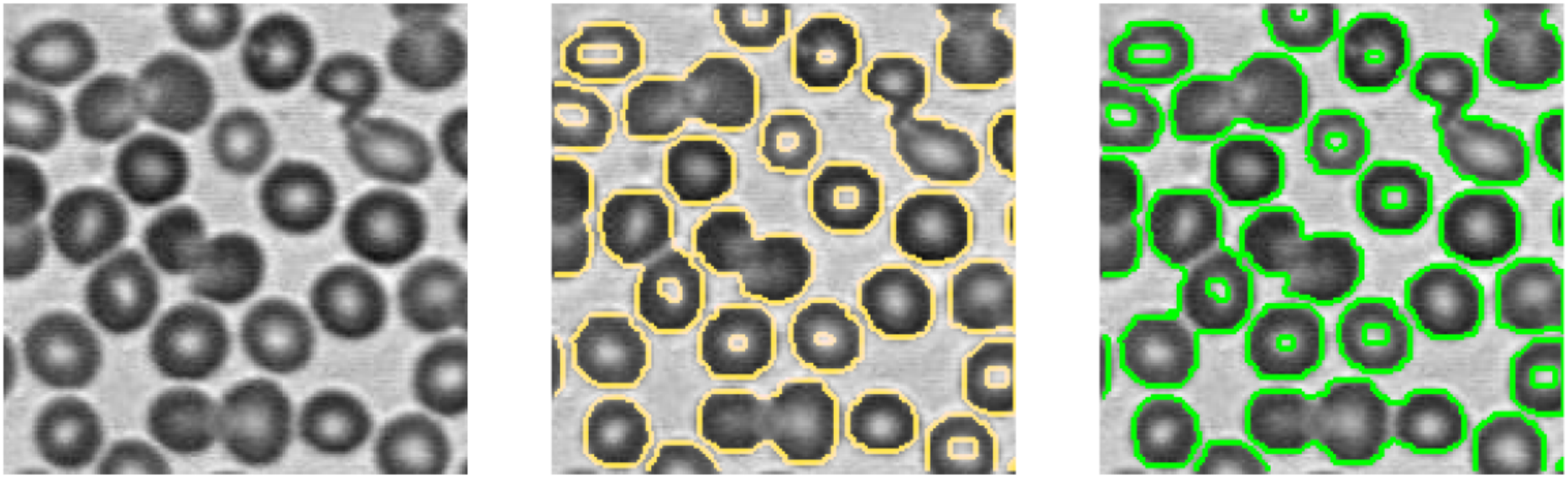
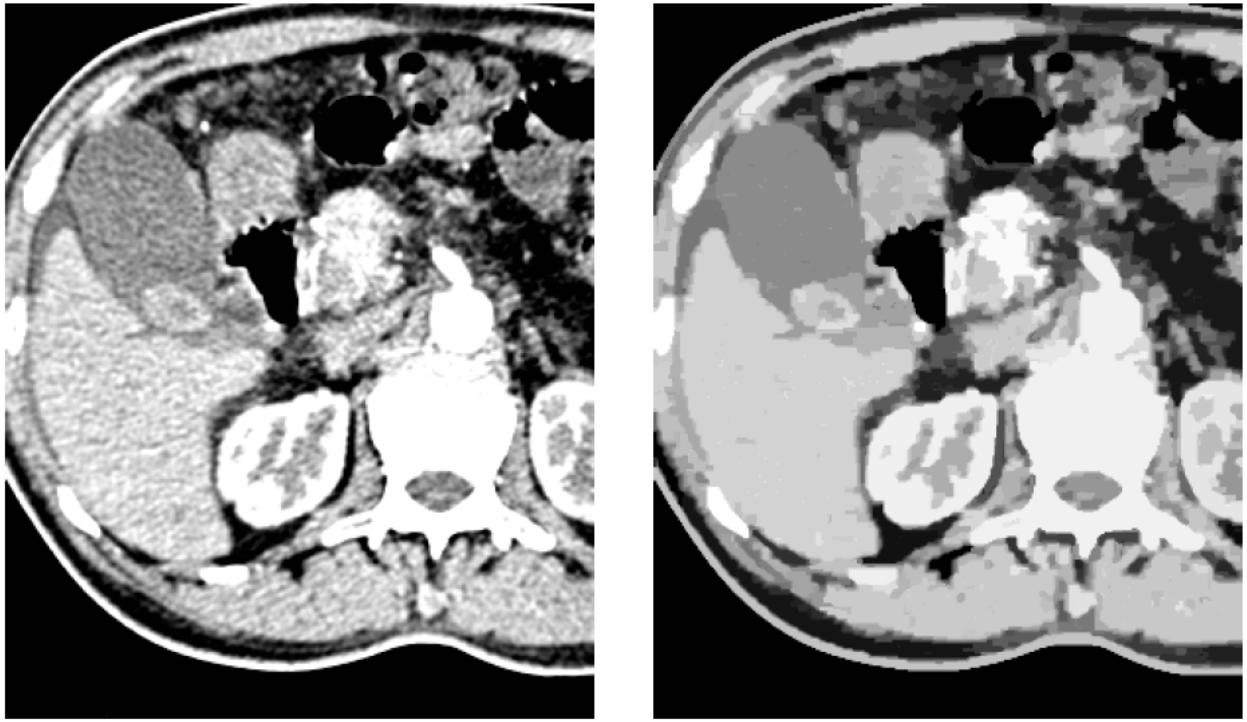


Figure 2: Unseeded segmentation with unary terms. (a) Image. (b) Segmentation by graph cuts. (c) Segmentation by watersheds.

Unary terms

In [1], Power watersheds with unary terms were used to compute anisotropic diffusion.



(a) Original image (b) PW result

Figure 3: Anisotropic diffusion with Power Watershed.

So, which method is better?

- Given the similarity between the presented method for seeded segmentation, how do we decide which one to use?
- In [2], an empirical comparison between a number of methods was presented.
- The study is based on the "Grabcut" database from Microsoft (available online). This dataset consists of 50 "natural" images provided with seeds and ground truth segmentations.

So, which method is better?

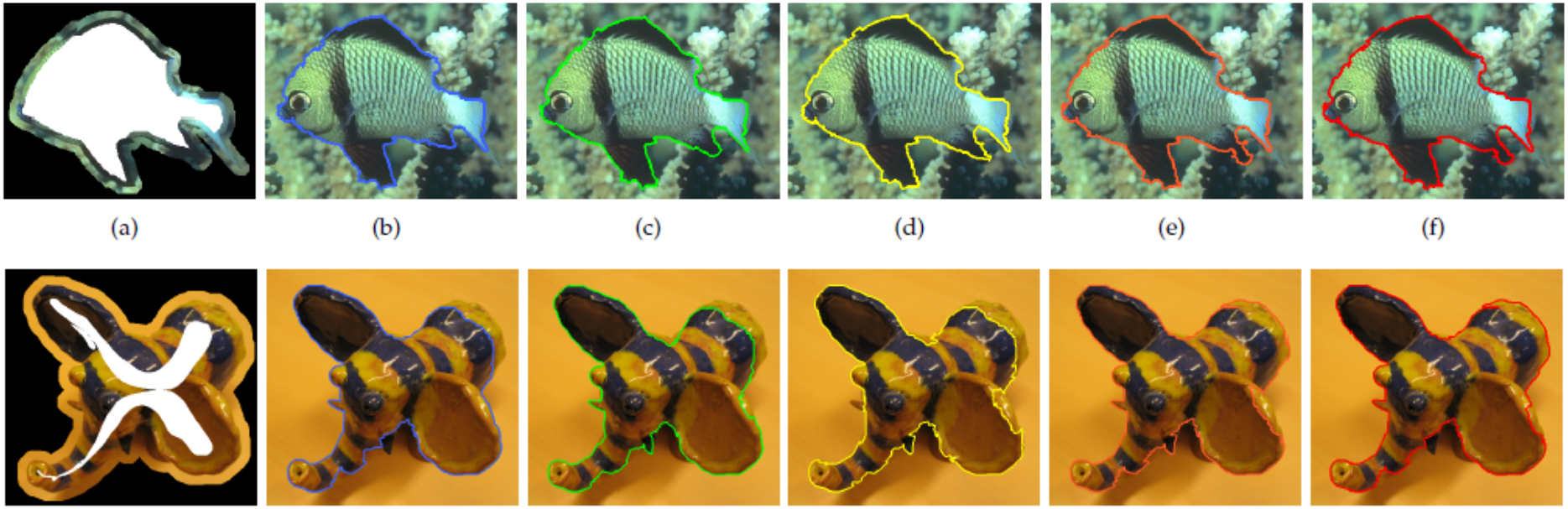


Figure 4: Example segmentations using the provided (top images) and skeletonized (bottom images) set of seeds on the Grabcut database images: (a) Seeds, (b) Graph cuts, (c) Random walker, (d) Shortest path, (e) Maximum spanning forest (standard watershed), and (f) Power watershed (q = 2).

Empirical comparison 1

	BE	RI	GCE	VoI	**Average rank**
Shortest paths	2.82	0.972	0.0233	0.204	**1**
Random walker	2.96	0.971	0.0234	0.204	**2.25**
MSF (Prim)	2.89	0.971	0.0244	0.209	**2.5**
Power wshed ($q = 2$)	2.87	0.971	0.0245	0.210	**3.25**
Graph cuts	3.12	0.970	0.0249	0.212	**5**

Figure 5: Results of comparison with symmetrically eroded seeds.

Empirical comparison 2

	BE	RI	GCE	VoI	**Average rank**
Graph cuts	4.70	0.953	0.0380	0.284	**1**
Power wshed ($q = 2$)	4.93	0.951	0.0407	0.297	**2.5**
Random walker	5.12	0.950	0.0398	0.294	**2.75**
MSF (Prim)	5.11	0.950	0.0408	0.298	**3.5**
Shortest paths	5.33	0.947	0.0426	0.308	**5**

Figure 6: Results of comparison with asymmetrically eroded seeds.

Computation time

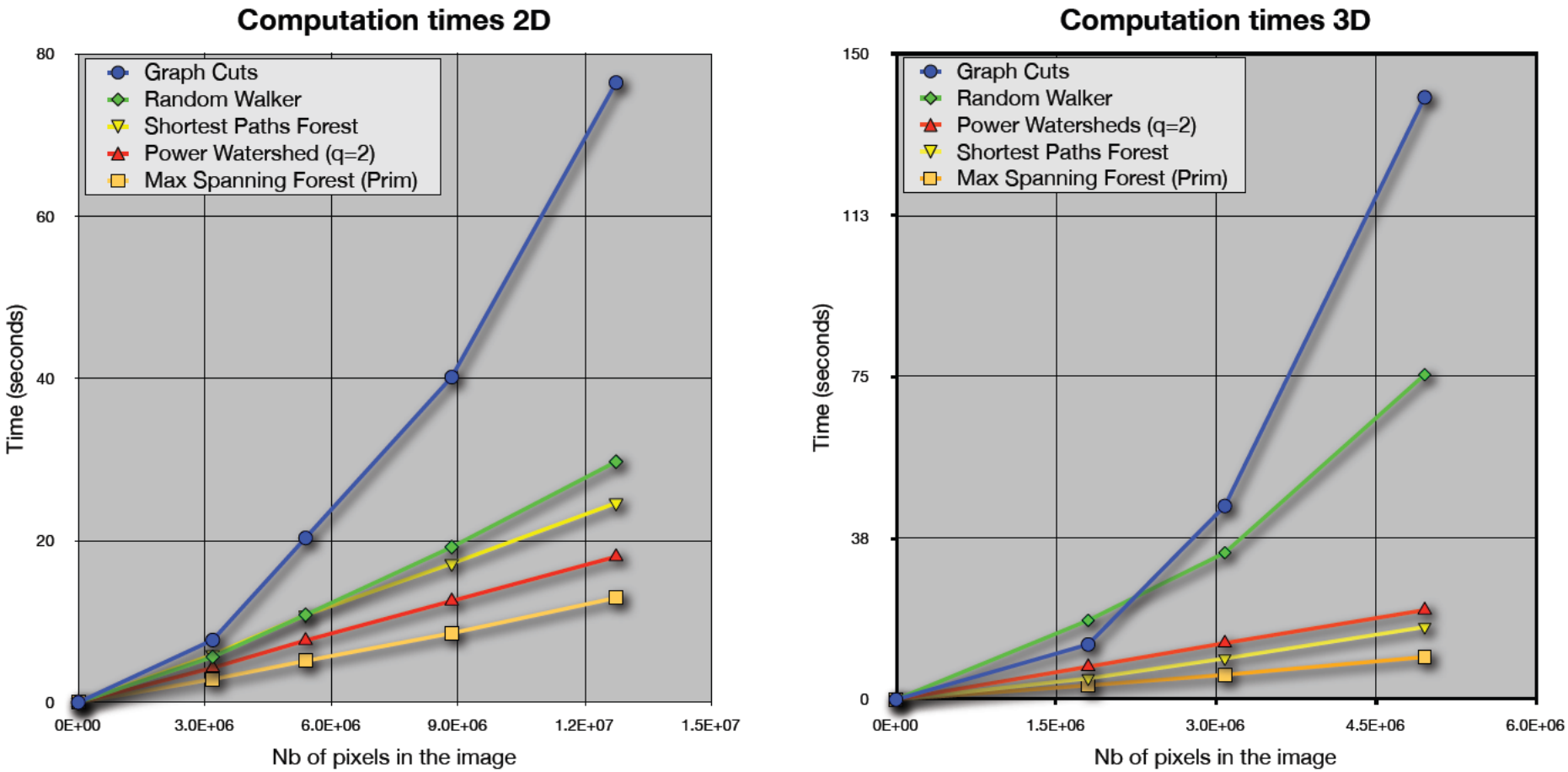


Figure 7: Computation time for the different algorithms in 2D and 3D.

Qualitative comparison

Min cut/max flow

+ Global optimization of weighted "area" (sum of edge weights in the cut).
+ Possible to approximate continuous "cut metrics" with arbitrary precision.
- Shrinking bias.
- Metrication artifacts on standard grids.
- NP-hard for more than two labels.
- Slower computation.

Qualitative comparison

Shortest paths

+ No shrinking bias.
+ Allows any number of labels.
+ Fast computation. Computation time indepent of the number of labels.
- Metrication artifacts on standard grids.
- Sensitive to noise and missing boundaries.

Qualitative comparison

MSF cuts

+ Global optimization of the max-norm of the cut.
+ Provably robust to variations in seed-point placement.
+ No shrinking bias.
+ Allows any number of labels.
+ Fast computation. Computation time indepent of the number of labels.
- Very sensitive to noise and leaks. (no penalty for "long" boundaries)

Qualitative comparison

Random walker

+ No shrinking bias.
+ Allows any number of labels.
+ No metrication artifacts.
+ Tolerant to noise and missing boundaries.
- Computation time depent of the number of labels.
- Slower computation.

Conclusions, Part 1

- Many of the methods for seeded segmentation that we have seen in this course (RW, GC, MSF, SPF) can be formulated as minimizing the l_p norm of the gradients of a potential field with boundary conditions.
- The theoretical framework does not directly provide algorithms for optimizing the different cases, but it provides theoretical insight into the similarities and differences between the methods.
- The general optimization problem of seeded segmentation can be extended to include unary terms. This allows, e.g., the use of watersheds for general optimization in computer vision.
- We have looked at an empirical study that compares various methods for seeded segmentation.

Part 2: Generalized hard constraints for graph segmentation

Filip Malmberg, Robin Strand, Ingela Nyström
Generalized Hard Constraints for Graph Segmentation
In Proceedings of SCIA 2011.

Segmentation as constrained optimization

- As we have seen in the course, image segmentation can often be phrased as an optimization problem.
- In graph-based interactive segmentation, we seek to find a cut (or labeling) that best matches the image content, while satisfying a set of constraints given by the user.
- These constraints are typically given in one of two forms:
  - *Regional constraints* where the cuts is required to separate all elements in a specified subset of the graph vertices. (e.g. seeded segmentation)
  - *Boundary constraints* where the cut is required to include a specified subset of the graph edges. (e.g. live-wire)

Generalized hard constraints

- We will now look at a generalized type of hard constraints for supervised segmentation.
- Informally, a generalized constraint is a pair of vertices that must be separated by any feasible cut.
- Both regional and boundary constraints can be seen as special cases of these generalized constraints.
- We will also look at a method for computing cuts that satisfy generalized constraints.

Segmentation with generalized hard constraints

(a) (b) (c)

Figure 8: Interactive segmentation of the liver in a slice from a CT volume image, using three different interaction paradigms. (a) Segmentation using boundary constraints. (b) Segmentation using regional constraints. (c) Segmentation using generalized constraints.

Preliminaries

- Let $S \subseteq$. We denote by $G - S$ the graph $(V, E \setminus S)$.
- We recall the definition of graph cuts.
  - Let $S \subseteq E$. If, for all $e_{v,w} \in S$, it holds that $v \underset{G-S}{\not\sim} w$, then S is a *(graph) cut* on G.
- Let S be a cut on G, and let $e \in S$. The *segment* S_e of S corresponding to e is defined as

$$S_e = \{e_{v,w} \mid e_{v,w} \in S,\ v \underset{G-S}{\sim} w\} \,. \tag{9}$$

Graph cut segments

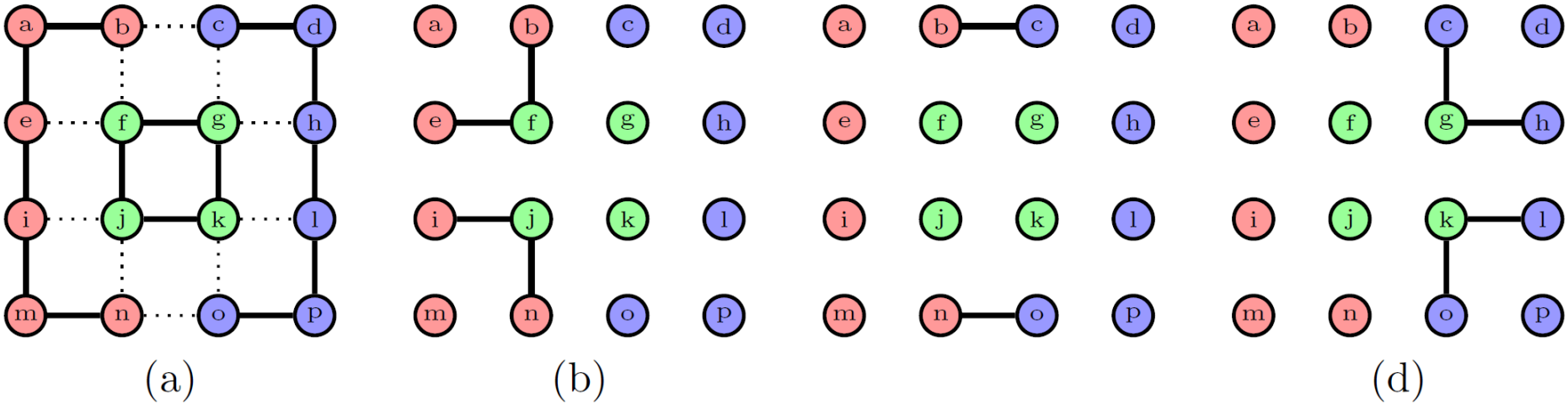


Figure 9: Graph cut segments.

Generalized constraints

- A constraint on G is an unordered pair of distinct vertices in V.
- Let $S \subseteq E$ and let C be a set of constraints. We say that S satisfies C if

$$\forall c_{v,w} \in C, v \underset{G-S}{\not\sim} w \tag{10}$$

and

$$\forall e \in S, \exists c_{v,w} \in C \text{ such that } v \underset{G-(S\setminus\{e\})}{\sim} w \ . \tag{11}$$

Generalized constraints

In other words, a cut satisfies a set of constraints iff

- It is not an *under-segmentation* (The cut separates all constraint pairs)
- It is not an *over-segmentation* (There is no strict subset of the cut that satisfies the above property)

Some nice properties

- If $S \subseteq E$ satisfies some set of constraints C, then S is a cut.
- Any set of regional or boundary constraints can be written as an equivalent set of generalized constraints.
- For any set of generalized constraints, there exists at least one feasible cut.
- Any feasible cut can be computed using a "region merging" procedure.

Regional and Boundary constraints

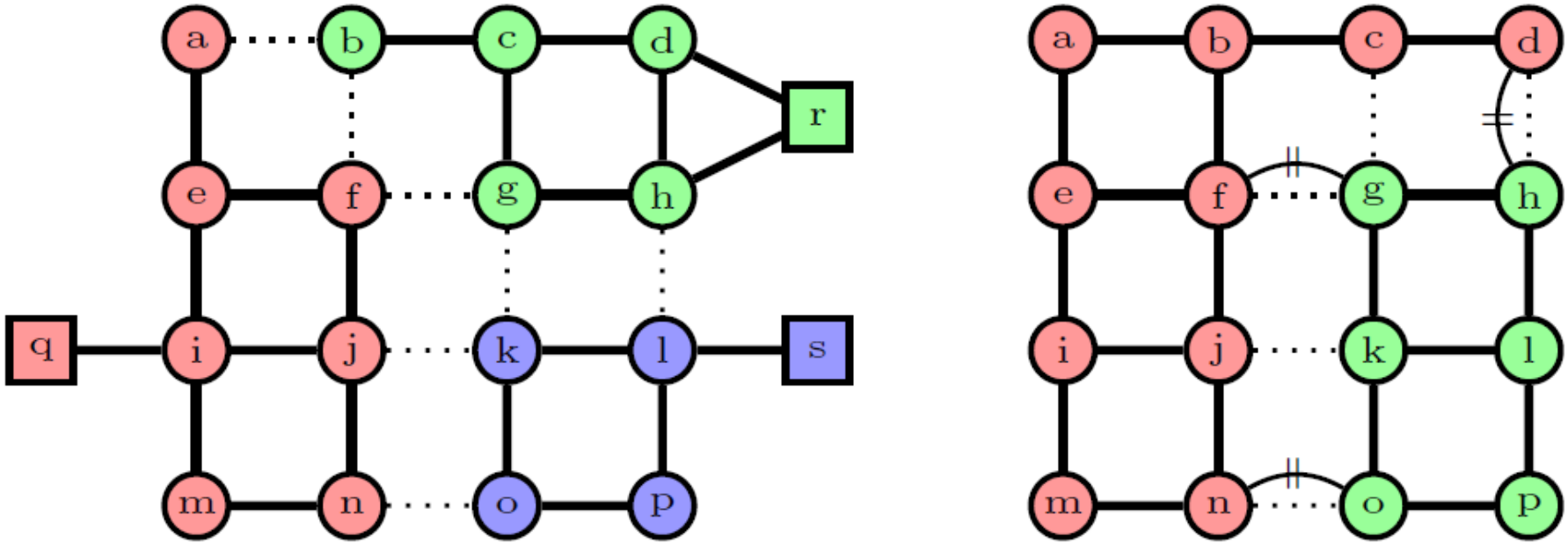


Figure 10: Graph cuts with respect to regional and boundary constraints.

Regional constraints

- A regional constraint is a vertex in V.
- Informally, a cut S satisfies a set of regional constraints $C_r \subseteq V$ if each connected component of $G - S$ contains exactly one element in C_r.
- For any set of regional constraints C_r, there exists a set of generalized constraints C, namely $C = \{c_{v,w} \mid v, w \in C_r\}$, such that $S \subseteq E$ satisfies C iff it satisfies C_r.

Boundary constraints

- A boundary constraint is an edge in E.
- A cut S satisfies a set of boundary constraints $C_b \subseteq E$ if
  - $C_b \subseteq S$.
  - Each segment of S contains at least one element of C_b.
- Note that edges and generalized constraints are both defined as unordered pairs of vertices. If a set of generalized constraints is a subset of E, then the definition of feasible cuts with respect to generalized constraints coincides with the above definition.

Merging operations and mergeable edges

- If $S \subseteq E$ is a cut on G, then each segment of S form a boundary between exactly two connected components of $G - S$. Therefore, the removal of a segment from a cut is called a *merging operation*.
- We use the following notation:

$$S \oslash e = S \setminus S_e \, . \tag{12}$$

- An edge $e \in S$ is said to be *mergeable* with respect to a set of constraints C if $S \oslash e$ is not an undersegmentation with respect to C.

A general algorithm

We now outline a general algorithm for computing a cut that satisfies a set C of generalized constraints.

- Start from a cut S that is not an undersegmentation w.r.t C. (E will do!)
- While there exists a mergeable edge e in S, set $S \leftarrow S \oslash e$.

At the termination of this algorithm, S satisfies C.

The general algorithm is "complete"

If S is a cut that satisfies a set of constraints C, the there exists a sequence of mergeable edges $\langle e_1, e_2, \ldots, e_k \rangle$ such that

$$S = E \oslash e_1 \oslash e_2 \oslash \ldots \oslash e_k \ . \tag{13}$$

In other words, *any* feasible cut can be computed using the general algorithm.

A greedy algorithm

We are interested in finding a cut that best matches the image content. Here, we assume that the edge weights represent *similarity* between adjacent vertices, and reformulate the general algorithm as follows:

- Start from a cut S that is not an undersegmentation w.r.t C. (E will do!)
- While there exists a mergeable edge in S, select a mergeable edge e for which the weight is maximum and set $S \leftarrow S \oslash e$.

The greedy algorithm is optimal

- Define the weight of a cut S as

$$W(S) = \max_{e \in S} W(e) \, . \tag{14}$$

- If S is a cut computed according to the greedy algorithm, then the weight of S is smaller than or equal to the weight of any other feasible cut.
- (Note the similarity with MSF:s)

Conclusions, Part 2

- The generalized hard constraints described in this lecture unify two common paradigms for user interaction in interactive segmentation.
- This allows us to develop general purpose segmentation algorithms, that are not restricted to a particular paradigm for user input.
- We have presented *one* such method, that computes cuts that are globally optimal with respect to the max-norm (e.g. closely related to MSF cuts). The development of other such algorithms is a field open for research!

References

[1] Camille Couprie, Leo Grady, Laurent Najman, and Hugues Talbot.

Anisotropic diffusion using power watersheds.

In *Proceedings of ICIP*, 2010.

[2] Camille Couprie, Leo Grady, Laurent Najman, and Hugues Talbot.

Power watersheds: A unifying graph-based optimization framework.

IEEE Transactions on Pattern Analysis and Machine Intelligence, 33(7), 2011.

doi:10.1109/TPAMI.2010.200.

[3] Ali Kemal Sinop and Leo Grady.

A seeded image segmentation framework unifying graph cuts and random walker which yields a new algorithm.

In *Proc. of ICCV 2007*. IEEE, 2007.