Random walks and anisotropic interpolation on graphs.

Filip Malmberg

Interpolation of missing data

- Assume that we have a graph where we have defined some (real) values for a subset of the nodes, and that we want to somehow "fill in" the missing data for the remaining nodes.

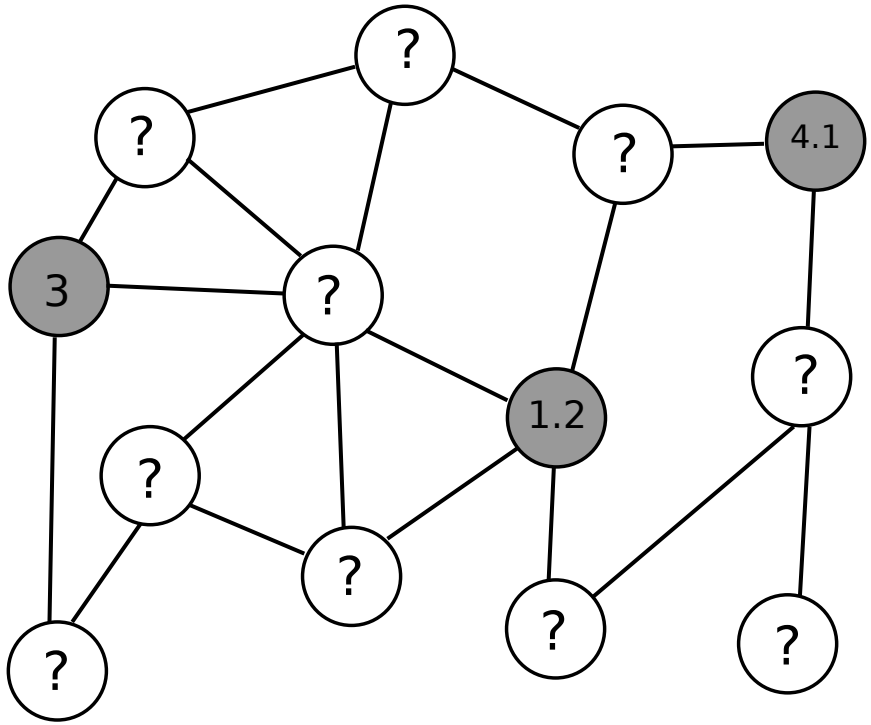


Figure 1: Interpolation of missing data.

Part 1: Smooth interpolation

Interpolation of missing data

- In the absence of other information, we might seek a solution that is "smooth" in some sense.
- Intuitively, a function is smooth if the value of a vertex is similar to that of its neighbors.
- In this lecture, we will consider a method for finding solutions $\mathbf{x}$ where

$$\sum_{w \in \mathcal{N}(v)} w(e_{vw}) |x_v - x_w|^2 \tag{1}$$

  is as small as possible for all $v \in V$.

Continuous Dirichlet problem

The *Dirichlet integral* may by defined as

$$D[u] = \frac{1}{2} \int_{\Omega} |\nabla u|^2 d\Omega \, . \tag{2}$$

for a field u and region Ω. A *harmonic function* is a function that satisfies the *Laplace equation*

$$\nabla^2 u = 0 \, . \tag{3}$$

The problem of finding a harmonic function subject to its boundary values is called the *Dirichlet problem*. The harmonic function that satisfies the boundary conditions minimizes the Dirichlet integral.

Harmonic functions

Harmonic functions have three properties that make them suitable for generating "smooth" interpolations:

- The *mean value theorem* states that the value at each point in the interior (i.e., not a boundary point) is the average value of its neighbors.
- The *maximum principle* states that harmonic functions may not take values on interior points that are greater (or less) than the values taken on the boundary.
- The Dirichlet integral is minimized by harmonic functions. This means that the integral of the gradient magnitudes for the system will be minimized, subject to fixed boundary conditions.

Moving to discrete calculus

- We will now formulate a disrete (combinatorial) version of the Dirichlet problem. This is one example of *discrete calculus* - an attempt to formulate a theoretical framework that provides a discrete counterpart of the continuous calculus.
- In contrast to traditional goals of finding an accurate discretization of conventional multivariate calculus, discrete calculus establishes a separate, equivalent calculus that operates purely in the discrete space without any reference to an underlying continuous process.
- A book on this subject ([4]) is available in the CBA library.

Preliminaries

- We consider a undirected graph $G = (V, E)$.
- We associate each edge $e \in E$ with a real valued weight $w(e) > 0$.
- The *degree* of a vertex v_i is defined as

$$d_i = \sum w(e_{ij}) \tag{4}$$

for all edges e_{ij} incident on v_i.

The discrete Dirichlet problem

We define the discrete Laplacian matrix as

$$L_{v_i v_j} = \begin{cases} d_{v_i} & \text{if } i = j \\ -w_{ij} & \text{if } v_i \text{ and } v_j \text{ are adjacent nodes} \\ 0 & \text{otherwise} \end{cases} . \tag{5}$$

where $L_{v_i v_j}$ is used to indicate that the matrix L is indexed by vertices v_i and v_j.

The discrete Dirichlet problem

- Let $\mathbf{x}$ be a vector of length $|V|$, representing a value at each vertex of the graph.
- A combinatorial formulation of the Dirichlet integral is

$$\mathbf{x}^T L \mathbf{x} \, . \tag{6}$$

- We seek a function $\mathbf{x}$ that minimizes (6) subject to the boundary conditions. Such a function is called a *combinatorial harmonic*.

A closer look at the combinatorial Dirichlet integral

What does $\mathbf{x}^T L\mathbf{x}$ mean? Let's spell it out:

- First, $L\mathbf{x}$ is a vector, whose elements are sums of difference equations

$$\sum_{w \in \mathcal{N}(v)} w(e_{vw})(x_v - x_w) \,, \tag{7}$$

  where x_v is the element in $\mathbf{x}$ that corresponds to the vertex v.
- Thus, $\mathbf{x}^T L\mathbf{x}$ is a scalar, that can be written as follows:

$$\sum_{v \in V} \sum_{w \in \mathcal{N}(v)} w(e_{vw}) |x_v - x_w|^2 \,. \tag{8}$$

A closer look at the combinatorial Dirichlet integral

What does $\mathbf{x}^T L\mathbf{x}$ mean? Let's spell it out:

- First, $L\mathbf{x}$ is a vector, whose elements are sums of difference equations

$$\sum_{w \in \mathcal{N}(v)} w(e_{vw})(x_v - x_w) \,, \tag{9}$$

  where x_v is the element in $\mathbf{x}$ that corresponds to the vertex v.
- Thus, $\mathbf{x}^T L\mathbf{x}$ is a scalar, that can be written as follows:

$$\sum_{v \in V} \sum_{w \in \mathcal{N}(v)} w(e_{vw})|x_v - x_w|^2 \,. \tag{10}$$

What is the role of the edge weights?

- Since we have required the edge weights to be greater than zero, we can interpret $1/w(e_{vw})$ as a distance between vertices v and w.
- Thus, the weights define a *metric*.

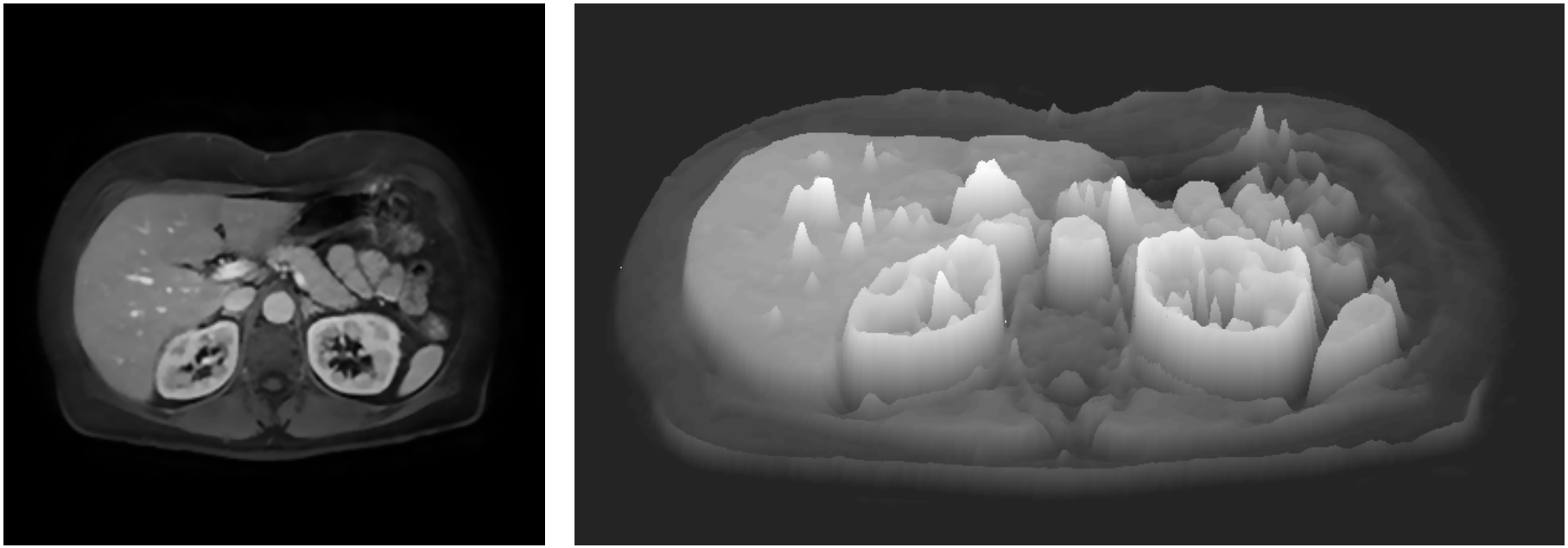


Figure 2: A 2D image, interpreted as a 3D surface.

The Discrete Dirichlet problem, solution

Partition the vertices of the graph into two sets, V_M (marked nodes/seedpoints) and V_U (unmarked nodes). We can then reorder the matrix L to reflect the subsets

$$L = \begin{bmatrix} L_M & B \\ B^T & L_U \end{bmatrix} . \tag{11}$$

The Discrete Dirichlet problem, solution

Let x_m be the (known) values at the marked vertices, and let x_u be the (unknown) values of the unmarked vertices. As shown in [5, 1], the solution to the combinatorial Dirichlet problem may be found by solving

$$L_U x_u = -B x_m \ . \tag{12}$$

This is a sparse, symmetric, positive-definite, system of linear equations with $|x_u|$ number of equations. If the graph is connected (or every connected component contains a marked vertex) then L_U is guaranteed to be non-singular. Thus the solution is guaranteed to exist and be unique.

Numerical practicalities

- We have seen that the Dirichlet problem can be reduced to solving a large, sparse, symmetric, linear system of equations. There are, of course, many methods for doing this.
- Direct methods (i.e. LU decomposition etc.) do not preserve the sparseness of the matrix, and are therefore not practical for the very large systems we consider here.
- Instead, iterative solvers can be used. See [3, 2] for details. In [3], a computation time of a few seconds was reported for a 1024×1024 image.

Interpolation, examples

Figure 3: Interpolating missing image information by solving the combinatorial Dirichlet problem.

Interpolation, examples

Figure 4: Interpolating missing image information by solving the combinatorial Dirichlet problem. (Left) Using weights determined by vertex distances. (Right) Using weights determined by vertex distances and image content.

Connection with random walks

We will now look at another interpretation of the combinatorial Dirichlet problem, describe in [2].

- Imagine a "random walker" moving through graph by stepping between adjacent vertices.
- At each step, the probability of going from the current node to one of its neighbors is given by the weight of the edge connecting those nodes.
- Assume that each marked vertex has either value 0 or 1. Then the solution to the combinatorial Dirichlet problem answers, for each unmarked vertex, the question "Given a random walker starting at this vertex, what is the probability that it first reaches a marked node with label 1?"

Application to seeded segmentation

We will now look at how these techniques can be used for seeded segmentation.

- For binary image segmentation, we can assign a value of 1 to foreground seeds and a value of 0 to background seeds. All non-seed vertices have unknown values. By solving this Dirichlet problem we obtain, for each vertex, the probabilite that a random walker starting at the node reaches a foreground seed first. If this probability is greater than $\frac{1}{2}$, we say that the vertex belongs to the foreground.
- For K object labels, we need to solve $K - 1$ linear systems (The probabilities must sum to one). This gives us, for all vertices and all labels, the probability that a random walker starting at the vertex reaches a seed with the given label first. A final segmentation is obtained by assigning to each vertex the label of the seed it is expected to reach first.

Application to seeded segmentation

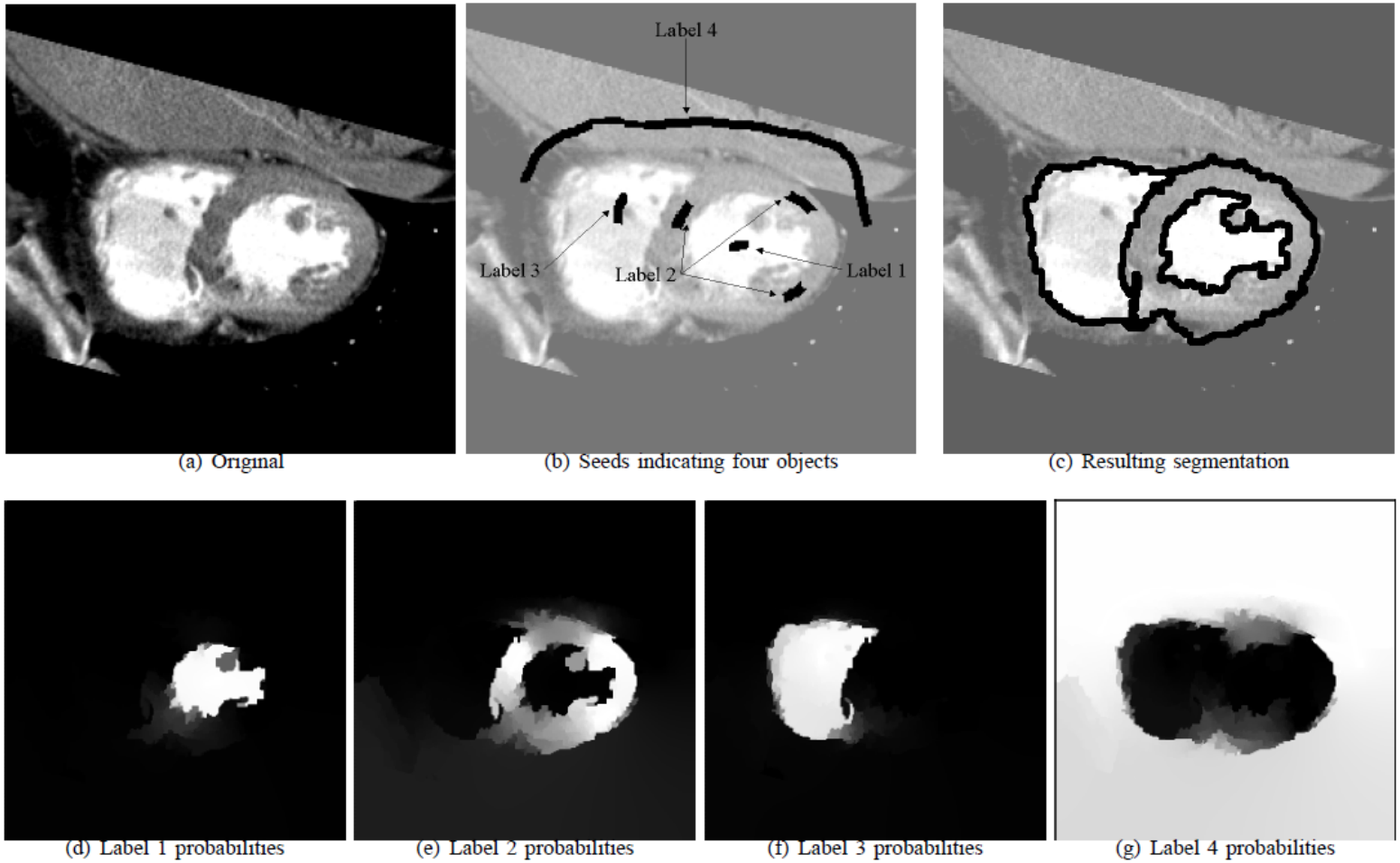


Figure 5: Seeded segmentation with Random walks.

Properties of the Random Walker segmentation method

The segmentations obtained by the Random Walker method satisfies the following nice properties

- Each segment is guaranteed to be connected to a seed point with the same label, i.e., there are no isolated components that do not contain seedpoints. (*Controversial!*)
- The K-tuple of probabilities at each vertex is equal to the weighted average of the K-tuples of the adjacent vertices.
- The solution for the probabilities is unique.
- The expected segmentation for an image of pure noise is equal to that obtained for a flat image. ("Voronoi-like")

Comparison with minimum cost paths

- With seeded segmentation based on minimum cost paths (IFT), the label of each vertex is determined by the minimum cost path to the set of seedpoints.
- With Random walker segmentation, we are considering the cost of *all* possible paths between each vertex and the set of seedpoints.

Robustness to weak boundaries

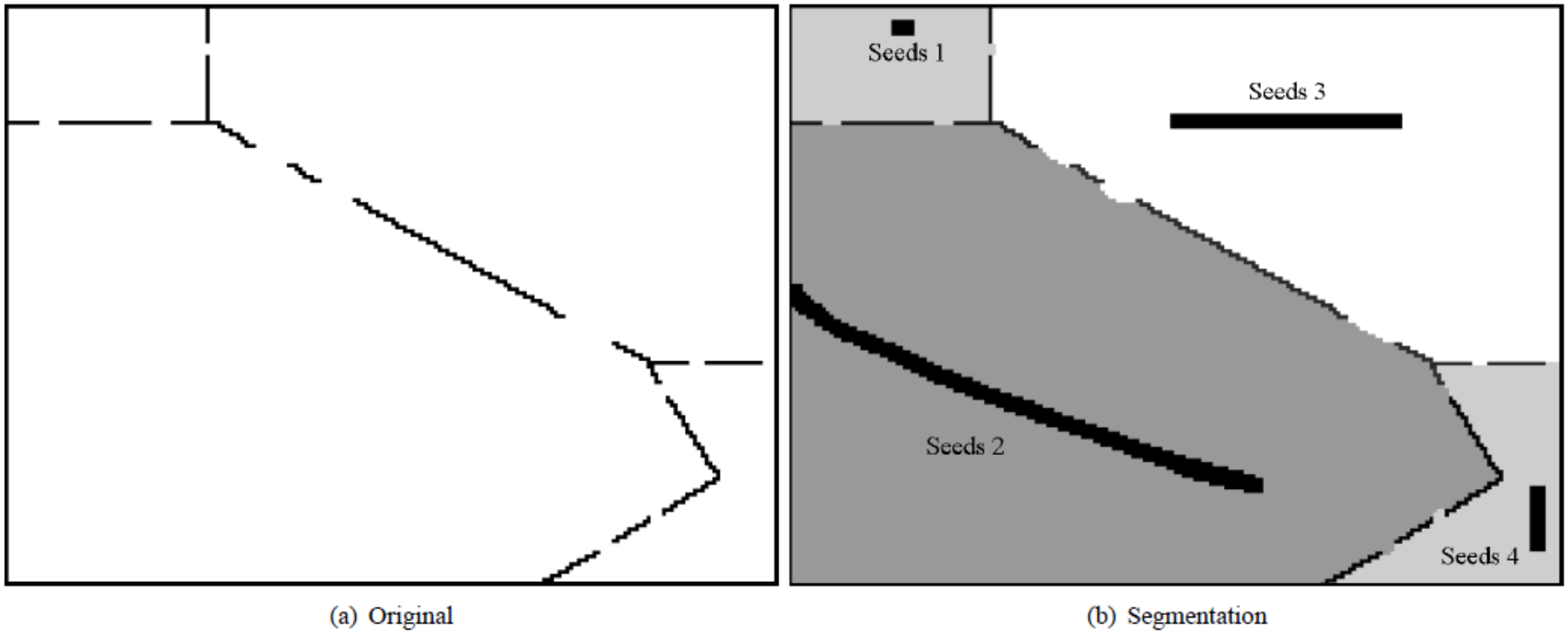


Figure 6: Example of seeded segmentation with Random walks preserving missing boundaries.

Robustness to weak boundaries

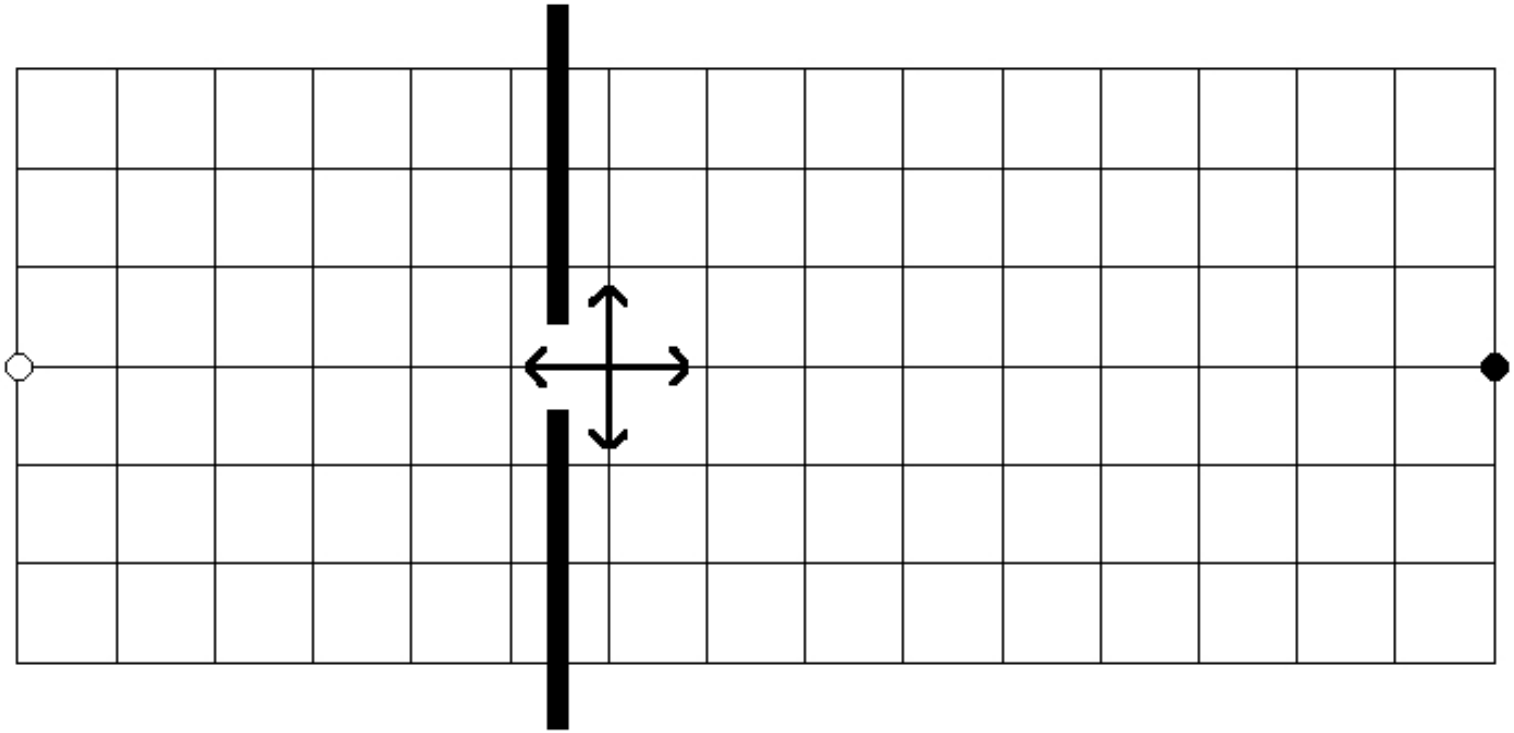
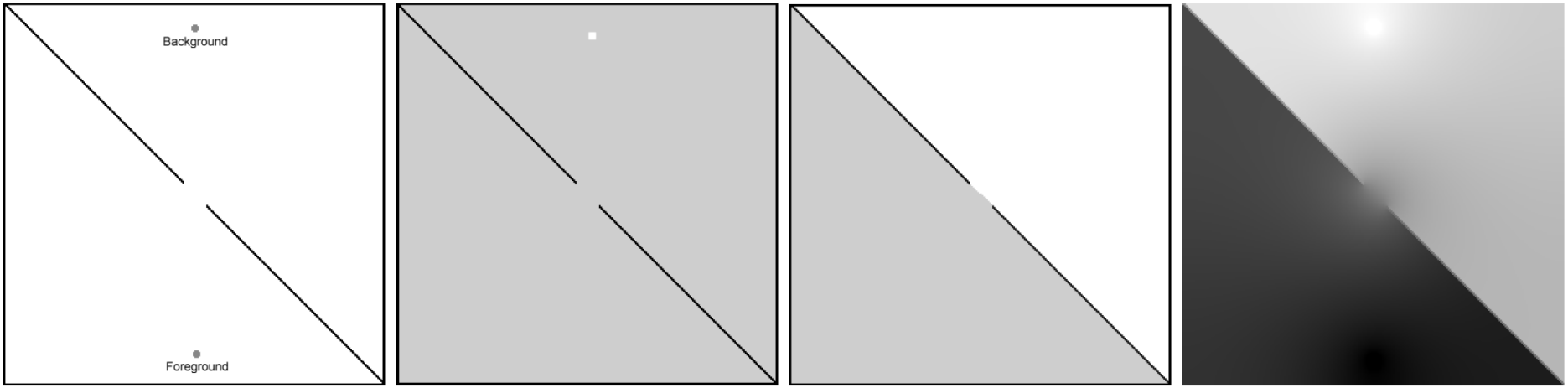


Figure 7: Illustration of why segmentation by Random walks preserves missing boundaries.

Robustness to weak boundaries



(a) Original (b) Graph cuts (c) Random walker (d) Probabilities

Figure 8: Comparison with minimal graph cuts.

Connection with anisotropic diffusion

- The solution of the Dirichlet problem can also be interpreted as the steady-state solution of a diffusion process.
- (Show Matlab example.)

Part 2: Guided interpolation

Guided interpolation: Poisson image processing

- In previous examples, we tried to minimimize gradients subject to boundary constraints.
- Now, we will instead try to make the gradients of the interpolated region match a pre-defined "gradient field".

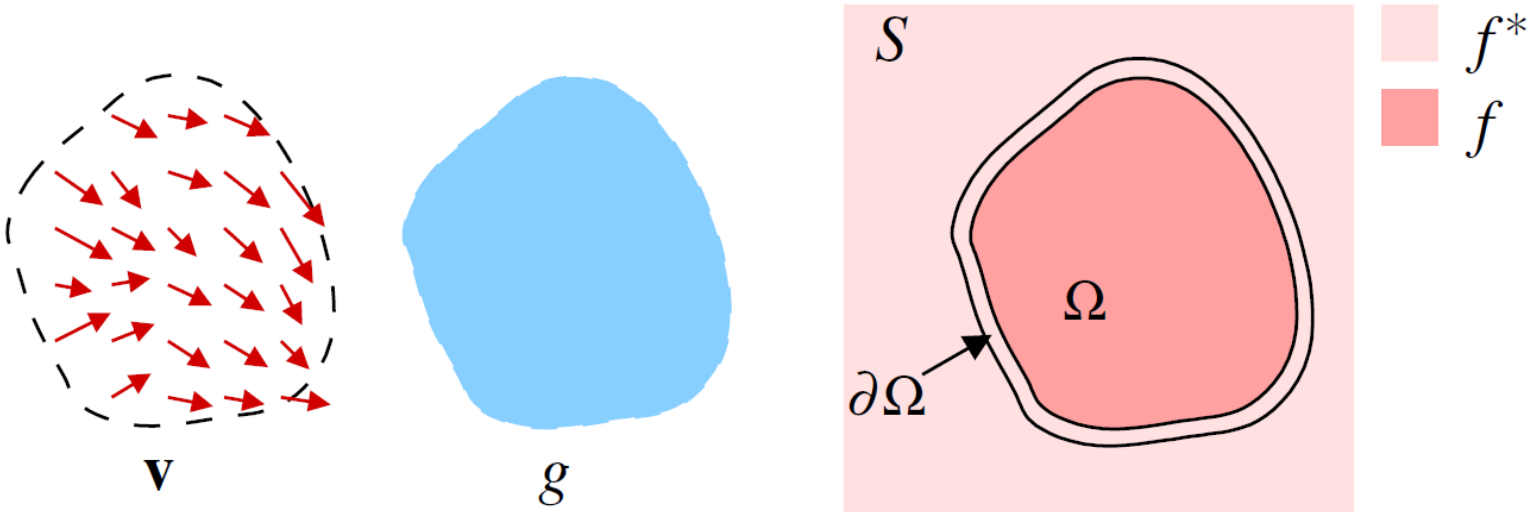


Figure 9: Guided interpolation. Unknown function f interpolates in domain Ω the destination function $f*$ under guidance of vector field $\mathbf{v}$, which may or may not be a gradient field of a source function g.

Continuous formulation

We consider an extended version of the Dirichlet integral:

$$D[u] = \frac{1}{2} \int_{\Omega} |\nabla u - \mathbf{v}|^2 d\Omega \, . \tag{13}$$

for a field u and region Ω. This integral is minimized by solutions of the *Poisson equation* with Dirichlet boundary conditions:

$$\nabla^2 u = div \; \mathbf{v} \, . \tag{14}$$

Discrete formulation

- Just like the Laplace equation, the Poisson equation has a combinatorial equivalent that can be written as a linear system of equations.
- For details, see [6].

Image manipulation in the "gradient domain"

The values of an image are uniquely determined by the gradient vectors (and a constant). This allows to use the following image manipulation pipeline:

- Compute the gradient vector field of (the whole or parts of) the image.
- Manipulate the gradient vectors in some way.
- Solve the Poisson equation to obtain the "inverse" of the gradient field.
- If the vector field is conservative (i.e., it is a "valid" gradient field of some function) then the gradients of the resulting image values will match the vector field exactly. Otherwise, it will still be as close as possible, according to the L_2-norm.

Poisson image editing, examples

Figure 10: By supressing low magnitude gradients in parts of the image, textures and noise can be removed.

Poisson image editing, examples

Figure 11: By selectively replacing parts of the gradient field with that from another image, we can perform seamless cloning of objects between images.

Poisson image editing, examples

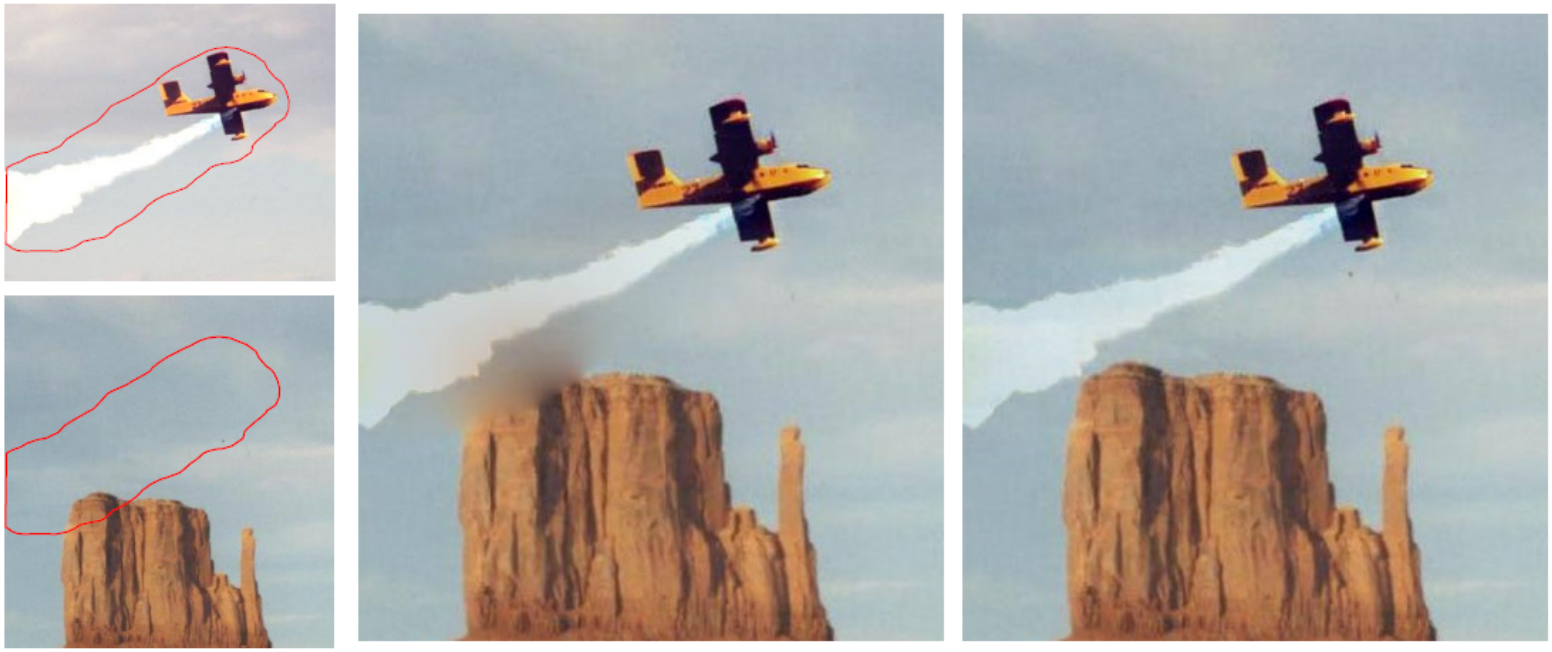


Figure 12: By selectively replacing parts of the gradient field with that from another image, we can perform seamless cloning of objects between images.

Poisson image editing, examples

Figure 13: Fuzzy segmentation obtained by solving the Poisson equation within unkown regions of a "trimap". Image from [7].

Poisson image editing, examples

Figure 14: Fuzzy segmentation obtained by solving the Poisson equation within unkown regions of a "trimap". Image from [7].

Conclusions

- We have introduced a set of tools for interpolating missing data on graph vertices.
  - By solving the (discrete) Laplace equation, we obtain interpolations that are as smooth as possible, according to some (possibly anisotropic) metric.
  - By solving the (discrete) Poisson equation, we obtain interpolation whose gradients match those of a given "guidance" vector field as well as possible.
- In both cases, solutions can be found by solving large, sparse, systems of linear equations.
- Solutions of the Laplace equation can be interpreted in terms of random walks in the graph.
- These tools have many applications in segmentation, image manipulation, etc.

References

[1] L. Grady.

Space-Variant Machine Vision — A Graph Theoretic Approach.

PhD thesis, Boston University, 2004.

[2] Leo Grady.

Random walks for image segmentation.

IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(11):1768–1783, 2006.

[3] Leo Grady.

A lattice-preserving multigrid method for solving the inhomogeneous poisson equations used in image analysis.

In *Computer Vision – ECCV 2008*, volume 5303 of *LNCS*, pages 252–264. Springer, 2008.

[4] Leo Grady and Jonathan R. Polimeni.

Discrete Calculus: Applied Analysis on Graphs for Computational Science.

Springer, 2010.

[5] Leo Grady and Eric Schwartz.

Anisotropic interpolation on graphs: The combinatorial Dirichlet problem.

Technical Report CAS/CNS-TR-03-014, Department of Cognitive and Neural Systems, Boston University, Boston, MA, July 2003.

[6] Patrick Pérez, Michel Gangnet, and Andrew Blake.

Poisson image editing.

ACM Trans. Graph., 22(3):313–318, 2003.

[7] Jian Sun, Jiaya Jia, Chi-Keung Tang, and Heung-Yeung Shum.

Poisson matting.

ACM Transactions on Graphics, 23(3), 2004.