Optimisation variationnelle discrète et applications en vision par ordinateur

Camille Couprie

Université Paris Est
École doctorale MSTIC

Thèse soumise pour obtenir le grade de
Docteur de l'Université Paris Est en Informatique

Camille COUPRIE

Directeur de thèse: Laurent Najman
Co-directeurs: Hugues Talbot et Leo Grady

Graph-based variational optimization and applications in computer vision

Optimisation variationnelle discrète et applications en vision par ordinateur

Soutenue le 10 octobre 2011

Membres du jury:

G. Strang	(Président)
A. Chambolle	(Rapporteur)
J. M. Morel	(Rapporteur)
J. Blanc-Talon	(Examinateur)
L. Grady	(Examinateur)
T. Pock	(Examinateur)
P. Perrot	(Invité)
L. Najman	(Directeur de thèse)
H. Talbot	(Co-Directeur de thèse)

Acknowledgments / Remerciements

I would like to acknowledge, in no particular order, the three persons who supervised me during these three years. I was extremely lucky to work with Leo Grady, Laurent Najman and Hugues Talbot who shared their skills and enthusiasm with me. Leo Grady was my master's degree internship advisor. He gave me the initial motivation and strong scientific basis that stayed with me for the rest of my PhD. I particularly want to thank him for captivating me with many interesting problems. Laurent Najman and Hugues Talbot, in addition to their own vision and enthusiasm for solving exciting problems, provided me with scientific guidance as well as continuing advice regarding the many important choices I had to make before and during my Ph.D. I am also thankful for their willingness to invite me to participate in many interesting projects and events. It has been a real pleasure to work with them.

I enjoyed working with Jean-Christophe Pesquet and Xavier Bresson, and I wish to thank them for the insights they brought to the problems I was studying.

I am also grateful to my referees Antonin Chambolle, Jean-Michel Morel, Thomas Pock, and Gilbert Strang for their time. I'm greatly honored by their participation in my jury.

J'ai bénéficié au long de ma thèse d'un financement de la Délégation Générale pour l'Armement (DGA), obtenu avec l'aide du commandant Patrick Perrot de l'Institut de Recherche Criminelle de la Gendarmerie Nationale, et de Jacques Blanc-Talon, responsable du domaine scientifique "Ingénierie de l'information et robotique" à la DGA, que je remercie également vivement pour leur participation à ma soutenance de thèse.

J'ai beaucoup apprécié la présence de mon compagnon de bureau Laszlo Marak, et plus récemment Juliette Charpentier, avec qui il était très agréable de travailler.

Je remercie également très chaleureusement tous les membres du laboratoire A3SI de l'ESIEE qui contribuent tous à une très bonne ambiance de travail, je citerai parmis eux Gilles Bertrand, Jean Serra, Jean Cousty, Yukiko Kenmochi, Denis Bureau parmi les professeurs, et John Chaussard, Benjamin Raynal, Roland Levillain, Olena Tankyevych, Yohan Thibault, Vincent Bismuth, Imen

Melki parmi les anciens ou actuels joyeux doctorants. L'équipe de l'IMAC, non moins joyeuse, Vincent Nozick, Venceslas Biri, Nadine Dommanget, Anthony Giroud, Adrien Herubel, ainsi qu'une partie de l'équipe de traitement du signal de l'université Paris Est Marne-la-Vallée, avec Nelly Pustelnik, Caroline Chaux, Emilie Chouzenoux, Ania Jezierska, Mireille El Gheche m'ont également accompagné durant cette thèse.

Pour terminer, je tiens à remercier ma famille, mon mari Arnaud et mes parents Michel et Michèle Couprie à plus d'un titre, pour tout leur soutien et précieux conseils.

Abstract

Many computer vision applications such as image filtering, segmentation and stereovision can be formulated as optimization problems. Recently discrete, convex, globally optimal methods have received a lot of attention.

Many graph-based methods suffer from metrication artifacts. Segmented contours are blocky in areas where contour information is lacking. In Chapter 2, we develop a discrete yet isotropic energy minimization formulation for the continuous maximum flow problem that prevents metrication errors. This new convex formulation leads us to a provably globally optimal solution. The interior point method can optimize the problem faster than existing continuous methods. The energy formulation is adapted and extended to multi-label problems in Chapter 3 and shows improvements over existing methods. Fast parallel proximal optimization tools have been tested and adapted for the optimization of this latest problem.

In Chapter 4 we introduce a framework that generalizes several state-of-the-art graph-based segmentation algorithms, namely graph cuts, random walker, shortest paths, and watershed. This generalization allowed us to exhibit a new case, for which we developed a globally optimal optimization method: "Power watershed". Our proposed power watershed algorithm computes a unique global solution to multilabeling problems, and is very fast. In Chapter 5, we further generalize and extend the framework to applications beyond image segmentation, for example image filtering optimizing an ℓ_0 norm energy, stereovision and fast and smooth surface reconstruction from a noisy cloud of 3D points.

Keywords: variational problems, discrete calculus, graphs, regularization, convex optimization, image processing, segmentation, restoration, surface reconstruction, mathematical morphology.

Résumé

De nombreuses applications en vision par ordinateur comme le filtrage, la segmentation d'images, et la stéréovision peuvent être formulées comme des problèmes d'optimisation. Récemment les méthodes discrètes, convexes, globalement optimales ont reçu beaucoup d'attention.

La méthode des "graph cuts", très utilisée en vision par ordinateur est basée sur la résolution d'un problème de flot maximum discret, mais les solutions souffrent d'un effet de blocs, notamment en segmentation d'images. Une nouvelle formulation basée sur le problème continu est introduite au chapitre 2 et permet d'éviter cet effet. La méthode de point intérieur employée permet d'optimiser le problème plus rapidement que les méthodes existantes, et la convergence est garantie. Dans le chapitre 3, la formulation proposée est efficacement étendue à la restauration d'image. Grâce à une approche duale contrainte et a un algorithme proximal parallèle, la méthode permet de restaurer (débruiter, déflouter, fusionner) des images rapidement et préserve un meilleur contraste qu'avec la méthode de variation totale classique.

Le chapitre 4 met en évidence l'existence de liens entre les méthodes de segmentation "graph-cuts", le "random walker", et les plus courts chemins avec un algorithme de segmentation par ligne de partage des eaux (LPE). Ces liens ont inspiré un nouvel algorithme de segmentation multi-labels rapide produisant une ligne de partage des eaux unique, moins sensible aux fuites que la LPE classique. Nous avons nommé cet algorithme "LPE puissance". L'expression de la LPE sous forme d'un problème d'optimisation a ouvert la voie à de nombreuses applications possibles au delà de la segmentation d'images, par exemple au chapitre 5 en filtrage pour l'optimisation d'un problème non convexe, en stéréovision, et en reconstruction rapide de surfaces lisses délimitant des objets à partir de nuages de points bruités.

Mots Clefs: traitement d'images, optimisation convexe, méthodes variationnelles, graphes, segmentation, restauration, vision par ordinateur, morphologie mathématique

Publications

Journals

[1] Camille Couprie, Leo Grady, Laurent Najman, and Hugues Talbot. Power watersheds: a unifying graph based optimization framework. In IEEE Transactions on Pattern Analysis and Machine Intelligence (**IEEE PAMI**), (vol. 33 no. 7), pp. 1384-1399, July 2011.

[2] Camille Couprie, Leo Grady, Hugues Talbot, and Laurent Najman. Combinatorial Continuous Max Flows. In **SIAM** journal on imaging sciences. 2011.

[3] Camille Couprie, Leo Grady, Laurent Najman, Jean-Christophe Pesquet and Hugues Talbot. Constrained TV-based regularization on graphs. Submitted. 2011.

International conferences

[4] Camille Couprie, Hugues Talbot, Jean-Christophe Pesquet, Laurent Najman, and Leo Grady. Dual constrained TV-based regularization. In *Proc. of International Conference on Acoustics, Speech and Signal Processing (***ICASSP***)*, 2011.

[5] Camille Couprie, Xavier Bresson, Laurent Najman, Hugues Talbot, and Leo Grady. Surface reconstruction using power watersheds. In *Proc. of International Symposium on Mathematical Morphology (***ISMM***)*, 2011.

[6] Camille Couprie, Leo Grady, Laurent Najman, and Hugues Talbot. Anisotropic diffusion using power watersheds. In *Proc. of International Conference on Image Processing (***ICIP***)*, pages 4153–4156, 2010.

[7] Camille Couprie, Leo Grady, Laurent Najman, and Hugues Talbot. Power watersheds: A new image segmentation framework extending graph cuts, random walker and optimal spanning forest. In *Proc. of International Conference on Computer Vision (***ICCV***)*, pages 731–738, Sept. 2009.

[8] Camille Couprie, Leo Grady, Laurent Najman, and Hugues Talbot. A new image segmentation framework: Power watersheds. In *Proc. of International Symposium on Mathematical Morphology* (**ISMM**), pages 53–55, 2009.

Contents

Notations

G	:	A graph.
V	:	Set of vertices of a graph.
E	:	Set of edges of a graph.
m	:	Number of edges in a graph.
n	:	Number of vertices in a graph.
A	:	Incidence node-edge matrix of a graph.
g	:	Vector of nodes weights.
w	:	Vector of edge weights.
s	:	Index of a source node.
t	:	Index of a sink node.
c	:	Vector indicator of the sink-source edge in a transport graph.
$\|A\|$	:	Matrix composed of the component-wise absolute values of the elements of A.
$\|A\|_i$	:	The i^{th} row of $\|A\|$: sum operator through neighbors of node i.

Let u, v be real vectors of $\mathbb{R}^N$.

$u \cdot v$	:	Hadamar product (component-wise) of u by v.
u^p	:	Component-wise power operator of u.
$u^\top v$	:	Scalar product of u by v.
$u \cdot / v$	:	Component-wise division of u by v.

Let C be a nonempty closed set, $C \subset \mathbb{R}^N$.

$\imath_C(u)$	:	Indicator function of C.
$\sigma_C(u)$	:	Support function of C.
$P_C u$	:	Projection of u on C.

Chapter 1

Introduction

Computer vision encompasses a large diversity of important and fascinating problems that have revolutionized many endeavors in medicine, industrial vision, visualization, and multimedia. The computer vision community was able to contribute to these applications by formulating their corresponding problems in a solvable fashion. Particularly, energy minimization methods have been very successful in this regard.

Problems statement

Distinguishing objects from their background may sometimes appear like an easy and intuitive task for humans, but it is in fact complex and difficult to reproduce on a computer. Effectively removing noise or reducing blur on images, although better defined, also remains an open problem. Example of such inverse problems are shown in Figure 1.1.

With respect to image segmentation, early optimization methods were formulated in terms of active contours and surfaces [133] and then later level sets [194]. These formulations were used to optimize energies of varied sophistication (e.g., using regional, texture, motion or contour terms [175]) but generally converged to a local minimum, generating results that were sensitive to initial conditions and noise levels. Consequently, focus has shifted on energy formulations (and optimization algorithms) for which a global optimum can be found.

Many energy-based problems in vision involve the resolution of Partial Differential Equations (PDEs). Since the development of computer science, there has been a need to translate the classical mathematical models into a computable form to obtain numerical solution. While classical solutions to PDEs involve either finite difference or finite elements schemes, in imaging data is available only in discrete form, typically as a regular graph. Designing formulations of PDE problems on graphs [119] has numerous advantages over previous schemes, such as removing the necessity to discretize the data again in a different form. In this way, it is possible to reuse existing efficient combinatorial optimization tools to minimize energies

(network flows, shortest paths, minimum spanning trees) and to develop new ones. Finally, this approach generalizes these discrete algorithms to arbitrary graph, which is useful for modeling other kinds of data than planar images (e.g. triangulated or parametric surfaces, unevenly sampled images, etc).

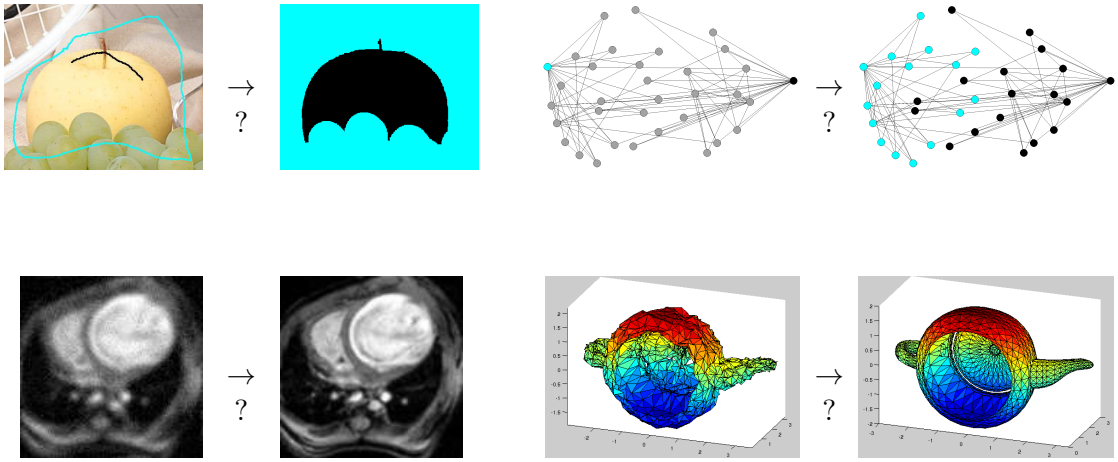


Figure 1.1: Clustering and restoration problems, in images and arbitrary graphs.

This thesis brings several new formulations of graph related problems, and presents results obtained in image segmentation and restoration problems among others. The generality of the proposed approaches makes it possible to use the developed techniques to deal with a larger class of data as shown in Figure 1.1, with for example mesh filtering and data classification. We begin our exposition by introducing our graph notations as well as a key operator used in this manuscript.

1.1 Discrete calculus

Discrete calculus [119, 124] has been used in recent years to produce a combinatorial reformulation of continuous problems onto a graph in such a manner that the solution behaves analogously to the continuous formulation (e.g., [89, 113]).

Discrete calculus has to be differentiated from traditional finite elements *discretization*. Such discretization aims to produce approximate solution to problems defined in the continuum. The approximation becomes better as the discretization becomes finer and finer. In contrast, discrete calculus does not refer to the continuum. For example, as noted in [119], social networks represent data that do not rely on a continuous domain. They are not a sampling of an underlying continuum. However, the tools of discrete calculus may be applied

as we shall see in Chapter 2.

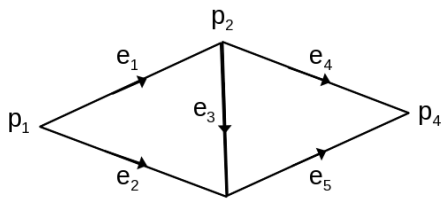
We introduce here the graph notations that are used in this manuscript.

A graph consists of a pair $G = (V, E)$ with vertices $v \in V$ and edges $e \in E \subseteq V \times V$. Let n represent the number of nodes, i.e. $n = |V|$, and m the number of edges of G, i.e. $m = |E|$. An edge, e, spanning two vertices, v_i and v_j, is denoted by e_{ij}. In this thesis, we deal with weighted graphs that include weights on both edges and nodes. An edge weight is a value assigned to each edge e_{ij}, and is denoted by w_{ij}. We assume that $w_{ij} \in \mathbb{R}_+^*$ and use w to denote the vector of $\mathbb{R}^m$ containing the w_{ij} for every edge e_{ij} of G. In addition to edge weights, we may also assign weights to nodes. The weight of node v_i is denoted by g_i. In this work, we also assume that $g_i \in \mathbb{R}_+^*$. We use g to denote the vector of $\mathbb{R}^n$ containing the g_i for every node v_i of G.

The incidence matrix of a graph is a key operator for defining combinatorial formulations of variational problems. Specifically, the incidence matrix $A \in \mathbb{R}^{n \times m}$ is known to define the discrete calculus analogue of the gradient, while $A^\top$ is known to define the discrete calculus analogue of the divergence (see [119] and the references therein). The incidence matrix maps functions on nodes (a scalar field) to functions on edges (a vector field) and may be defined as

$$A_{e_{ij}v_k} = \begin{cases} -1 & \text{if } i = k, \\ +1 & \text{if } j = k, \\ 0 & \text{otherwise,} \end{cases} \tag{1.1}$$

for every vertex v_k and edge e_{ij}. An example of such matrix is given in Fig. 1.2.



$$A = \begin{array}{c|cccc} & p_1 & p_2 & p_3 & p_4 \\ \hline e_1 & -1 & 1 & 0 & 0 \\ e_2 & -1 & 0 & 1 & 0 \\ e_3 & 0 & -1 & 1 & 0 \\ e_4 & 0 & -1 & 0 & 1 \\ e_5 & 0 & 0 & -1 & 1 \end{array}$$

Figure 1.2: A graph and its incidence matrix $A \in \mathbb{R}^{m \times n}$.

As explained in [119], several important theorems such as the fundamental theorem of calculus or the Helmholtz decomposition hold in this discrete framework.

In the next section, we briefly introduce some classical regularization methods used as references, as well as some of the optimization methods employed in this thesis.

1.2 Regularization methods

Numerous problems in signal processing and computer vision involve the optimization of an energy function composed of a regularization term and a data fidelity term. For example, image denoising problems may be solved by considering that the optimal recovered image is somehow regularized, that is to say that the variations (the local differences) of intensity are limited. For image segmentation problems, the sum of variations between neighboring labels may be penalized. In stereovision variational approaches, estimation of depth maps also assume a local consistency for the depth, and limit the sum of variations in order to obtain a piecewise smooth estimation.

In all these cases, the problem may be expressed as follows. A labeling x is estimated, x being the solution to the minimization of its total variations added to a term enforcing data fidelity.

$$\boxed{\min_x \int_\Omega \underbrace{||\nabla x(z)||\ \ \mathrm{d}z}_{\text{Regularization}} + \underbrace{\mathcal{D}(x)}_{\text{Data fidelity}}} \tag{1.2}$$

A simple example of data fidelity term in an image denoising context could be

$$\mathcal{D}(x) = \int_\Omega (x(z) - f(z))^2 \ \mathrm{d}z,$$

appropriate in the case where f is an image corrupted with Gaussian noise.

Remaining questions include, in particular "How do we translate the norm of the gradient of x numerically?", and "How to optimize the resulting the energy efficiently?". Also the assumption of local smoothness may be relaxed by considering of non-local gradient, for instance by extending graphs with long distance edges [47]. Before presenting the different regularization terms explored in this thesis, we shortly review two classical and efficient models for this matter.

1.2.1 Total variation

The Total Variation problem (TV) was introduced originally in computer vision by Shulman and Hervé [80] as a regularizing criterion for optical flow computation and later Rudin, Osher and Fatemi [189] as a regularizing criterion for image denoising. It has been shown to be quite efficient for smoothing images without affecting contours so much. Moreover, a major advantage of TV is that it is a convex problem, making it possible to find a global minimizer.

When applied to a 2D discrete image $(x_{i,j})$, $1 \leq i, j \leq N$, the total variation minimization

(a) Original image (b) Noisy image (c) Non-local TV result

Figure 1.3: Image denoising example by TV minimization on a non-local graph. Figure from X. Bresson [44].

problem becomes

$$\min_x \sum_{1\leq i,j\leq N} \left((x_{i+1,j}-x_{i,j})^2+(x_{i,j+1}-x_{i,j})^2\right)^{\frac{1}{2}} + \mathcal{D}(x). \tag{1.3}$$

using a finite-difference discretization popularized in [50]. We note that other discrete formulation exist, for example a recent upwind variant of this model preserving more efficiently isotropic features was introduced in [53]. The Total variation problem has been expressed in weighted and non-local graphs in [29, 103], leading to better penalization of the gradient.

A number of convex optimization techniques suitable for solving this problem have been employed for several decades. The most recent and efficient approaches are compared in [51].

Among the most efficient methods, one can cite Nesterov's algorithm [169], Split-Bregman / Douglas-Rachford methods [103, 108], and Chambolle-Pock's Primal-dual algorithm [54]. Most methods minimizing TV focus on image filtering as an application, and even if those methods are remarkably fast in denoising applications, we observe in Chapters 2 and 3 that segmentation problems require more iterations for those algorithms to converge.

1.2.2 Max flow - Min cut

The max-flow/min-cut problem on a graph is a classical problem in graph theory, for which the earliest solution algorithm goes back to Ford and Fulkerson [97]. Initial methods for global optimization of the boundary length of a region formulated the energy on a graph and relied on max-flow/min-cut methods for solution [32]. In the context of image segmentation, the seeded max-flow / min-cut model is also know as "Graph cuts". Graph cuts algorithms provide a mechanism for the discrete optimization of an energy functional [141], which have been used in a variety of applications such as image segmentation, stereo, image stitching

and texture synthesis (See Fig. 1.4). More specifically, the labeling x produced by graph cuts algorithms is an optimal (not necessarily unique) solution to

$$\min_{x\in\{0,1\}^n} \sum_{e_{i,j}\in E} w_{ij}|x_i - x_j| + \mathcal{D}(x). \tag{1.4}$$

This energy may be optimized using the non-polynomial but experimentally fast method of [33]. As explained in [141], more general – submodular – energies than (1.4) may be optimized using Graph cuts. Consequently, the graph cuts technique has been adapted to multilabel problems using various graph constructions. One notable construction is that of Ishikawa [129] presenting the multilabel problem as a segmentation problem.

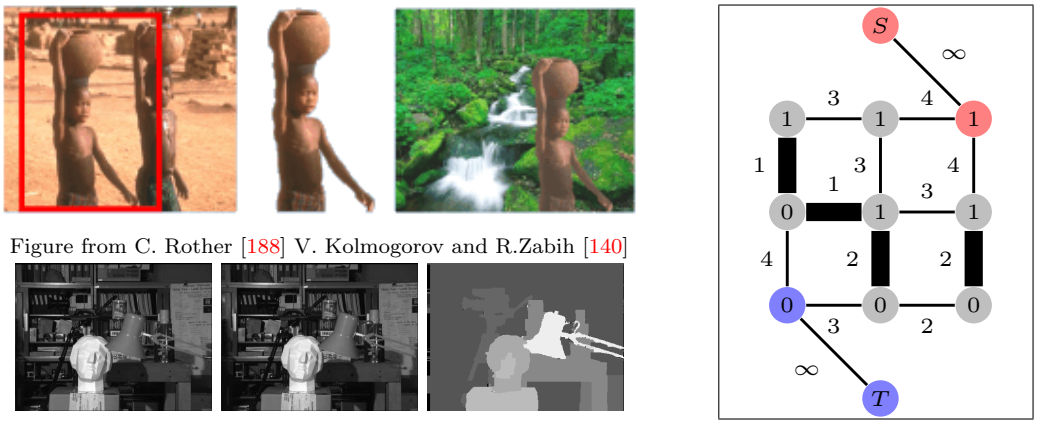


Figure from C. Rother [188] V. Kolmogorov and R.Zabih [140]

Figure 1.4: A transport graph with its resulting max-flow/min-cut solution, and illustration of applications in image segmentation and stereovision.
The bold edges represent saturated edges, and constitute a minimal cut.

1.2.3 Combinatorial Dirichlet Problem

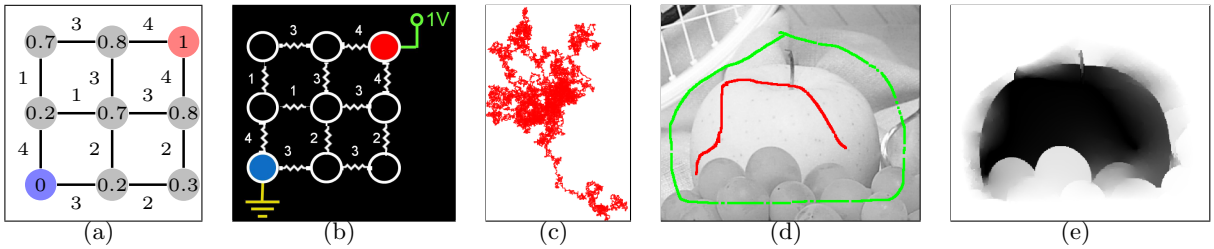
The Laplace equation arises in many physical situations : heat propagation, fluid dynamic, and electronics among others. The Dirichlet problem aims at finding a solution satisfying the Laplace equation subject to some boundary constraints.

In computer vision and image processing problems, solving the Dirichlet problem provide effective solution to labeling problems, given some markers. In this context, the combinatorial Dirichlet problem is written

$$\min_{x\in\mathbb{R}^n} \sum_{e_{i,j}\in E} w_{ij}(x_i - x_j)^2 + \mathcal{D}(x). \tag{1.5}$$

In image processing, this problem has been applied for example to interpolation [115], inpainting [14], image filtering [24], and seeded image segmentation [110]. Figure 1.5 shows

an application of the "random walker" [110], an adaptation of Dirichlet problem to image segmentation.



(a) (b) (c) (d) (e)

Figure 1.5: Analogies and illustration of results obtained optimizing the combinatorial Dirichlet problem
(a) A graph with two constraints on x: one value set to '0', and another to '1'. Given those boundary constraints, the labeling x optimizing (1.5) is shown on the nodes. (b) Analogy with an electrical network resistance: electric voltage of all nodes are given by the solution to (1.5) in (a). (c) Analogy with random walks: in (a), each node value x corresponds to the probability for a random walker to reach the node labeled '1' before the node labeled '0'. (d) Example of constraints (foreground and background seeds) for seeded image segmentation. (e) Obtained result x optimizing (1.5).

A summary of the different graph-based regularization terms previously introduced is given in Table 6.1, presenting as well the different novel regularization terms introduced in this thesis. Before the exposition of our contributions, we recall in the next section some combinatorial optimization methods useful in Chapter 4 and 5, as well as three algorithms for convex optimization that are used in Chapters 2 and 3.

1.3 Some optimization tools

1.3.1 Combinatorial methods

A good overview of combinatorial optimization techniques is presented in [174]. We focus our exposition in this section on the combinatorial methods futher studied in Chapter 4.

1.3.1.1 Maximum Flow/Minimum cut algorithms

A transport graph G is as a graph with exactly one source s, with no entering edge, and one sink t, with no exiting edge.

A flow in a transport graph is a function associating a value to every edge according to some rules. The value of the flow in each edge must be less than the capacity (weight) of the edge, and the sum of the flow entering in each node – except s and t – must be equal to the sum of flow exiting the node. The max-flow problem consists of maximizing the amount of

flow sent from the source toward the sink. This process creates a bottleneck for some edges. If the flow of an edge is equal to its capacity, we say that the edge is saturated.

A cut is a partition of the nodes into two sets S and T, such that s is in S and t is in T. The capacity of a cut is the sum of the weights of all the edges crossing the cut, from the region S to the region T. Ford and Fulkerson [97] proved that the maximum flow is equals to the capacity of a minimal cut. As explained in Section 1.2.2, this result has been employed in various applications. Figure 1.4 shows image segmentation and stereovision results obtained using max-flow algorithms.

Most max-flow algorithms can be sorted in two categories, augmenting path [87, 97] and push-relabel methods [105, 185]. Also min-cut algorithms without computing a maximum flow exist, a well known one being the Stoer-Wagner algorithm [201]. Its complexity of $O(nm+n^2\log(m))$ is one of the most effective for general purpose min-cut/max-flow methods, but in practice the runtime on image data is not the fastest.

The regular lattice structure of images was exploited in the dedicated max-flow algorithm of [35], resulting in faster runtimes.

1.3.1.2 Shortest Path methods

The shortest path problem consists of finding, from two nodes of a graph, a path of minimal cost (such that the sum of weights of component edges is minimized) in the graph. When the graph does not contain negative weights, Dijkstra's algorithm [85] is very efficient. When implemented using Fibonacci heaps, its complexity is $O(m+n\log(n))$. First introduced as a byproduct of reinitializing level set energies, the fast marching algorithm [194] has been used for shortest path computation [60] over sampled continuous domains.

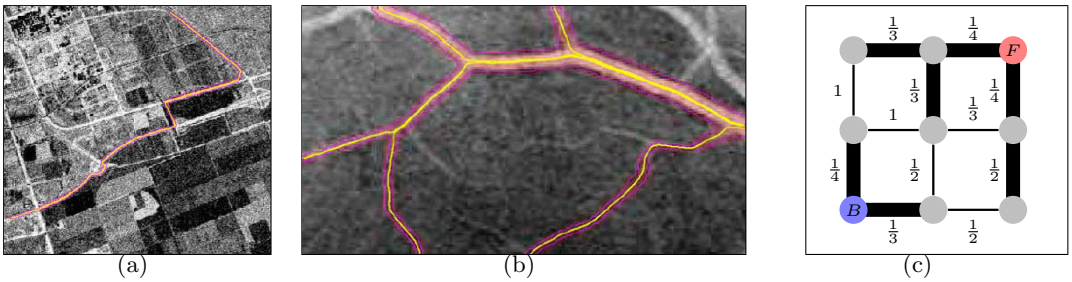


Figure 1.6: Example of shortest path computations for image segmentation of thin, tubular objects.
(a,b) Figure from F. Benmansour [17], (c) Example of shortest path forest for general purpose image segmentation.

Shortest path methods are popular in image processing and particularly for thin object segmentation. The "Live wire", as known as "Intelligent scissors" [160] consists for example

of computing shortest paths between several points placed on the contour of objects to be extracted. Improvements have been proposed to limit the user interaction, and to define better weighting strategies [178]. Other strategies for image segmentation using shortest path based algorithms include the construction of shortest path forests. Given foreground and background seeds, each pixel is assigned to the foreground label if there is a shorter path from that pixel to a foreground seed than to any background seed. This approach was recently popularized by Bai and Sapiro [16], but variants of this idea have appeared in other sources [6, 79, 91].

1.3.1.3 Maximum Spanning Trees and Forests

A spanning tree of a graph is a tree (a connected graph without cycles) connecting all the nodes of the graph. The weight of a spanning tree is defined by the sum of edge weights composing the tree. There exists several greedy algorithms minimizing or maximizing this weight, called respectively minimum or maximum spanning tree algorithms. Borůvka [28] was the first to propose an algorithm for this problem. The Prim [182], and Kruskal [144] algorithms are the most commonly used. When using a union-find data structure [207] for cycles detection, the complexity of maximum spanning tree algorithms is quasi-linear [57].

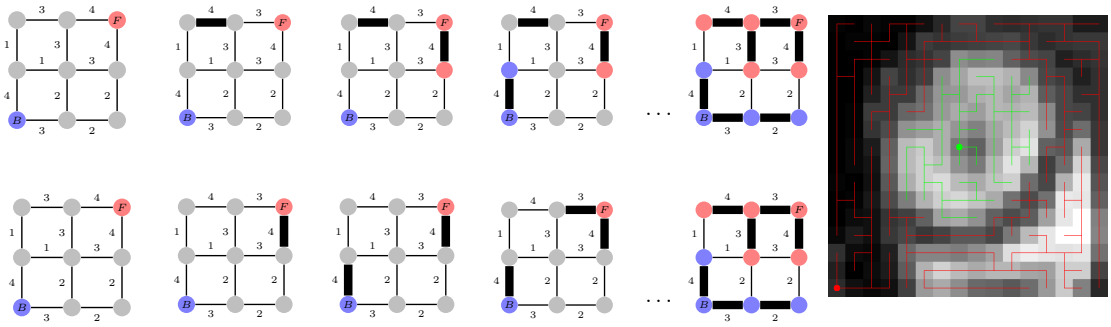


Figure 1.7: Illustration of two Maximum Spanning Forest (MSF) algorithms behavior, and segmentation result on a real image.

Top row : Different steps of Kruskal algorithm for MSF computation. Edges are successively added to the forest (bold edges) by decreasing weight order, without creating cycles, and without merging trees of different labels. Bottom row : Some steps of Prim algorithm for MSF computation. In Prim algorithm, the next edge to be added to the forest is chosen between the edges connected to a labeled node.

In clustering applications, at set of several maximum/minimum spanning trees may be computed, where different trees correspond to the different defined labels. The resulting set of trees is called a Maximum (or resp. Minimum) Spanning Forest. An illustration of Prim and Kruskal algorithms behavior for the construction of Maximum Spanning Forest (MSF) is

given on Figure 1.7 in the context of image segmentation. The segmentation is given by the MSF cut, defined by the set of edges that separates different trees. The first appearance of such forest in image processing dates from 1986 with the work of Morris *et al.* [173]. It was later introduced by Meyer in a morphological context in [156]. Different criteria for regions merging appear in the literature, as for example in the widely used algorithm of Felzenszwalb *et al.* [95]. If the makers are located on the minima of the weight function, the cut obtained by minimum spanning forest computation was shown by Cousty *et al.* to be a watershed cut [77].

We shall see in Chapter 4 that these greedy procedures, which are very fast, may be used for optimizing meaningful and useful energies. The following section introduces a few convex optimization methods employed in Chapters 2 and 3.

1.3.2 Interior Point Method

Interior point methods have been successfully applied to large size problems, and in particular in image processing [22, 55, 137].

As detailed in [31], interior point methods solve problems of the form

$$\begin{aligned} &\min_x \ f_0(x), \\ \text{s. t.} \quad &f_i(x) \leq 0 \quad i = 1, ..., M, \\ &Hx = b, \end{aligned} \tag{1.6}$$

where x is a vector $\in \mathbb{R}^N$, H a matrix of $\mathbb{R}^{P \times N}$ such that $\text{rank}(H) = P < N$, b a vector $\in \mathbb{R}^P$, and $f_0, ..., f_M$ are convex, twice differentiable functions of $\mathbb{R}^N \to \mathbb{R}$.

In Chapter 2 we shall be interested in the dual solution of the considered optimization problem, this is why we focus our exposition on a primal-dual interior point method.

The dual of Problem (1.6) can be computed by mean of a Lagrangian function. The basic idea is to take the constraints into account by augmenting the objective function with a weighted sum of the constraint functions.

The Lagrangian associated with the problem (1.6) is defined by

$$L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{M} \lambda_i f_i(x) + \sum_{i=1}^{M} \nu_i (H_i x - b_i),$$

with $\lambda \in \mathbb{R}^M$ and $\nu \in \mathbb{R}^M$ two Lagrange multipliers. Then the dual objective function is defined by

$$h(\lambda, \nu) = \inf_x L(x, \lambda, \nu).$$

Taking the best lower bound of the Lagrangian function leads to the dual problem

$$\begin{aligned} \max_{\lambda,\nu} \quad & h(\lambda,\nu), \\ \text{s. t.} \quad & \lambda \geq 0. \end{aligned}$$

The primal-dual interior point (PDIP) algorithm computes iteratively the primal x and dual λ, ν variables so that the Karush-Kuhn-Tucker (KKT) optimality conditions are satisfied. The KKT conditions guarantee that if the functions f_i are convex, and x, λ and ν are satisfying

$$\begin{aligned} f_i(x) &\leq 0, \qquad i = 1, ..., M \\ Hx &= b \\ \lambda_i &\geq 0 \qquad i = 1, ..., M \\ \lambda_i f_i(x) &= 0 \qquad i = 1, ..., M \\ \nabla f_0(x) + \sum_{i=1}^{M} \lambda_i \nabla f_i(x) + H^\top \nu &= 0, \end{aligned} \tag{1.7}$$

then x and (λ, ν) are primal-dual optimal.

The PDIP algorithm solves the problem (1.6) by applying the Newton method to a sequence of a slightly modified version of KKT conditions. We define the vector $f(x)$ of $\mathbb{R}^M$ as $f(x) = [f_1(x), ..., f_M(x)]$.

Algorithm 1: Primal-dual interior point algorithm

Input x that satisfies $f_1(x) < 0, ..., f_M(x) < 0, \lambda > 0, \mu > 1, \epsilon > 0$
Output x such that $f_0(x)$ is minimized under the constraints of (1.6)

Repeat
1. Compute the surrogate duality gap $\widehat{\eta} = -f(x)^\top \lambda$, and set $t = \mu M/\nu$.
2. Compute the primal-dual search direction Δ_y such that $M\Delta_y = r$.
3. Determine a step length $s > 0$ and set $y = y + s\Delta_y$. $(y = [x, \lambda, \nu]^\top)$

Until $||r_p||_2 \leq \epsilon, ||r_d||_2 \leq \epsilon,$ and $\widehat{\eta} \leq \epsilon$.

The second step of the PDIP algorithm is called the Newton step and consists of solving

the $M\Delta_y = r$ system

$$\begin{bmatrix} \nabla^2 f_0(x) + \sum_{i=1}^N \lambda_i \nabla^2 f_i(x) & Df(x)^\top & H^\top \\ -\mathrm{diag}(\lambda) Df(x) & -\mathrm{diag}(f(x)) & 0 \\ H & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta_x \\ \Delta_\lambda \\ \Delta_\nu \end{bmatrix} = - \begin{bmatrix} r_d = \nabla f_0(x) + Df(x)^\top \lambda + H^\top \nu \\ r_c = -\mathrm{diag}(\lambda) f(x) - (1/t)\mathbb{1} \\ r_p = Hx - b \end{bmatrix} \tag{1.8}$$

with r_d, r_c, and r_p the dual, central, and primal residuals, and $Df(x) = \begin{bmatrix} \nabla f_1(x)^\top \\ \vdots \\ \nabla f_M(x)^\top \end{bmatrix}$.

Although the complexity of the given primal-dual interior point method is not studied in [31], the authors observe that the growth of the number of iterations in function of the problem size is approximately logarithmic for a linear programming problem. We note that for solving the nonlinear problem addressed in Chapter 2, the number of iterations with respect to the problem size is observed to be linear.

1.3.3 Some proximal algorithms

Proximal methods provide efficient solutions to convex optimization problems where the objective function can be split into a sum of convex functions. This body of methods are applicable to large size problems as well as to non smooth or non-finite functions.

Let us denote $\Gamma_0(\mathbb{R}^N)$ the class of convex functions f of $\mathbb{R}^N$ to $]-\infty, +\infty]$ that are proper (such that $\mathrm{dom}\, f = \left\{ z \in \mathbb{R}^N \;\middle|\; f(z) < +\infty \right\} \neq \varnothing$) and lower semi-continuous.

Specifically, proximal methods may be employed to solve problems of the form

$$\min_{x \in \mathbb{R}^N} \sum_{i=1}^M f_i(L_i x) \tag{1.9}$$

given M linear operators $L_i \in \mathbb{R}^{Q_i \times N}$, $i \in \{1, \ldots, M\}$, where $M, (\forall i \in \{1, \ldots, M\}) Q_i$, and N are natural positive integers and $f_i \in \Gamma_0(\mathbb{R}^{Q_i})$.

The proximal operator of a function is a natural extension of the notion of projection onto a convex set, and is defined as follows.

Definition 1.3.1. *For every $u \in \mathbb{R}^N$, the function $v \mapsto \varphi(v) + \|u - v\|^2/2$, where $\varphi \in \Gamma_0(\mathbb{R}^N)$, achieves its minimum at a* unique point *denoted by* $\mathrm{prox}_\varphi u$*, i.e.,*

$$(\forall u \in \mathbb{R}^N), \qquad \mathrm{prox}_\varphi u = \arg\min_{v \in \mathbb{R}^N} \varphi(v) + \frac{1}{2} \|u - v\|^2 . \tag{1.10}$$

For example, if $\varphi = \iota_C$, $\mathrm{prox}_{\iota_C} = \mathsf{P}_C$, where ι_C denotes the indicator function over a closed convex set C and P_C denotes the projection onto C.

More formally, given $x \in \mathbb{R}^N$, and a nonempty closed convex set $C \subset \mathbb{R}^N$,

$$\imath_C \colon x \mapsto \begin{cases} 0 & \text{if } x \in C, \\ +\infty & \text{otherwise,} \end{cases} \tag{1.11}$$

and the projection $\mathsf{P}_C x$ of $x \in \mathbb{R}^N$ onto C is the solution y to

$$\min_{y \in \mathbb{R}^N} \imath_C(y) + \frac{1}{2}\|x - y\|^2. \tag{1.12}$$

Among classical existing proximal methods, one can cite the augmented Lagrangian methods, Simultaneous Direction Method of Multipliers, and Douglas Rachford algorithms [152] among others. More recently parallel proximal methods have been introduced in order to more easily take advantage of parallel architectures [46, 66].

1.3.3.1 Parallel Proximal Algorithm

Algorithm 2: Parallel ProXimal Algorithm (PPXA) optimizing (1.13)

$\gamma \in \,]0, +\infty[\,, \lambda \in]0, 2[$(relaxation parameter)
$(\omega_1, \dots, \omega_M) \in]0, 1[^M, \sum_{i=1}^M \omega_i = 1, k = 1$
$x_1 = \sum_{i=1}^M \omega_i t_{i,1}$
Repeat until convergence
  For (in parallel) $i = 1, \dots, M$
    $p_{i,k} = \operatorname{prox}_{\frac{\gamma}{\omega_i} f_i} t_{i,k}$
  $c_k = \sum_{i=1}^M \omega_i p_{i,k}$
  For (in parallel) $i = 1, \dots, M$
    $t_{i,k+1} = t_{i,k} + \lambda\big(2c_k - x_k - p_{i,k}\big)$
  $x_{k+1} = x_k + \lambda(c_k - x_k)$
  $k = k + 1$

The Parallel ProXimal Algorithm, also called PPXA, has recently been introduced by Combettes and Pesquet [66]. It computes a sequence of $(x_k)_{k \in \mathbb{N}_*}$ converging toward the minimizer of

$$\min_{x \in \mathbb{R}^N} \sum_{i=1}^M f_i(x) \tag{1.13}$$

for $f_i \in \Gamma_0(\mathbb{R}^{M_i})$, $i \in \{1, \dots, M\}$.

The algorithm, shown in Algorithm 2 computes in parallel the proximal operators of the functions f_i of (1.13), merges them, and updates the iterates also in parallel. Examples of

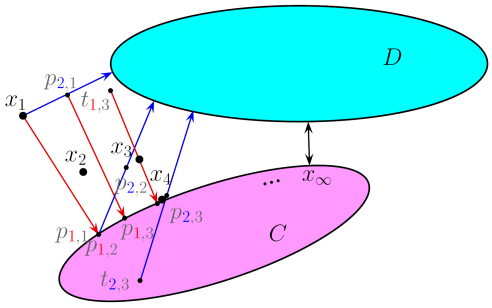
applications, and comparison with existing proximal methods can be found in [67, 183]. A generalization of the algorithm optimizing (1.9), and also including the simultaneous direction of multipliers method is present in [177]. A simple example illustrating the behavior of PPXA is given in Figure 1.8.

Example: find a point x_∞ in set C at the closest distance from D.

$$\min_{x\in\mathbb{R}^N} \underbrace{\iota_C(x)}_{f_1(x)} + \underbrace{\min_{v\in\mathbb{R}^N} \iota_D(v) + \frac{1}{2}||x-v||^2}_{f_2(x)}$$

$\lambda = \gamma = k = 1, \omega_1 = \omega_2 = \frac{1}{2}, t_{1,1} = t_{2,1} = x_1.$

$$\begin{aligned}
p_{1,k} &= \mathsf{P}_C t_{1,k} \\
p_{2,k} &= \tfrac{1}{2}\left(t_{2,k} + \mathsf{P}_D t_{2,k}\right) \\
c_k &= \tfrac{1}{2}(p_{1,k} + p_{2,k}) \\
t_{1,k+1} &= t_{1,k} + p_{2,k} - x_k \\
t_{2,k+1} &= t_{2,k} + p_{1,k} - x_k \\
x_{k+1} &= c_k
\end{aligned}$$



$\text{prox}_{f_1} t_1 = \mathsf{P}_C t_1$

$\text{prox}_{f_2} t_2 = \frac{1}{2}(t_2 + \mathsf{P}_D t_2)$

Figure 1.8: Figure illustrating the behavior of the Parallel Proximal Algorithm for minimizing a sum of two convex functions.

1.3.3.2 M+SFBF algorithm

The "Monotone + Skew Forward Backward Forward" (M+SFBF) algorithm of Briceño and Combettes [46][Prop. 4.4] may be employed to perform the joint optimization of the two following problems

$$\min_{x\in\mathbb{R}^N} \sum_{i=1}^{M} \omega_i f_i(L_i x), \tag{1.14}$$

given M with $i \in \{1,\dots,M\}$ linear operators $L_i \in \mathbb{R}^{Q_i\times N}$, $f_i \in \Gamma_0(\mathbb{R}^{Q_i})$, $\omega_i \in]0,1]$ such that $\sum_{i=1}^{M}\omega_i = 1$, and

$$\min_{\substack{v_1\in\mathbb{R}^{Q_1},\dots,v_M\in\mathbb{R}^{Q_M} \\ \sum i=0^M \omega_i L_i^* v_i = 0}} \sum_{i=1}^{M} \omega_i f_i^*(v_i). \tag{1.15}$$

where for all $i \in \{1,\dots,M\}$, f_i^* denotes the conjugate function of f_i, whose definition is recalled below.

Definition 1.3.2. *Let $f : \mathbb{R}^N \to \mathbb{R}$. The conjugate function $f^* : \mathbb{R}^N \to \mathbb{R}$ of f is defined as*

$$f^*(x) = \sup_{y \in \operatorname{dom} f} (x^\top y - f(y)).$$

Algorithm 3: M+S Forward Backward Forward algorithm

Initialization
- For every $i \in \{1, \dots, M\}$, $\omega_i \in]0, 1]$ such that $\sum_{i=1}^M \omega_i = 1$
- For every $i \in \{1, \dots, M\}$, $x_i \in \mathbb{R}^N, v_i \in \mathbb{R}^{Q_i}$
- $\beta = \max_{1 \le i \le M} \|L_i\|$, let $\epsilon \in]0, 1/(\beta + 1)[$, let in $[\epsilon, (1 - \epsilon)/\beta]$

Repeat until convergence
- $x = \sum_{i=1}^M \omega_i x_i$
- For $i = 1, \dots, M$
  - $y_{1,i} = x_i - \gamma L_i^* v_i$
  - $y_{2,i} = v_i + \gamma L_i x_i$
- $p_1 = \sum_{i=1}^M \omega_i y_{1,i}$
- For $i = 1, \dots, M$
  - $p_{2,i} = \operatorname{prox}_{\gamma f_i^*} y_{2,i}$
  - $q_{1,i} = p_1 - \gamma L_i^* p_{2,i}$
  - $q_{2,i} = p_2 + \gamma L_i p_1$
  - $x_i = x_i - y_{1,i} + q_{1,i}$
  - $v_i = v_i - y_{2,i} + q_{2,i}$

The weak convergence of the sequence of iterates x toward a minimizer of $\sum_{i=1}^M \omega_i f_i(L_i)$ is established (and further generalized) in [46] under the assumption that

$$\exists x \in \mathbb{R}^N | \sum_i \omega_i L_i^* \circ (\delta f_i) \circ L_i x = 0.$$

An advantage of the M+SFBF algorithm is the absence of matrix inversion for the linear operator handling, contrary to the generalized PPXA+ method [177]. We discuss in Chapter 3 cases where the M+SFBF algorithm is more efficient than PPXA in an application to image restoration.

1.4 Manuscript organization

The manuscript is organized as follows. In the next two chapters, two new flow-based methods are introduced. Chapter 2 deals with image segmentation, and Chapter 3 adresses image restoration problems. The optimization tools introduced in Sections 1.3.2 and 1.3.3 are employed for solving the considered problems, exploiting primal-dual formulations. In Chapter 4, we introduce a framework that unifies and generalizes several state-of-the-art graph-based

segmentation algorithms, namely graph cuts, random walker, shortest paths, and watershed. This generalization allows us to exhibit a new case, for which we developed a globally optimal optimization method. In Chapter 5, we further generalize and extend the framework to applications beyond image segmentation, specifically to image filtering, stereovision and surface reconstruction. The appendix includes some additional technical points refereed in the chapters, and an index is available at the end of the manuscript. Finally, the conclusion contains a summary of the work, some perspectives, and a table presenting the studied and proposed energy formulations on a unified way.

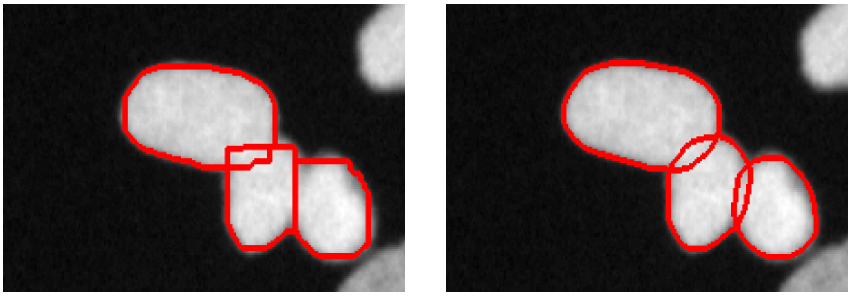
Chapter 2

Combinatorial Continuous Maximum Flow

Maximum flow (and minimum cut) algorithms have had a strong impact on computer vision. In particular, as briefly reviewed in Chapter 1, graph cut algorithms provide a mechanism for the discrete optimization of an energy functional which has been used in a variety of applications such as image segmentation, stereo, image stitching and texture synthesis. Algorithms based on the classical formulation of max-flow defined on a graph are known to exhibit metrication artifacts (as shown in Figure 2.1) in the solution. Therefore, a recent trend has been to instead employ a spatially *continuous* maximum flow (or the dual min-cut problem) in these same applications to produce solutions with no metrication errors. However, known fast continuous max-flow algorithms have no stopping criteria or have not been proved to converge.

In this work[1], we revisit the continuous max-flow problem and show that the analogous

[1]The content of this chapter is to appear in *SIAM journal on imaging sciences* [73]



Maximum flow (Graph Cuts) Continuous Maximum flow [10]

Figure 2.1: Example of metrication (blockiness) artifacts in cells segmentation obtained with classic max flow, and solution obtained optimising the continuous max flow problem.

discrete formulation is *different* from the classical max-flow problem. We then apply an appropriate combinatorial optimization technique to this combinatorial continuous max-flow (CCMF) problem to find a null-divergence solution that exhibits no metrication artifacts and may be solved exactly by a fast, efficient algorithm with provable convergence. Finally, by exhibiting the dual problem of our CCMF formulation, we clarify the fact, already proved by Nozawa in the continuous setting, that the max-flow and the total variation problems are not always equivalent.

2.1 Introduction

Energy formulations typically include a term which minimizes the boundary length (or surface area in 3D) of a region or the total variation of a scalar field in addition to a data term and/or hard constraints. In this chapter, we focus on image segmentation as the example application on which to base our exposition. Indeed, segmentation has played a prominent (and early) role in the development of the use of global optimization methods in computer vision, and often forms the basis of many other applications [36, 83, 129].

Metrication errors from the min-cut approach

Briefly described in Section 1.2.2, the max-flow/min-cut problem on a graph is a classical problem in graph theory. It was soon realized that these methods introduced a metrication error. Metrication errors are clearly visible when gradient information is weak or missing, for example in the case of medical image segmentation or in some materials science applications like electron tomography, where weak edges are unavoidable features of the imaging modality. Metrication errors are also far more obvious in 3D than in 2D and increasingly so, as the dimensionality increases (see for example the 3D-lungs example in [10]). Furthermore, metrication artifacts are even more present in other applications such as surface reconstruction and rendering, where the artifacts are a lot worse than in image segmentation. Various solutions for metrication errors were proposed. One solution involved the use of a highly connected lattice with a specialty edge weighting [222], but the large number of graph edges required to implement this solution could cause memory concerns when implemented for large 2D or 3D images [10].

Continuous approaches

To avoid the metrication artifacts without increasing memory usage, one trend in recent years has been to pursue spatially *continuous* formulations of the boundary length and the related problem of total variation [10, 50, 170]. Historically, a continuous max-flow (and dual min-cut problem) was formulated by Iri [128] and then Strang [203]. Strang's continuous

formulation provided an example of a *continuization* (as opposed to *discretization*) of a classically discrete problem, but was not associated to any algorithm. Work by Appleton and Talbot [10] provided the first PDE-based algorithm to find Strang's continuous max-flows and therefore optimal min-cuts. This same algorithm was also derived by Unger *et al.* from the standpoint of minimizing continuous total variation [210]. Adapted to image segmentation, this algorithm is shown to be equivalent [209] to the Appleton-Talbot algorithm and has been demonstrated to be fast when implemented on massively parallel architectures. Unfortunately, this algorithm has no stopping criteria and has not been proved to converge. Works by Pock *et al* [180], Zach *et al* [223] and Chambolle *et al* [52] present different algorithms for optimizing comparable energies for solving multilabel problems, but again, those algorithms are not proved to converge. Some other works have been presenting provably converging algorithms, but with relatively slow convergence speed. For example, Pock and Chambolle introduce in [54] a general saddle point algorithm that may be used in various applications, but needs half an hour to segment a 350×450 image on a CPU and still more than a dozen seconds on a GPU (Details are given in the experiments section).

Links and differences with total variation minimization

G. Strang [204] has shown the continuous max-flow problem for the l_2 norm to be the dual of the total variation (TV) minimization problem under some assumptions. Most methods minimizing TV focus on image filtering as an application, and even if those methods are remarkably fast in denoising applications, segmentation problems require a lot more iterations for those algorithms to converge. As stated previously, continuous max-flow is dual with total variation *in the continuous setting under restrictive regularity conditions [171].* In fact, Nozawa [172] showed that there is a duality gap between weighted TV and weighted max-flow under some conditions in the continuous domain. Thus, it is important to note that weighted TV problems are not equivalent to the weighted maximum flow problem studied in this chapter. Furthermore, many works assume that the continuous-domain duality holds algorithmically, but we show later in this chapter that at least in the combinatorial case this is not true.

The previous works for solving the max-flow problem illustrate two difficulties with continuous-based formulation, that are (1) the discretization step, which is necessary for deriving algorithms, but may break continuous-domain properties; and (2) the convergence, both of the underlying continuous formulation, and the associated algorithm, which itself depends on the discretization. It is very well known that even moderately complex systems of PDEs may not converge, and even existence of solutions is sometimes not obvious [94]. Even when existence and convergence proofs both exist, sometimes algorithmic convergence may be slow in practice. For these reasons, combinatorial approaches to the maximal flow

and related problems are beginning to emerge.

Combinatorial approaches

Lippert presented in [153] a combinatorial formulation of an isotropic continuous max-flow problem on a planar lattice, making it possible to obtain a provably optimal solution. However, in Lippert's work, parameterization of the capacity constraint is tightly coupled to the 4-connected squared grid, and the generalization to higher dimensions seems non-trivial. Furthermore it involves the multiplication of the number of capacity constraints by the degree of each node, thus increasing the dimension of the problem. This formulation did not lead to a fast algorithm. The author compared several general solvers, quoting an hour as their best time for computing a solution on a 300×300 lattice.

Motivation and contributions

In this chapter, we pursue a combinatorial reformulation of the continuous max-flow problem initially formulated by Strang, which we term *combinatorial continuous maximum flow* (CCMF). Viewing our contribution differently, we adopt a discretization of continuous max-flow as the primary problem of interest, and then we apply fast combinatorial optimization techniques to solve the discretized version.

Our reformulation of the continuous max-flow problem produces a (divergence-free) flow problem defined on an arbitrary graph. Strikingly, CCMF are not equivalent to the discretization of continuous max-flows produced by Appleton and Talbot or that of Lippert (if for no other reason than the fact that CCMF is defined on an arbitrary graph). Moreover, *CCMF is not equivalent to the classical max-flow on a graph*. In particular, we will see that the difference lies in the fact that capacity constraints in classical max-flow restrict flows along graph edges while the CCMF capacity constraints restrict the total flow passing through the nodes.

The CCMF problem is convex. We deduce an expression of the dual problem, which allows us to employ a primal-dual interior point method to solve it, such as interior point methods have been used for graph cuts [22] and second order cone problems in general [106]. The CCMF problem has several desirable properties, including:

1. the solution to CCMF on a 4-connected lattice avoids the metrication errors. Therefore, the gridding error problems may be solved without the additional memory required to process classical max-flow on a highly-connected graph;

2. in contrast to continuous max-flow algorithms of Appleton-Talbot (AT-CMF) and equivalent, the solution to the CCMF problem can be computed with guaranteed convergence in practical time;

3. the CCMF problem is formulated on an arbitrary graph, which enables the algorithm to incorporate nonlocal edges [47, 89, 117] or to apply it to arbitrary clustering problems defined on a graph; and

4. the algorithm for solving the CCMF problem is fast, easy to implement, compatible with multi-resolution grids and is straightforward to parallelize.

Our computation of the CCMF dual further reveals that duality between total variation minimization and maximum flow does not hold for CCMF and combinatorial total variation (CTV). The comparison between those two combinatorial problems is motivated by several interests:

1. for clarification: in the continuous domain, the duality between TV and MF holds under some regularity constraints. In the discrete anisotropic domain (i.e. Graph Cuts), this duality always holds. However, In the isotropic weighted discrete case, i.e. in the CTV, AT-CMF or CCMF cases, we are not aware of any discretization such that the duality holds. It is not obvious to realize that the duality does not hold in the discrete setting even though it may in the continuous case. We present clearly the differences between the two problems, theoretically, and in term of results;

2. to expose links between CCMF and CTV: We prove that the weak duality holds between the two problems; and

3. for efficiency: in image segmentation, the fastest known algorithms to optimize CTV are significantly slower than CCMF. Therefore, there is reason to believe that CCMF may be used to efficiently optimize energies for which TV has been previously shown to be useful (e.g., image denoising).

In the next section, we review the formulation of continuous max-flow, derive the CCMF formulation and its dual and then provide details of the fast and provably convergent algorithm used to solve the new CCMF problem.

2.2 Method

Our combinatorial reformulation of the continuous maximum flow problem leads to a formulation on a graph which is *different* from the classical max-flow algorithm. Before proceeding to our formulation, we review the continuous max-flow and the previous usage of this formulation in computer vision.

2.2.1 The continuous max-flow (CMF) problem

First introduced by Iri [128], Strang presents in [203] an expression of the continuous maximum flow problem

$$\begin{aligned} &\max F_{st}, \\ \text{s.t.} \quad &\nabla \cdot \overrightarrow{F} = 0, \\ &||\overrightarrow{F}|| \leq g. \end{aligned} \tag{2.1}$$

Here we denote by F_{st} the total amount of flow going from the source location s to the sink location t, $\overrightarrow{F}$ is the flow, and g is a scalar field representing the local metric distortion. The solution to this problem is the exact solution of the geodesic active contour (GAC) (or surface) formulation [49]. In order to solve the problem (2.1), the Appleton-Talbot algorithm (AT-CMF) [10] solves the following partial differential equation system:

$$\begin{aligned} \frac{\partial P}{\partial \tau} &= -\nabla \cdot \overrightarrow{F}, \\ \frac{\partial \overrightarrow{F}}{\partial \tau} &= -\nabla P, \\ \text{s. t. } &||\overrightarrow{F}|| \leq g. \end{aligned} \tag{2.2}$$

Here P is a potential field similar to the excess value in the Push-Relabel maximum flow algorithm [105]. AT-CMF is effectively a simple continuous computational fluid dynamics (CFD) simulation with non-linear constraints. It uses a forward finite-difference discretization of the above PDE system subject to a Courant-Friedrich-Levy (CFL) condition, also seen in early level-sets methods. At convergence, the potential function P approximates an indicator function, with 0 values for the background labels, and 1 for the foreground, and the flow $\overrightarrow{F}$ has zero-divergence almost everywhere. However, there is no guarantee of convergence for this algorithm and, in practice, many thousands of iterations can be necessary to achieve a binary P, which can be very slow.

Although Appleton-Talbot is a continuous approach, applying this algorithm to image processing involves a discretization step. The capacity constraint $||\overrightarrow{F}|| \leq g$ is interpreted as $\max {F_{x(i)}}^2 + \max {F_{y(i)}}^2 \leq g_i^2$, with $F_{x(i)}$ the outgoing flow of edges linked to node i along the x axis, and $F_{y(i)}$ the outgoing flow linked to node i along the y axis. We notice that the weights are associated with point locations (pixels), which will correspond later to a node-weighted graph.

In the next section, we use the operators of discrete calculus to reformulate the continuous max-flow problem on a graph and show the surprising result that the graph formulation of

the continuous max-flow leads to a different problem from the classical max-flow problem on a graph.

2.2.2 A discrete calculus formulation

Before establishing the discrete calculus formulation of the continuous max-flow problem, we specify our notation. We refer to Section1.1 for general definitions on graphs. In this chapter, a transport graph $G(V, E)$ comprises two additional nodes, a source s and a sink t, and additional edges linking some nodes to the sink and some to the source. Including the source and the sink nodes, the cardinalities of G are given by $n = |V|$ and $m = |E|$. We define a flow through edge e_{ij} as F_{ij} where $F_{ij} \in \mathbb{R}$ and use the vector $F \in \mathbb{R}^m$ to denote the flows through all edges in the graph . Each edge is assumed to be oriented, such that a positive flow on edge e_{ij} indicates the direction of flow from v_i to v_j, while a negative flow indicates the direction of flow from v_j to v_i.

The incidence matrix is a key operator for defining a combinatorial formulation of the continuous max-flow problem. In our formulation of continuous max-flow, we use the expression $|A|$ to denote the matrix formed by taking the absolute value of each entry individually.

Given these definitions, we now produce a discrete (combinatorial) version of the continuous max-flow of (2.1) on a transport graph. As in [89, 113, 119], the continuous vector field indicating flows may be represented by a vector on the edge set, F. Additionally, the combinatorial divergence operator allows us to write the first constraint in (2.1) as $A^\top F = 0$. The second constraint in (2.1) involves the comparison of the point-wise norm of a vector field with a scalar field. Therefore, we can follow [89, 113, 119] to define the point-wise ℓ_2 norm of the flow field F as $\sqrt{|A^\top|F^2}$. In our notations here, as in the rest of the chapter, $F^2 = F \cdot F$ denotes a element-wise product, "$\cdot$" denoting the Hadamard (element-wise) product between the vectors, and the square root of a vector is also here and in the rest of the chapter the vector composed of the square roots of every elements.

Putting these pieces together, we obtain

$$\begin{aligned} &\max F_{st}, \\ \text{s.t.} \quad &A^\top F = 0, \\ &|A^\top|F^2 \leq g^2. \end{aligned} \tag{2.3}$$

Written differently, the capacity constraint is $\forall v_i \in V, \quad \sum_{e_{ij} \in E} F_{ij}^2 + \sum_{e_{ji} \in E} F_{ji}^2 \leq g_i^2$. Compare this formulation to the classical max-flow problem on a graph, given in our notation

as

$$\begin{aligned} &\max F_{st}, \\ \text{s.t.} \quad &A^\top F = 0, \\ &|F| \leq w. \end{aligned} \tag{2.4}$$

By comparing these formulations of the traditional max-flow with our combinatorial formulation of the continuous max-flow, it is apparent that the key difference is in the capacity constraints. In both formulations, the flow is defined through edges, but in the classical case the capacity constraint restricts flow through an edge while the CCMF formulation restricts the amount of flow passing through a node by taking in account the neighboring flow values. This contrast-weighting applied to nodes (pixels) has been a feature of several algorithms with a continuous formulation, including geodesic active contours [49], CMF [10], TVSeg [210], and [54]. On the other hand, the problem defined in (2.3) is defined *on an arbitrary graph* in which contrast weights (capacities) are almost always defined on *edges*. The node-weighted capacities fit Strang's formulation of a scalar field of constraints in the continuous max-flow formulation and therefore our graph formulation of Strang's formulation carries over these same capacities. The most important point here is not having expressed the capacity on the nodes of the graph but rather how the flow is enforced to be bounded by the metric in each node. Bounding the norm of neighboring flow values in each node simulates a closer behavior of the continuous setting than if no contribution of neighboring flow was made as in the standard max-flow problem. Figure 2.2 illustrates the relationship of the edge capacity constraints in the classical max-flow problem to the node capacity constraints in the CCMF formulation. In particular, the minimal cut is localized on saturated edges in the classic max flow case, whereas the minimal cut appears on saturated nodes in the CCMF case. More formally,

Definition 2.2.1. *A node v_i is* saturated *if $|A^\top|_i F^2 = g_i^2$, where $|A^\top|_i$ indicates the i^{th} row of $|A^\top|$.*

In the context of image segmentation, the vector g varies inversely to the image gradient. We propose to use, as in [210],

$$g = \exp(-\beta||\nabla I||_2), \tag{2.5}$$

where I indicates the image intensity. For simplicity, this weighting function is defined for greyscale images, but g could be used to penalize changes in other relevant image quantities, such as color or texture. Before addressing the solution of the CCMF problem, we consider the dual of the CCMF problem and, in particular, the sense in which it represents a minimum cut. Since the cut weights are present on the nodes rather than the edges, we must expect the minimum cut formulation to be different from the classical minimum cut on a graph.

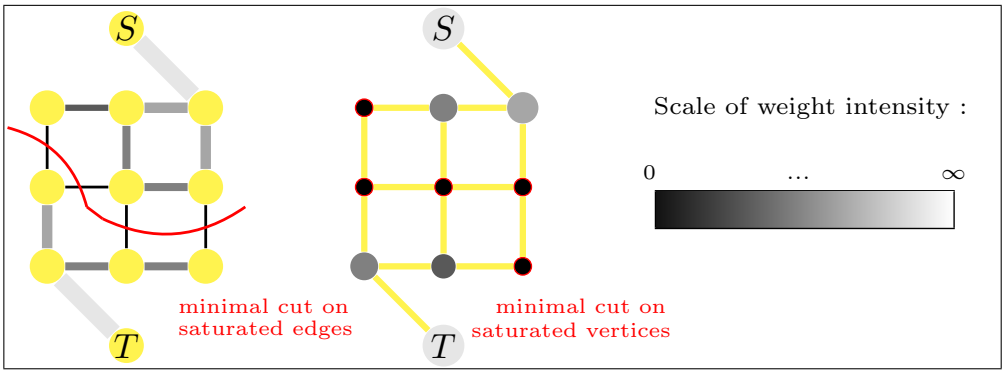


Figure 2.2: Different discretization of metric and different type of cuts obtained with classical max-flow and with the combinatorial continuous max-flow (CCMF) formulations.
The difference between classical max-flow and CCMF is that classical max-flow uses edge-weighted capacities while CCMF uses node-weighted capacities. This difference is manifest in the different solutions obtained for both algorithms and the algorithms required to find a solution. Specifically, the solution to the CCMF problem on a lattice does not exhibit metrication bias.

2.2.3 The CCMF dual problem

Classically, the max-flow problem is dual to the min-cut problem, allowing a natural geometric interpretation of the objective function. In order to provide the same interpretation, we now consider the dual problem to the CCMF and show that we optimize a node-weighted minimum cut. In the following proposition, we denote the element-wise quotient of two vectors $u = [u_1, \dots, u_k]$ and $v = [v_1, \dots, v_k]$ by $u \cdot / v = [u_1/v_1, \dots, u_k/v_k]$. We also denote $\mathbf{1}^n$ a unit vector $[1, \dots, 1]$ of size n, and recall that the square exponent v^2 of a vector v represents the resulting vector of the element-wise multiplication $v \cdot v$.

Property 1. *In a transport graph G with m edges, n nodes, we define a vector c of $\mathbb{R}^m$, composed of zeros except for the element corresponding to the source/sink edge which is -1. Let λ and ν be two vectors of $\mathbb{R}^n$. The CCMF problem*

$$\begin{aligned} \max_{F \in \mathbb{R}^m} \quad & -c^\top F, \\ s.\ t. \quad & A^\top F = 0, \\ & |A^\top| F^2 \leq g^2. \end{aligned} \tag{2.6}$$

has for dual

$$\begin{aligned} \min_{\lambda \in \mathbb{R}^n,\ \nu \in \mathbb{R}^n} \quad & \lambda^\top g^2 + \tfrac{1}{4} \Big(\mathbf{1}^n \cdot / (|A|\lambda) \Big)^\top \Big((c + A\nu)^2 \Big), \\ s.\ t. \quad & \lambda \geq 0, \end{aligned} \tag{2.7}$$

equivalently written in (2.10)*, and the optimal solution* (F^*, λ^*, ν^*) *verifies*

$$\max_F c^\top F = c^\top F^* = 2{\lambda^*}^\top g^2, \tag{2.8}$$

and the n *following equalities*

$$\lambda^* \cdot |A^\top| \Big((c + A\nu^*) \cdot / (|A|\lambda^*) \Big)^2 = 4\lambda^* \cdot g^2. \tag{2.9}$$

The proof, using the Lagrange duality theorem, is given in Appendix A.1.1. The vectors $\lambda \in \mathbb{R}^n$ and $\nu \in \mathbb{R}^n$ correspond to two Lagrange multipliers.

The expression of the CCMF dual may be written in summation form as

$$\min_{\lambda,\nu} \sum_{v_i \in V} \overbrace{\lambda_i g_i^2}^{\text{weighted cut}} + \overbrace{\frac{1}{4} \sum_{e_{ij} \in E \setminus \{s,t\}} \frac{(\nu_i - \nu_j)^2}{\lambda_i + \lambda_j}}^{\text{smoothness term}} + \overbrace{\frac{1}{4} \frac{(\nu_s - \nu_t - 1)^2}{\lambda_s + \lambda_t}}^{\text{source/sink enforcement}} \tag{2.10}$$
$$\text{s. t. } \lambda_i \geq 0 \quad \forall i \in V.$$

Interpretation: The optimal value λ^* is a weighted indicator of the saturated vertices (See Def. 2.2.1):

$$\lambda^*(v_i) \begin{cases} > 0 & \text{if } |A^\top|_i F^2 = g(v_i)^2, \\ = 0 & \text{otherwise.} \end{cases} \tag{2.11}$$

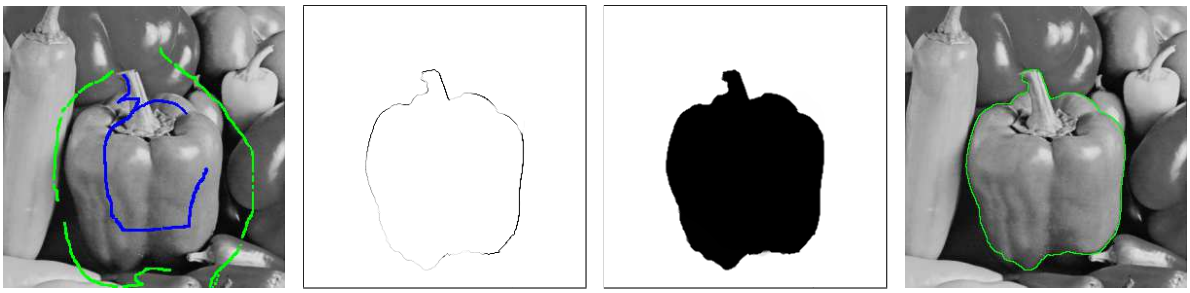
The variables ν_s and ν_t are not constrained to be set to 0 and 1, only their difference is constrained to be equal to one. Thus their value range between a constant and a constant plus one. Let us call this constant δ, and without loss of generality one can consider that $\delta = 0$. The term ν is at optimality a weighted indicator of the source/sink/saturated vertices partition:

$$\nu^*(v_i) = \begin{cases} 0 + \delta & \text{if } v_i \in S, \\ \text{a number between } (0+\delta) \text{ and } (1+\delta) & \text{if } |A^\top|_i F^2 = g(v_i)^2, \\ 1 + \delta & \text{if } v_i \in T. \end{cases}$$

The expression (2.8) of the CCMF dual shows that the problem is equivalent to finding a minimum weighted cut defined on the nodes.

Finally, the "weighted cut" is recovered in (2.10), and the "smoothness term" is compatible with large variations of ν at the boundary of objects because of a large denominator (λ) in the contour area. An illustration of optimal λ and ν on an image is shown on Fig. 2.3.

Several optimization methods have been tested for solving the CCMF problem, its dual and both jointly. We report in Appendix A.1.3 a proximal method for solving the primal



(a) (b) λ (c) ν (d) Threshold ν at .5

Figure 2.3: Solution to the dual of the CCMF problem in a seeded segmentation example. The dual problem to CCMF is a node-weighted minimum cut in which the variable λ is a weighted indicator vector labeling boundary nodes and the variable ν is a nearly binary vector indicating the source/sink regions. As a result, the contours of ν are slightly blurry. This is due to the equilibrium effect between the two dual variables. In practice, as λ is nonzero only in presence of a contour, ν is binary almost everywhere, except on a very thin line.

problem and a first order method for solving the dual problem. The most efficient method we found, solving both problems jointly, is the primal dual interior point method detailed in the next section.

2.2.4 Primal Dual Interior Point Method

When considering how to solve the CCMF problem (2.3), the first key observation is that the constraints bound a convex feasible region.

The real valued functions $f_i : \mathbb{R}^m \to \mathbb{R}$ defined for every node $i = 1, \ldots, n$ as $f_i(F) = |A^\top|_i F^2 - g_i^2$ are non-negative quadratic, so convex functions. We define the vector $f(F)$ of $\mathbb{R}^n$ as $f(F) = [f_1(F), \ldots, f_n(F)]^\top$.

Since the constraints are convex, the CCMF problem may be solved with a fast primal dual interior point method (see [31]), which we now review in the specific context of CCMF.

Algorithm 4: Primal dual interior point algorithm optimizing CCMF

Data: $F \in \mathbb{R}^m = 0, \lambda \in (\mathbb{R}_+^*)^n, \nu \in \mathbb{R}^n$, $\mu > 1, \epsilon > 0$.

Result: F solution to the CCMF problem (2.6) such that F_{st} is maximized under the divergence free and capacity constraints, and ν, λ solution to the CCMF dual problem (2.8).

repeat

1. Compute the surrogate duality gap $\widehat{\eta} = -f(F)^\top \lambda$ and set $t = \mu n / \widehat{\eta}$.
2. Compute the primal-dual search direction Δ_y such that $M\Delta_y = r$.
3. Determine a step length $s > 0$ and set $y = y + s\Delta_y$. $\quad (y = [F, \lambda, \nu]^\top)$

until $||r_p||_2 \leq \epsilon$, $||r_d||_2 \leq \epsilon$, *and* $\widehat{\eta} \leq \epsilon$.

The primal dual interior point (PDIP) algorithm iteratively computes the primal F and dual λ, ν variables so that the Karush-Kuhn-Tucker (KKT) optimality conditions are satisfied. This algorithm solves the CCMF problem (2.6) by applying Newton's method to a sequence of a slightly modified version of the KKT conditions. The steps of the PDIP procedure are given in Algorithm 4. Specifically for CCMF, the system $M\Delta_y = r$ system may be written

$$\begin{bmatrix} \sum_{i=1}^n \lambda_i \nabla^2 f_i(F) & Df(F)^\top & A \\ -\text{diag}(\lambda) Df(F) & -\text{diag}(f(F)) & 0 \\ A^\top & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta_F \\ \Delta_\lambda \\ \Delta_\nu \end{bmatrix} = - \begin{bmatrix} r_d = c + Df(F)^\top \lambda + A\nu \\ r_c = -\text{diag}(\lambda) f(F) - (1/t) \\ r_p = A^\top F \end{bmatrix} \tag{2.12}$$

with r_d, r_c, and r_p representing the dual, central, and primal residuals. Additionally, the derivatives are given by

$$\nabla f_i(F) = 2|A^\top|_i \cdot F, \quad Df(F) = \begin{bmatrix} \nabla f_1(F)^\top \\ \vdots \\ \nabla f_n(F)^\top \end{bmatrix}, \quad \nabla^2 f_i(F) = \text{diag} \begin{bmatrix} 2|A^\top|_{i1} \\ \vdots \\ 2|A^\top|_{im} \end{bmatrix},$$

with "$\cdot$" denoting the Hadamard (element-wise) product between the vectors.

Consequently, the primary computational burden in the CCMF algorithm is the linear system resolution required by (2.12). In the result section we present execution times obtained in our experiments using a conventional CPU implementation.

Observe that, although this linear system is large, it is very sparse and is straightforward to parallelize on a GPU [27, 114, 143], for instance using an iterative GPU solver. If it does not necessarily imply a faster solution, the asymptotic complexity of modern iterative and modern direct solvers is about the same and, in our experience, there has been a strong improvement in the performance of a linear system solver when going from a direct solver CPU solution to a conjugate gradient solver GPU solution, especially as the systems get larger.

2.3 Comparison between CCMF and existing related approaches

2.3.1 Min cut

As detailed in Section 2.3, there is a difference between the CCMF and the classic max flow formulation in the capacity constraint. Therefore, the dual problems are different. As proved by Ford and Fulkerson [97], and formalized for example in [22], the min cut problem writes with our notations of section 2.3

$$\min_u \tilde{g}^T |Au + c|. \tag{2.13}$$

We note that the ℓ_1 norm of the gradient of the solution u of the above min-cut expression turns into an ℓ_2 norm of the gradient of the solution ν in the CCMF dual (2.7).

2.3.2 Primal Dual Total variations

Unger *et al.* [209] proposed to optimize the following primal dual TV formulation

$$\begin{aligned} \min_u \max_F \sum_{i,j} (u_{i+1,j} - u_{i,j})F^1_{i,j} + (u_{i,j+1} - u_{i,j})F^2_{i,j} + \text{data fidelity}, \\ \text{s.t. } \sqrt{(F^1_{i,j})^2 + (F^2_{i,j})^2} \leq g_{i,j}. \end{aligned} \tag{2.14}$$

We can note that the CCMF flow capacity constraint in a 4-connected lattice is similar to the constraint in (2.14) with a slight modification, and would be with finite-element notations

$$\sqrt{(F^1_{i-1,j})^2 + (F^2_{i,j-1})^2 + (F^1_{i,j})^2 + (F^2_{i,j})^2} \leq g_{i,j}. \tag{2.15}$$

In contrast to this finite element discretization, the provided CCMF formulation of continuous max-flow is defined arbitrary graphs. Furthermore, we note that the optimization procedure employed to solve (2.14) generalizes Appleton-Talbot's algorithm.

2.3.3 Combinatorial total variations

We now compare the dual CCMF problem with the Combinatorial Total Variation (CTV) problem. We note that the CCMF dual is different than CTV and discuss the weak duality of the two problems.

We recall the continuous expression of total variation given by Strang (in 2D)

$$\begin{aligned} \min_u \iint_\Omega g \, ||\nabla u|| \;\; \mathrm{d}x \, \mathrm{d}y, \\ \text{s. t.} \quad \int_\Gamma uf \, \mathrm{ds} = 1, \end{aligned} \tag{2.16}$$

where Ω represents the image domain and Γ the boundary of Ω. The source and sink membership is represented by f, such that $f(x,y) > 0$ if (x,y) belongs to the sink, $f(x,y) < 0$ if (x,y) belongs to the source, and $f = 0$ otherwise. Considering a transport graph G, and a vector u defined on the nodes, this continuous problem may be written with combinatorial operators

$$\begin{aligned} &\min_{u \in \mathbb{R}^n} g^\top \sqrt{|A^\top|(Au)^2}, \\ &\text{s. t.} \quad u_s - u_t = 1, \end{aligned} \tag{2.17}$$

where the square root operator of a vector $v = [v_1, \ldots, v_k]$ is an element-wise square root operator $\sqrt{v} = [\sqrt{v_1}, \ldots, \sqrt{v_k}]$. Another way to write the same problem is

$$\begin{aligned} &\min_{u} \sum_{v_i \in V} g_i \sqrt{\sum_{e_{ij} \in E} (u_i - u_j)^2}, \\ &\text{s. t.} \quad u_s - u_t = 1. \end{aligned} \tag{2.18}$$

We note that in these equations the capacity g must be defined on the vertices. Although we describe this energy as "combinatorial total variation", due to its derivation in discrete calculus terms, it is important to note that this formulation is very similar to the discretization which has appeared previously in the literature from Gilboa and Osher [103], and Chambolle [50] (if we allow $g = 1$ everywhere).

2.3.3.1 CCMF and CTV are not dual

Strang proved that the continuous max-flow problem is dual to the total variation problem under some assumptions on g. Remarkably, this duality is not observed in the discrete case in which the CCMF dual and CTV problems are given by

$$\begin{aligned} &\min_{\lambda,\nu} \ \lambda^\top g^2 + \tfrac{1}{4}\Big(\mathbf{1}^n \cdot /(|A|\lambda)\Big)^\top \Big((c + A\nu)^2\Big), \\ &\text{s. t.} \ \lambda \geq 0, \end{aligned} \quad \not\equiv \quad \begin{aligned} &\min_{u} g^\top \sqrt{|A^\top|(Au)^2}, \\ &\text{s. t.} \quad u_s - u_t = 1. \end{aligned}$$

We note that the analytic expressions are different and also not equivalent. The duality of the classical max-flow problem with the minimum cut holds because in the expression of the Lagrangian function, is possible to deduce a value of λ by substituting it into the dual problem so that it only depends on ν. However, λ in the dual CCMF problem (2.8) depends on several values of neighboring λ and ν. Thus, the CCMF dual problem cannot be simplified by removing a variable, for example by identifying ν and u in the CTV problem. Said differently, combining a null-divergence constraint with an *anisotropic* capacity constraint

in the classical max-flow formulation allows the duality to hold; in contrast, in the CCMF formulation, combining a null-divergence constraint with an *isotropic* capacity constraint does not allow such duality to hold.

Numerically we can also show on examples that the value $\max F_{st}$ of the flow optimizing CCMF is not equal to the minimum value of CTV (See Table 2.1).

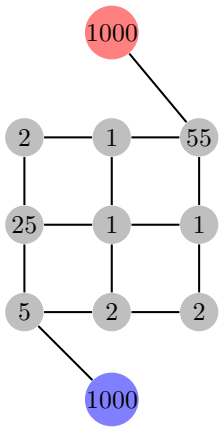
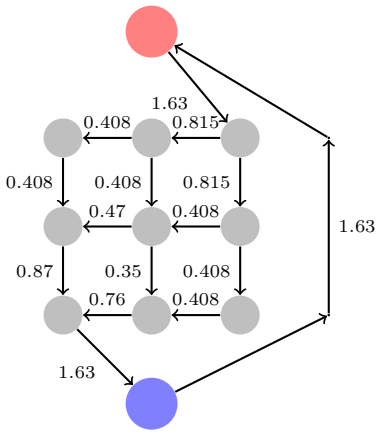
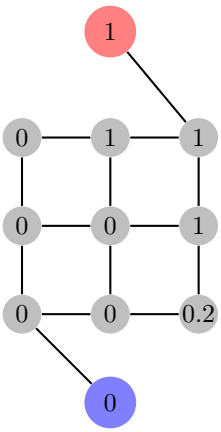
Transport graph with weights g on the nodes	**Combinatorial Continuous Max-Flow (CCMF)**	**Combinatorial Total Variation (CTV)**
	$\max_F \quad F_{st},$ s. t. $\quad A^\top F = 0,$ $\quad\quad\; \lvert A^\top \rvert F^2 \leq g^2.$	$\min_u \quad g^\top \sqrt{\lvert A^\top \rvert (Au)^2},$ s. t. $\quad u_s - u_t = 1.$
		
	$F^*_{st} = 1.63$	$g^\top \sqrt{\lvert A^\top \rvert (Au^*)^2} = 8.16$

Table 2.1: Example illustrating the difference between optimal solutions of combinatorial continuous maximum flow problem (CCMF), and combinatorial total variation (CTV).

Even if CCMF is not dual with CTV, the two problems are weakly dual (more details are given in Appendix A.1.2).

2.4 Results

We now present applications of Combinatorial Continuous Maximum Flow in image segmentation. To be used as a segmentation algorithm, three solutions are possible: the dual variable ν may be used directly if the user needs a matted result, otherwise ν may be thresholded, or finally an isoline or isosurface may be extracted from ν.

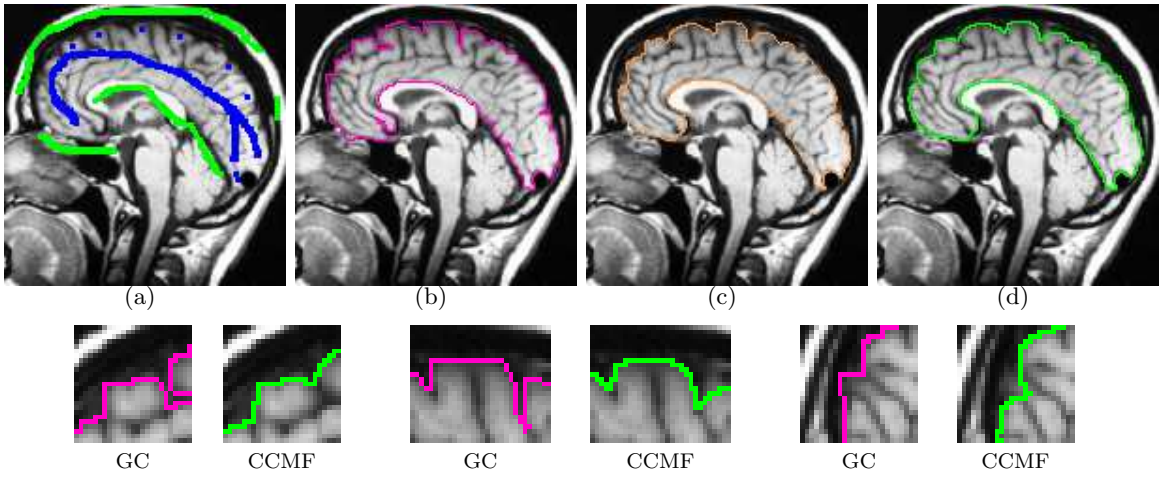
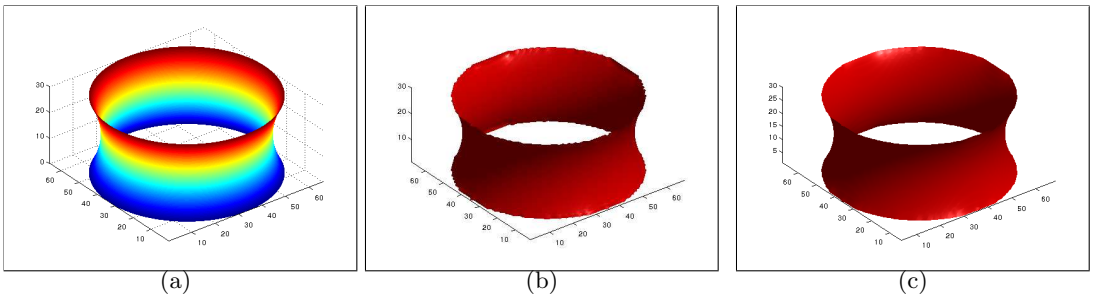


Figure 2.4: Comparison of brain segmentation results using GC, AT-CMF, and CCMF. (a) Original image with foreground and background seeds. (b,c,d) Segmentation obtained with (b) graph cuts (GC), (c) AT-CMF, threshold of P (obtained after 10000 iterations), (d) CCMF, threshold of ν (15 iterations).

In the introduction we discussed how several works are related to CMF. Some are equivalent or slight modifications of AT-CMF for non-segmentation applications [180, 209, 210, 223]. As in the original AT-CMF work, none of these come with a convergence proof. In contrast, the work of [179] and [153] are provably convergent but both are very slow, and do not generalize easily to 3D image segmentation. Consequently, it seems reasonable to compare our segmentation method to the original AT-CMF as representative of the continuous approach, as well as graph cuts, which represent the purely discrete case. Finally, even if the total variation problem is different from the CMF problem (existence of a duality gap in the continuous [172]), the two problems are related enough to be compared. Specifically, we will also compare with Chambolle and Pock's recent work [54] which presented an algorithm for optimizing a class of primal dual problems, and in [54][Section 6.3.4], optimize a TV-based energy for image segmentation.

Our validation is intended to establish three properties of the CCMF algorithm. First, we establish that the CCMF does avoid the metrication artifacts exhibited by conventional graph cuts (on a 4-connected lattice). This property is established by examples on a natural

image and the recovery of the classical catenoid structure as the minimal surface spanning two rings. Second, we compare the convergence of the CCMF algorithm to the AT-CMF algorithm to show that the CCMF algorithm converges more quickly and in a more stable fashion. Finally, we establish that our formulation of the CMF problem does not degrade segmentation performance on a standard database. In fact, because of the reduction in metrication error our algorithm gives improved numerical results. For this experiment, we use the GrabCut database to compare the quality of the segmentation algorithms. In addition to the above tests, we demonstrate through examples that the CCMF algorithm is also flexible enough to incorporate prior (unary) terms and to operate in 3D.



(a) (b) (c)

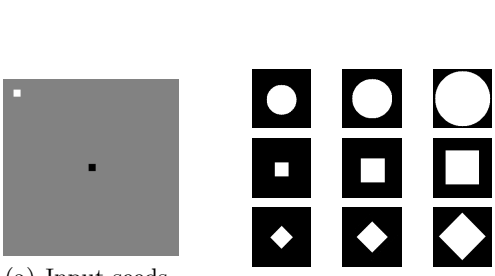
Figure 2.5: Comparison of AT-CMF and CCMF results on the catenoid test problem. The source is constituted by two full circles and sink by the remaining boundary of the image. (a) Surface computed analytically, (b) isosurface of P obtained by CMF, (c) isosurface of ν obtained by CCMF. The root mean square error (RSME) has been computed to evaluate the precision of the results to the surface computed analytically. The RSME for CMF is 1.98 and for CCMF 0.75. The difference between those results is due to the fact that the CMF algorithm enforces exactly the source and sink points, leading to discretization around the disks. In contrast, the boundary localized around the seeds of ν is smooth, composed of grey levels. Thus the resulting isosurface computed by CCMF is more precise.

2.4.1 Metrication artifacts and minimal surfaces

We begin by comparing the CCMF segmentation result with the classical max-flow algorithm (graph cuts). Figure 2.4 shows the segmentation of a brain, in which the contours obtained by graph cuts are noticeably blocky in the areas of weak gradient, while the contours obtained by both AT-CMF and CCMF are smooth.

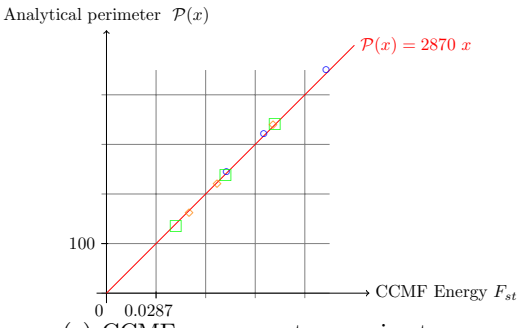
In the continuous setting, the maximum flow computed in a 3D volume produces a minimal surface. The CCMF formulation may be also recognized as a minimal surface problem. In the dual formulation, the objective function is equivalent to a weighted sum of surface nodes. In [10], Appleton and Talbot compared the surfaces obtained from their algorithm with the analytic solution of the catenoid problem to demonstrate that their algorithm was a good

approximation of the continuous minimal surface and was not creating discretization artifacts. The catenoid problem arises from consideration of two circles with equal radius whose centers lie along the z axis. The minimal surface which forms to connect the two circles is known as a catenoid. The catenoid appears in nature, for example by creating a soap bubble between two rings. In order to demonstrate that CCMF is also finding a minimal surface, we performed the same catenoid experiment as was in [10]. The results are displayed in Figure 2.5, where we show that CCMF approximates the analytical solution of the catenoid with even greater fidelity than the AT-CMF example.

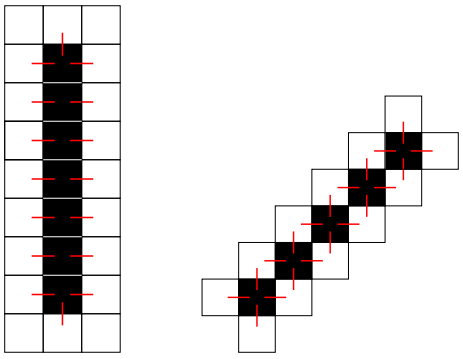


(a) Input seeds

(b) Input images

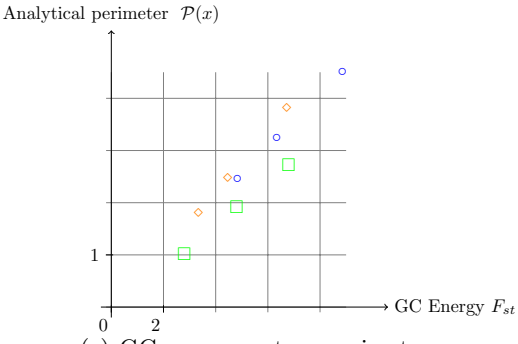


(c) CCMF energy vs true perimeters



GC cut cost : 16 GC cut cost : 20

(d) Graph cut cost for lines of approximately same length



(e) GC energy vs true perimeters

Figure 2.6: Comparison of perimeters computed analytically and with Graph cuts and CCMF. To estimate the boundary length with CCMF, segmentations have been produced using the input images in (b) with the foreground and background seeds that appears in (a). (c) : plot showing the linear relation between the true perimeters and the energies obtained with CCMF. The symbols used to represent the dots correspond to the objects segmented in the images. A similar relation of proportionality is obtained with CP-TV. (d) : value of the cut obtained for a diagonal and a vertical line of approximately same length using graph cuts. The difference observed explains (e) : plot showing the nonlinear relation between the true perimeters and the energies obtained with Graph cuts.

In order to verify that the solution obtained by CCMF approximates perimeters of planar objects in 2D, we segmented several shapes – squares, discs – with known perimeters, scaled at different size, and compared the analytic perimeters with the energies obtained by CCMF. The results are reported in Fig. 2.6, where we observe that the true boundary length is proportional to the energy obtained with CCMF.

2.4.2 Stability, convergence and speed

We may compare the segmentation results using ν to Appleton-Talbot's result using P. We recall that AT-CMF solves the partial differential equation system (2.3) in order to solve the continuous maximum flow problem (2.1), but no proof was given for convergence. The potential function P approximates an indicator function, with 0 values for the background labels, and 1 for the foreground. It can be difficult to know when to stop the AT-CMF algorithm, since its iterations may oscillate, as displayed in Figure 2.7 on a synthetic image. In contrast, the CCMF algorithm is guaranteed to converge and smoothly approaches the optimum solution.

We also compare segmentation results with results obtained by the recent algorithm of Chambolle and Pock [54] optimizing a total variation based energy using a dual variable. In comparison to our work, the dual variable is not a flow (no constraint on the divergence and no proof of convergence toward a minimal surface as defined in [203] or [10]). In the literature, the continuous-domain duality between TV and max-flow is assumed, but in computational practice they are really different. Nevertheless, even as the method of Chambolle-Pock is solving a different optimization problem, we performed numerical comparative tests. The problem of Chambolle and Pock (now denoted CP-TV) applied to binary segmentation optimizes

$$\min_{x\in[0,1]^n} \max_{F\in\mathbb{R}^m, ||F||_\infty \leq 1} F^\top (Ax) + h^\top x + \text{ hard constraints}, \tag{2.19}$$

where $A \in \mathbb{R}^{m\times n}$ is the incidence matrix of the graph of n nodes and m edges defined on the image, $h \in \mathbb{R}^n$ is the metric on the nodes, defined in equation (2.5).

The CCMF algorithm is faster than both AT-CMF and CP-TV for 2D image segmentation. We have implemented CCMF in Matlab. We used an implementation of Appleton-Talbot's algorithm in `C++` provided by the authors, and a Matlab software implementing CP-TV on CPU, also provided by the respective authors. The three implementations make use of multi-threaded parallelization, and the CPU times reported here were computed on a Intel Core 2 Duo (CPU 3.00GHz) processor, with 2 Gb of RAM. The CCMF average computation time on a 321×481 image is 181 seconds after 21 iterations. For AT-CMF, 80000 iterations require 547 seconds. For CP-TV, the average number of iterations needed for convergence is 36000, requiring an average time of 1961 seconds on CPU, and the median

computation time is 1406 seconds. Note that a GPU implementation of CP-TV, also provided by the authors, achieves a speed gain factor of about 100 on a NVidia Quadro 3700. Although it was not pursued here, it should be noted that the CCMF optimization approach could also fully benefit from parallelization (e.g., on a GPU) because the core computation is to solve a linear system of equations. Realizing the benefit of a GPU solution would require the use of an iterative algorithm such as conjugate gradients or multigrid [27, 143], which would make comparisons to the direct solver used here less immediate. However, previous examples in the literature have suggested that this change in solver has not created difficulties in order to realize the benefits of a GPU architecture [114].

Sometimes, it may not always be necessary to wait so long for the complete convergence of CCMF (or AT-CMF) to obtain acceptable results. In cases where images exhibiting sufficiently strong gradients, one iteration may be enough to obtain a satisfying segmentation. On such an image, one iteration of CCMF, and 100 iterations of AT-CMF show acceptable approximate results reached only after about 2 seconds for either algorithm. However, one cannot always rely on strong edges, and the power of our CCMF formulation is to behave in a predictable fashion even when edges are weak.

We may also compare the computation time of CCMF to the optimization of total variation using the Split Bregman method [108]. Optimizing TV on a 100×100 image requires 5000 iterations and takes 23 seconds with a sequential implementation of Split Bregman in C. On the same image, CCMF required only 17 iterations to reach the convergence criteria $||r_d|| < 1$ and $||\widehat{\eta}|| < 2$, taking 4.7 seconds with a Matlab implementation (using a 2-threaded solver). We conclude that TV-based methods appear to be quite slow in the context of image segmentation, although we employed the Split Bregman algorithm which is known as one of the fastest algorithms for TV optimization for image denoising, and the very recent CP-TV algorithm. This difference in speed between the denoising and segmentation applications is due to the very large number of iterations required to convergence to a binary segmentation.

2.4.3 Segmentation quality

In this experiment, we compared our CCMF formulation to the AT-CMF formulation, CP-TV approach for segmentation and conventional graph cuts on the problem of image segmentation, to determine if there were strong performance differences between the formulations. We expect that there would not be, since in principle all three formulations are trying to "minimize the cut", but have slightly different definitions of the cut length. The primary advantage that we expect from our formulation is in the reduction of metrication artifacts (as compared to conventional graph cuts) and speed/convergence (as compared to AT-CMF).

Our experiment consists of testing the Graph Cut, AT-CMF, CP-TV, and Combinatorial Continuous Max-Flow algorithms on a database with the same seeds. We used the Microsoft

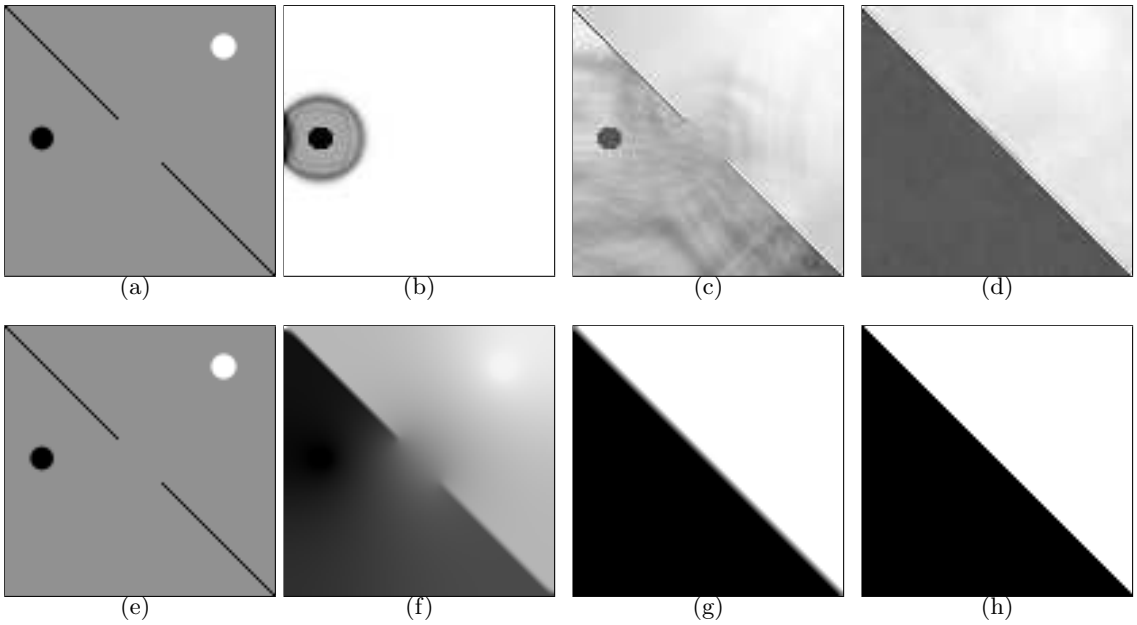


Figure 2.7: Segmentation of an artificial image with AT-CMF (top row) and CCMF (bottom row).
Top row (AT-CMF): (a) Image where the black and white discs are seeds. AT-CMF result stopped in (b) after 100 iterations, (c) 1000 iterations, (d) 10000 iterations. Bottom row (CCMF): (e) Image where the black and white discs are seeds, CCMF result ν after (f) 1 iterations, (g) ν after 15 iterations iterations and (h) threshold of the final ν.

'GrabCut' database available online [188], which is composed of fifty images provided with seeds. From the seed images provided, it is possible to extract two different possible markers for the background seed. Examples of such seeds are shown in Fig. 2.8(a). Different convergence criteria are available for CCMF, such as the duality gap and norms of the residuals. However, for the CMF algorithm, we do not have any satisfying criteria. Bounding the number of non binary occurrences of P does not mean that the convergence is reached, because of possible oscillations. An intermediate result after ten thousand iterations may be significantly different from the result reached after one hundred thousand. Consequently, we have run the AT-CMF algorithm until we were convinced to have reached convergence, *i.e.* when P was nearly binary and did not change significantly when we doubled the number of iterations. For half of the images in the GrabCut database, AT-CMF algorithm required more than 20000 iterations to reach convergence, for a third of the images, more than 80000 iterations, and for 1/4 of the images, more than 160000 iterations. Binary convergence was still not reached after even 500000 iterations for the rest of the images (1/4), but we stopped the computation anyway. The Chambolle-Pock TV based algorithm for binary segmentation is provably convergent and benefits also from several possible convergence criteria. The con-

	Dice Coeff.	GC	AT-CMF	CP-TV	CCMF
First set of seeds	mean	95.2	94.9	95.2	95.3
	median	96.2	96.1	96.3	96.3
	stand. dev.	4.3	4.1	4.1	4.1
Second set of seeds	mean	89.3	89.7	89.1	89.5
	median	91.2	92.0	91.7	92.3
	stand. dev.	8.4	7.7	8.5	8.6

Table 2.2: Dice coefficient (percentage) computed between the segmentation mask and the ground truth image (provided by GrabCut database).
The lines above the double bar show results for the first set of seeds, and the lines below the double bar show results obtained with the second set of seeds.

vergence criteria used is a ratio of changed pixels from two successive intervals of iterations being smaller than an epsilon that we fixed to 10^{-7} in order to obtain satisfying results. In practice, 1/4 of the images converged in less than 60000 iterations and for the rest of them we increased the maximum number of iterations up to 200000. In contrast, CCMF only needed 21 iterations on average, and never more than 27, to reach the convergence criteria $||r_d|| < 1$ and $||\widehat{\eta}|| < 2$. We noted that for all methods, segmentation quality degraded quickly if these conditions were not respected.

Table 2.2 displays the performance results for these algorithms. The Dice coefficient is a similarity measure between sets (segmentation and ground truth), ranging from 0 to 1.0 for no match and a perfect match respectively. All the tested algorithms show very good results, with a Dice coefficient of 0.95–0.97 for the well positioned seeds, and 0.89–0.92 for the second set of seeds, further away from the objects. The CCMF and AT-CMF results are really close, and better in mean than the GC and the CP-TV results.

2.4.4 Extensions

The primary focus of our experiments were to demonstrate that our CCMF algorithm achieves a solution which does not exhibit metrication artifacts, that it is fast with provable convergence and that the segmentation quality is high. In the remainder of this section, we show that the algorithm can also incorporate unary terms and be equally applied in 3D. In fact, since CCMF is defined for an arbitrary graph, it could even be used to perform clustering in this more generalized framework.

2.4.4.1 Unary terms

A simple modification of the transport graph G permits the use of unary terms to automatically specify objects to be segmented. This approach is similar to the use of unary terms

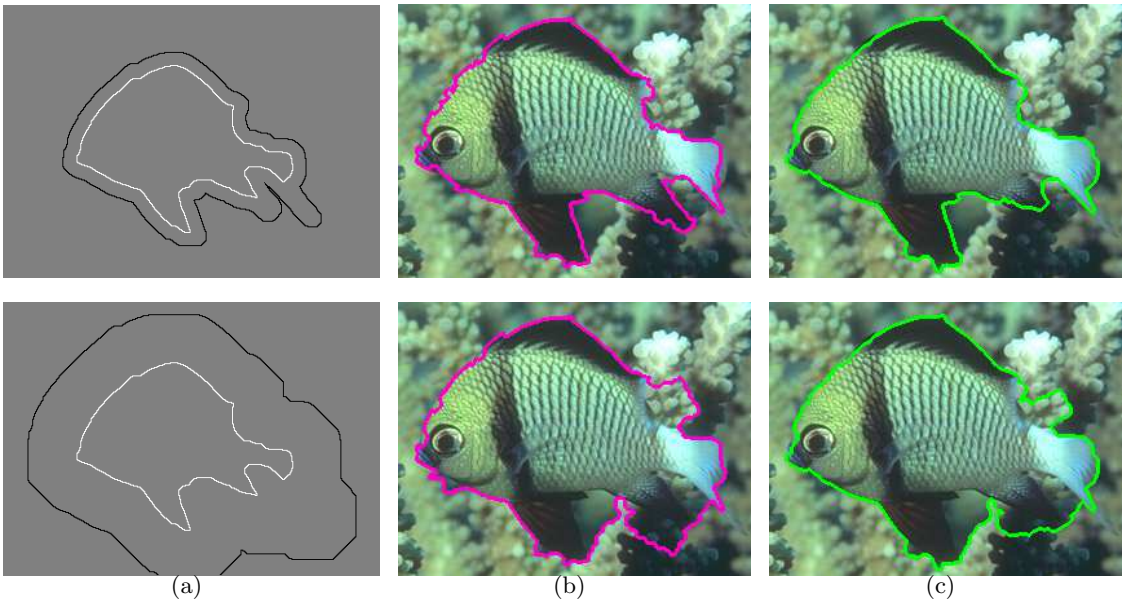


Figure 2.8: GC and CCMF segmentation results on a image from the GrabCut database. (a) : Seed images belonging to the first (top) and second (bottom) set of seeds provided by the GrabCut database. (b) Segmentations by Graph Cuts computed with seeds from the two sets, (c) Segmentations by CCMF.

in the graph cuts computation [32]. The placement of unary terms for adding data fidelity constraints in the max-flow problem is performed by adding edges linking every node of the lattice to the source and to the sink. In the case of the CCMF problem, as weights are defined on the nodes, we need to add intermediary nodes between the grid and source on the one hand, and the grid and sink on the other. However, since these intermediary nodes have just two edges incident on them, these node weights are equivalent to edge weights, meaning that our construction is equivalent to the use of unary terms for AT-CMF that was pursued by Unger *et al.* [209]. In our case, we used weighted intermediary nodes simply to keep the CCMF framework consistent. Considering that the original lattice is composed of n nodes, we add for each node an "upper" node linked to the source and a "lower" node linked to the sink. An example of this construction is shown in Fig. 2.9.

The weights of the additional nodes may be set to reflect the node priors for a particular application. For image segmentation, given mean background and foreground color BC and FC, we can set the capacities of the background nodes to $g_i = \exp(-\beta(BC_i - I_i)^2)$ and foreground nodes to $g_i = \exp(-\beta(FC_i - I_i)^2)$. Examples of results using these appearance priors are shown on Fig. 2.9.

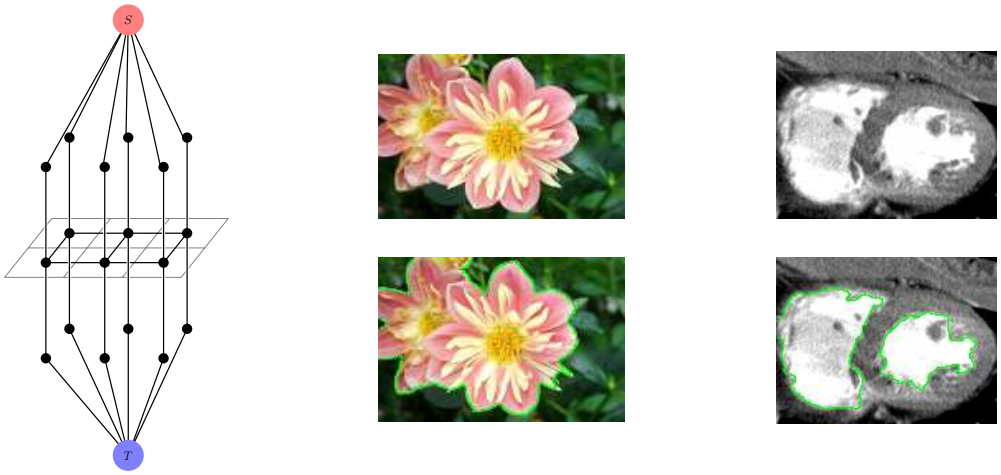
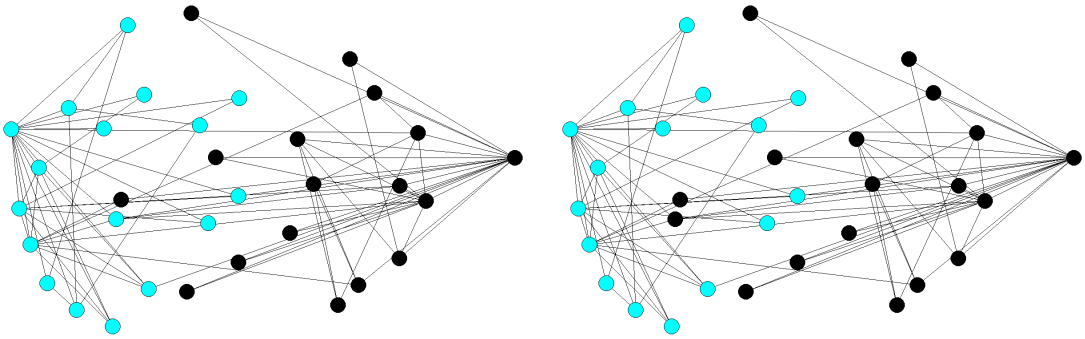


Figure 2.9: Example of graph construction to incorporate unary terms for unsupervised segmentation.
Each node is connected to a source node and sink node through an intermediary node which is weighted to reflect the strength of the positive and negative unary term. This construction is then applied with a simple appearance prior to automatically segment two images.

2.4.4.2 Classification

As the general CCMF formulation is defined on arbitrary graphs, it is possible to employ it in tasks beyond image segmentation. For instance CCMF may be employed to solve general problems of graph clustering, even on networks with no meaningful embedding such as social networks. We show a possible application in a classification example, considering the social network studied by Zachary [224]. After the split of a university karate club due to a conflict between its two leaders, Zachary's goal was to study if it was possible to predict the side joined by the members, based only on the social structure of the club. Classically, the graph is built by associating each member to a node, and edges link two members when special affinities are known between them. As weights are associated here to the strength of coupling between members, different strategies are possible for assigning the weights onto the nodes, which is necessary for using CCMF. For example, a given node/member may be assigned by the mean of affinity with its neighboring members in the graph. This weighting strategy works well in this example (See Fig. 2.10). Another strategy would be to compute the solution in the dual graph, that is in the graph where nodes are replaced by edges and (weighted) edges are replaced by (weighted) nodes. We have shown in an example the ability of CCMF for classification, a task that can not be treated with finite element-based methods such as AT-CMF. Evaluating the performances of CCMF for classification may be a topic for future research.



(a) Network and true membership after split (b) Classification using CCMF

Figure 2.10: Zachary's karate club network and CCMF classification result.
The two leaders in conflict are represented by the left hand side and right hand side nodes. Just one node is misclassified with CCMF. However, this node is known classically to be an unusual situation within the social network (see [224]) which is not captured by any known algorithm.

2.4.4.3 3D segmentation

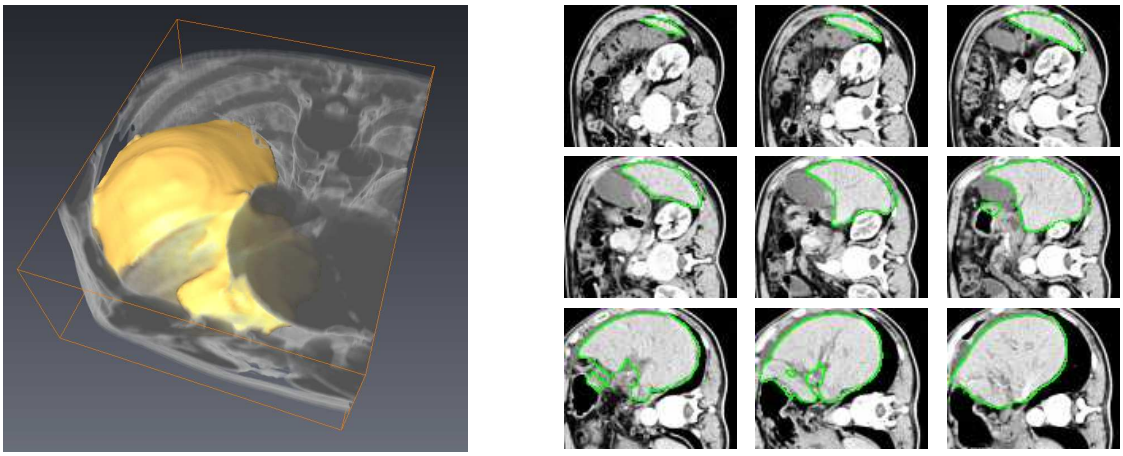


Figure 2.11: Liver segmentation in 3D by CCMF.

For 3D image segmentation, the minimal surface properties of CCMF generate good quality results, as shown in Fig 2.11. The CCMF formulation applies equally well in 2D or 3D, since CCMF is formulated on an arbitrary graph (which may be a 2D lattice, a 3D lattice or an even more abstract graph). In 3D, our CCMF implementation is suffering from memory limitations in the direct solver we used, limiting its performances. Future work will address this issue using a dedicated solver.

2.5 Conclusion

In this chapter, we have presented a new combinatorial formulation of the continuous maximum flow problem and a solution using an interior point convex primal-dual algorithm. This formulation allows us to optimize the maximum flow problem as well as its dual for which we provide an interpretation as a minimal surface problem. This new combinatorial expression of continuous max-flow avoids blockiness artifacts compared to graph cuts. Furthermore, the formulation of CMF on a graph reveals that it is actually the fact that the capacity constraints are applied to point-wise norms of flow *through nodes* that allows us to avoid metrication errors, as opposed to the conventional graph cut capacity constraint *through individual edges*. Additionally, it was shown that our CCMF formulation provides a better approximation to the analytic catenoid than the conventional AT-CMF discretization of the continuous max-flow problem. Contrary to graph cuts, we showed that CCMF estimates the true boundary length of objects. Finally, unlike finite-element based methods such as AT-CMF, the CCMF formulation is expressed for arbitrary graphs, and thus can be employed in a large variety of tasks such as classification.

We also provide in this chapter an exact analytic expression of the dual problem, convergence of the algorithm being guaranteed by the convexity of the problem. In terms of speed, when an approximate solution is sufficient, our implementation of CCMF in Matlab is competitive to the Appleton-Talbot approach, which uses a system of PDEs, and a `C++` implementation. The Appleton-Talbot algorithm has the significant drawback of not providing a criterion for convergence. In practice, this translates to long computation times when convergence is difficult for the AT-CMF algorithm. In contrast, our algorithm in this case is much faster, as we observe that the number of iterations required for convergence does not vary much for CCMF (less than a factor of two). The CCMF algorithm is simple to implement, and may be applied to arbitrary graphs. Furthermore, it is straightforward to add unary terms to perform unsupervised segmentation.

We have also compared our algorithm to known weighted combinatorial TV minimization methods. We have shown that our results are generally better for segmentation than combinatorial TV-based methods, and that our implementation on CPU is much faster and more predictable. Specifically, the number of iterations required for convergence does not vary much for CCMF, whereas it can vary of more than a factor of ten for CP-TV. The deep study of the relationships between CCMF and combinatorial TV reveals that, in contrast with expectations from their duality under restrictive assumptions in the continuous domain, their duality relationship is only weak in the combinatorial setting. In fact max-flow and total variation are different problems with different constraints, yielding different algorithms and different results. One key difference between max-flow and total variation is that max-flow algorithms were developed as segmentation algorithms, following the graph cuts framework.

One consequence is that max-flow formulations have a null divergence objective, which is not present in total variation formulations. Null divergence can be viewed as a consequence or a necessity in order to obtain constant partitions almost everywhere, in particular binary ones. This difference is important because in the proposed CCMF framework we *impose* a tight null divergence constraint, which to our knowledge is novel for an isotropic formulation (graph-cuts have always had this constraint). AT-CMF and similar frameworks achieve null divergence if and when convergence is achieved. There is no null divergence constraint at all in total variation frameworks. As a consequence, while in practice it is possible to compare total variation and max flow formulations, they should be treated differently. However, the strong computational performance and segmentation quality results of CCMF as compared to TV (using the strongest known optimizations methods) suggests that it may be advantageous to apply our CCMF formulation to problems for which TV has proven effective (such as filtering).

Several further optimizations of CCMFs are possible, for instance using multi-grid implementations, the possibility to use GPU to solve the iterative linear system, the use of a dedicated solver for the particular sparse linear system involved in the computation. We also plan to compare the efficiency obtained with the interior point method to a first order algorithm for solving the CCMF problem.

Ultimately, we hope to employ combinatorial continuous maximum flow as a powerful segmentation algorithm which avoids metrication artifacts and provides a fast solution with provable convergence. Furthermore, we intend to explore the potential of CCMF to optimize other energy functions for which graph cuts or total variation have proved useful, such as surface reconstruction or efficient convex filtering.

In the next chapter, inspired by the CCMF formulation, we propose a combinatorial formulation for the weighted Total Variation primal dual formulation. In contrast to CCMF, the new TV-based model is adapted to multi-label problems.

Chapter 3

Dual Constrained TV-based regularization framework

Total Variation minimization based algorithms are prevalent in computer vision. They are used in a variety of applications such as image denoising, compressive sensing and inverse problems in general. In this work [1], we extend the TV dual framework that includes Chambolle's and Gilboa-Osher's projection algorithms for TV minimization. We use a flexible graph data representation that allows us to generalize the constraint on the projection variable. We show how this new formulation of the TV problem may be solved by means of fast parallel proximal algorithms. We show on denoising and deblurring examples that the proposed approach performs better than classical TV. We also show that the proposed method is applicable to a variety of other inverse problems including image fusion and mesh filtering.

3.1 Introduction

The Total Variation (TV) model was introduced as a regularization criterion for image denoising by Rudin, Osher and Fatemi (ROF) [189], and has been shown to be quite efficient for smoothing images while preserving contours. Total variation minimization is used for solving a large number of inverse problems, including deconvolution [66], inpainting, MRI reconstruction, optical flow estimation [15], stereovision [181], among others. A major advantage of the ROF-TV minimization problem is that it is a convex problem. However, the non-differentiability of the involved objective function requires the use of specific optimization techniques, particularly with respect to speed and efficiency. Much progress has been achieved by employing primal-dual approaches [48, 55], the dual approach of Chambolle [50], and fast first order methods [13].

[1] This chapter was published in part in *Proc. of ICASSP 2011* [74]

Qualitative improvements have also been obtained by introducing a weighted model [29, 89, 102, 103]. In this model, a discrete TV energy is optimized in edge-weighted graphs. The methods described in [102] and [103] employ a non-local weighting strategy following the non-local means concept [47]. Furthermore, Gilboa and Osher provided an efficient dual algorithm in [103], based on Chambolle's [50] method to address this problem. Currently, one of the fastest methods for optimizing weighted TV is an alternating direction of multipliers method [108, 225], related to augmented Lagrangian approaches [3], and also called split-Bregman. Another (node) weighting strategy, and a fast primal dual algorithm, is proposed in [54].

Recently, several approaches were proposed to reduce the staircaising effect of total variations methods. The Total Generalized Variation method of Bredies *et al.* [39, 40] employs higher order derivatives of the function to regularize. The benefit is to replace the staircase effect by smoothed variations, while preserving contours. The work of Jalalzai and Chambolle aims at achieving the same goal [130]. In the hybrid TV approach of Combettes and Pesquet [66](See [184] for further investigations) an energy combining wavelets and TV is optimized to jointly reduce the staircase effect of both TV and wavelet artifacts.

In addition to the staircase effect, TV approaches typically result in a loss of contrast. A few method have been proposed [59, 81] for contrast preservation.

The projection algorithms in [50, 103] are based on a relatively simple local constraint on the norm of the projection variable. In this work, we extend the constraint on this variable. This extension allows us to better adapt the optimization procedure to the local information. We name this new approach "Dual-Constrained Total Variation" (DCTV) regularization.

Our main contributions are the following:

1. By using an appropriate reformulation and decomposition of our energy functional, we are able to optimize it efficiently, owing to the use of fast parallel proximal algorithms.

2. By formulating the primal dual total variation problem on arbitrary graphs, we are able to use our regularization method on graphs that are not uniformly sampled, for example on a graph representations of the fovea or an arbitrary mesh.

3. By providing a formulation taking into account weights both on nodes and edges of the graph, we obtain a more flexible regularization.

4. These improvements lead to better results, in particular a better contrast preservation and a more detailed recovery in image processing examples.

Proximal methods provide efficient solutions to convex optimization problems where the objective function can be split into a sum of convex functions [67, 132]. This core of methods are applicable to large size problems, as well as to non-smooth and non-finite functions.

Note that the constrained TV-based energy optimized in this work is different than the one considered in previous works [64, 154]. Combettes and Pesquet [64] introduce a total variation bound as a constraint in a convex energy minimization problem. Malgouyres [154] defines a constraint in order to combine the ROF-TV model with wavelet-based soft thresholding methods.

The chapter is organized as follows: in Section 3.2, we generalize TV models on a graph by extending the dual formulation under a constrained form. In Section 3.3, we demonstrate how our constrained TV-based optimization problem can be efficiently solved by using proximal algorithms. Finally, results obtained with the proposed approach are presented in four applications and compared with classical weighted TV.

3.2 Graph extension of TV models

Given an original corrupted image f, the purpose of variational methods for image restoration is to deduce a restored image u close to the original image f under the assumption of smooth variations of intensity values inside objects. Let $\lambda \in]0, +\infty[$ be a real positive value representing a regularization parameter. In a functional setting, given a planar domain Ω, and denoting by u and v two arbitrary points of Ω, the weighted TV model [103] is given by

$$\min_x \int_\Omega \Big(\int_\Omega w_{u,u'}(x_{u'} - x_u)^2 du' \Big)^{1/2} du + \frac{1}{2\lambda} \int_\Omega (x_u - f_u)^2 du, \tag{3.1}$$

where w is a non-negative valued function defined on Ω^2. As shown by Chan *et al.* in [55] the TV minimization problem (3.1) is equivalent to the min-max problem

$$\min_x \Big(\max_{||p||_\infty \leq 1} \int_{\Omega^2} w_{u,u'}^{1/2}(x_{u'} - x_u) p_{u,u'} du\, du' + \frac{1}{2\lambda} \int_\Omega (x_u - f_u)^2 du, \Big) \tag{3.2}$$

where p a two-variable function and

$$\|p\|_\infty = \sup_{u \in \Omega} \Big(\int_\Omega p_{u,u'}^2 du' \Big)^{1/2}.$$

Let us now establish the formulation of the DCTV energy within a discrete framework, using the notations specifyied in the introduction chapter.

Now, $f = (f_i)_{1 \leq i \leq n}$ and $x = (x_i)_{1 \leq i \leq n}$ are vectors in $\mathbb{R}^n$. The combinatorial TV model as defined by [29, 103] is

$$\min_{x \in \mathbb{R}^n} \sum_{i \in J} \Big(\sum_{j \in N_i} w_{i,j}(x_j - x_i)^2 \Big)^{1/2} + \frac{1}{2\lambda} \sum_{i=1}^{n} (x_i - f_i)^2 \tag{3.3}$$

where, for every $i \in \{1,\dots,n\}$, $N_i = \{j \in \{1,\dots,n\} \mid e_{i,j} \in E\}$, $J = \{i \in \{1,\dots,n\} \mid N_i \neq \varnothing\}$. The dual formulation, which can be optimized by a projection algorithm [103], is given by

$$\min_{x\in\mathbb{R}^n} \Big(\max_{\|p\|_\infty \leq 1} p^\top \big((Ax)\cdot\sqrt{w}\big) + \frac{1}{2\lambda}\|x-f\|^2 \Big), \tag{3.4}$$

where $\|\cdot\|$ denotes the Euclidean norm, $w \in \mathbb{R}^m$ is a vector with components $w_{i,j}$ for every couple of nodes (i,j) with $i \in J$ and $j \in N_i$ and similarly denoting by $p_{i,j}$ the components of $p \in \mathbb{R}^m$,

$$\|p\|_\infty = \max_{i\in J} \Big(\sum_{j\in N_i} p_{i,j}^2 \Big)^{1/2}. \tag{3.5}$$

By introducing a vector $F \in \mathbb{R}^m$ with components $F_{i,j}$ such that, for every $i \in J$ and $j \in N_i$, $F_{i,j} = p_{i,j}\sqrt{w_{i,j}}$, the problem can be reformulated as

$$\min_{x\in\mathbb{R}^n} \Big(\max_{F\in C} F^\top Ax + \frac{1}{2\lambda}\|x-f\|^2 \Big), \tag{3.6}$$

$$C = \{(F \in \mathbb{R}^m \mid (\forall i \in J) \sum_{j\in N_i} \frac{F_{i,j}^2}{w_{i,j}} \leq 1\}. \tag{3.7}$$

The objective of this work is to extend the discrete weighted TV variational formulation in (3.6) by investigating the following optimization problem

$$\min_{x\in\mathbb{R}^n} \Big(\sup_{F\in C} F^\top Ax + \frac{1}{2}(Hx-f)^\top \Lambda^{-1}(Hx-f) \Big) \tag{3.8}$$

where $f \in \mathbb{R}^q$ is an observed vector of data, $H \in \mathbb{R}^{q\times n}$ and Λ is a weighting symmetric definite-positive matrix in $\mathbb{R}^{q\times q}$. The metrix H may simply be the identity matrix for image denoising, a convolution operator in restoration tasks, or a projection matrix in reconstruction problems. The matrix Λ may be proportional to the covariance matrix of the noise corrupting the data, as commonly used in weighted least squares approaches. In addition, the main contribution of this work is to consider more general convex sets C than those given by (3.7).

3.2.1 Considered class of constraint sets

The proposed optimization approach will allow us to address nonempty convex sets C that can be decomposed as an intersection of closed convex subsets $(C_r)_{1\leq r\leq s}$ of $\mathbb{R}^m$, the projections

onto which take closed forms. An example of a set C of interest is given by

$$\begin{aligned} &C = \bigcap_{r=1}^{s} C_r \\ &(\forall r \in \{1,\dots,s\}) \qquad C_r = \{F \in \mathbb{R}^m \mid (\forall i \in S_r)\ \|\theta^{(i)} \cdot F\|_\alpha \leq g_i\}. \end{aligned} \tag{3.9}$$

where $(S_r)_{1\leq r\leq s}$ is a partition of $\{1,\dots,n\}$, $\|\cdot\|_\alpha$ is the ℓ_α norm of $\mathbb{R}^m$ with $\alpha \in [1,+\infty]$ and, for every $i \in \{1,\dots,n\}$, $\theta^{(i)} \in]0,+\infty[^m$ is a vector of multiplicative constants for every couple of node (i',j) with $i' \in J$ and $j \in N_{i'}$.

The form of C in (3.9) obviously includes (3.7) as a particular case where $\alpha = 2$ and

$$\begin{aligned} &(\forall i \in \{1,\dots,n\}) \quad (\forall i' \in J)(\forall j \in N_{i'})\ \ \theta^{(i)}_{i',j} = \begin{cases} \frac{1}{\sqrt{w_{i',j}}} & \text{if } i' = i \text{ and } j \in N_{i'} \\ 0 & \text{otherwise} \end{cases} \\ &\qquad\qquad g_i = 1. \end{aligned} \tag{3.10}$$

However, the proposed approach offers much more flexibility.

In this chapter, we are mainly interested in the case when for every $i \in \{1,\dots,n\}$, $\theta^{(i)}$ is the i-th line vector of $|A^\top|$. Specifically, C may be defined as

$$C = \{F \in \mathbb{R}^m \quad | \quad g^{\cdot 2} - |A^\top| F^{\cdot 2} \in [0,+\infty[^n\}, \tag{3.11}$$

where $g = (g_i)_{1\leq i\leq n}$ and, for every vector a, $a^{\cdot 2} = a \cdot a$. The constraint given in the convex set of (3.11) also appears in the Combinatorial Continuous Maximum Flow (CCMF) problem of Chapter 2, [73]. The problem studied in the present work may be seen as an extension of the CCMF problem (applied to clustering problems in graphs, such as image segmentation) to multi-label problems.

Concerning the choice of the node weight g_i at the i-th node, a simple strategy consists of considering a monotonically decreasing function of the data gradient. More specifically, given positive reals ϵ and χ, we suggest using g defined as

$$(\forall i \in \{1,\dots,n\}) \qquad g_i = \exp(-\chi \|\nabla \overline{x}_i\|_2) + \epsilon, \tag{3.12}$$

where $\overline{x}$ is some reference data defined on a graph, for instance image data, which corresponds to some rough estimate of x, and

$$\|\nabla \overline{x}_i\|_2 = \begin{cases} (\sum_{j\in N_i} (\overline{x}_i - \overline{x}_j)^2)^{\frac{1}{2}} & \text{if } i \in J \\ 0 & \text{otherwise} \end{cases} \tag{3.13}$$

corresponds to the Euclidean norm of its discrete gradient $\nabla \overline{x}_i$ at node i. In the absence

of a contour, g_i takes large values, so are the components of F corresponding to nonzero values of $\theta^{(i)}$, preventing large local variations of x in the minimization (3.8). Conversely, in the presence of a contour, $g_i \simeq \epsilon$, and the components of F corresponding to nonzero values of $\theta^{(i)}$ are small, thus allowing large local variations of x. In image filtering applications, in addition to intensities information, g may be used to penalize changes in other relevant image quantities such as color or texture.

3.3 Proposed algorithms

We show in this section that it is possible to efficiently solve Problem (3.8) by proximal methods [67]. To do so, define the support function σ_C of the closed convex constraint set C as

$$\sigma_C \colon \mathbb{R}^m \to]-\infty, +\infty] \colon a \mapsto \sup_{F \in C} F^\top a. \tag{3.14}$$

This is a proper lower-semicontinuous convex function, the conjuguate of which is the indicator function of C,

$$\imath_C \colon F \mapsto \begin{cases} 0 & \text{if } F \in C, \\ +\infty & \text{otherwise.} \end{cases} \tag{3.15}$$

This leads us to consider the following optimization problem:

$$\min_x \sigma_C(Ax) + \frac{1}{2}(Hx - f)^\top \Lambda^{-1}(Hx - f) + \frac{\eta}{2}\|Kx\|^2 \tag{3.16}$$

where $\eta \in]0, +\infty[$ and $K \in \mathbb{R}^{n \times n}$ is the projection matrix onto the nullspace of H, specifically, $K = I - H^\top (HH^\top)^{-1} H$. When H is injective ($\operatorname{rank} H = n$), the last term vanishes and (3.16) is strictly equivalent to (3.8). The term $x \mapsto \eta\|Kx\|^2/2$ thus aims at introducing an additional regularization when H is not injective, so that the objective function remains strictly convex. The following holds:

Property 2. *Problem (3.16) admits a unique solution $\widehat{x}$. The dual Fenchel-Rockafellar form of the problem is*

$$\min_F \varphi(F) + \imath_C(F), \tag{3.17}$$

where $\varphi \colon F \mapsto \frac{1}{2} F^\top A \Gamma A^\top F - F^\top A \Gamma H^\top \Lambda^{-1} f$ and $\Gamma = (H^\top \Lambda^{-1} H + \eta K)^{-1}$. The optimal solution to the primal problem (3.16) is deduced from any optimal solution $\widehat{F}$ of the dual problem by the relation

$$\widehat{x} = \Gamma \left(H^\top \Lambda^{-1} f - A^\top \widehat{F} \right). \tag{3.18}$$

Note that the dual forward-backward algorithm proposed in [63] is not applicable to Problem 3.17 since the projection onto C is not explicit in general. Dykstra's algorithm [37] to

compute exact projections on an intersection of convex sets would not be applicable either, as the operator $A\Lambda A^\top$ is singular. In order to numerically solve (3.17), recall that $C = \bigcap_{r=1}^{s} C_r$, so that $\imath_C$ can be decomposed into the sum of the indicator functions of the convex subsets $(C_r)_{1\leq r\leq s}$. Hence, the problem is equivalent to solving

$$\min_{F\in\mathbb{R}^m} \sum_{r=1}^{s} \imath_{C_r}(F) + \varphi(F). \tag{3.19}$$

3.3.1 Parallel proximal algorithm (PPXA)

The above sum of $(s+1)$ convex functions can be efficiently optimized by resorting to the Parallel ProXimal Algorithm (PPXA) proposed in [66]. PPXA was generalized in [177], where links with augmented Lagrangian methods [3] were established.

As shown in Algorithm 5, this requires computing in parallel projections onto each set C_r with $r \in \{1,\ldots,s\}$, which are defined $\forall F \in \mathbb{R}^m$ as $\mathsf{P}_{C_r}(F) = \arg\min_{\Phi\in C_r} \|\Phi - F\|$. Note that the convergence of the sequence $(F_k)_k$ generated by this algorithm to a solution $\widehat{F}$ of (3.17) is guaranteed, which allows us to deduce a solution to (3.16) by using Relation (3.18).

Algorithm 5: Parallel proximal algorithm solving (3.19)

Fix $\gamma > 0$ and $\nu \in]0,2[$. Set $k = 0$.
Choose $y_{1,0} = y_{2,0} = \ldots = y_{s+1,0} \in \mathbb{R}^m$ and F_0.
repeat
  for $r = 1,\ldots,s+1$ **do in parallel**
    $\pi_{r,k} = \begin{cases} \mathsf{P}_{C_r}(y_{r,k}) & \text{if } r \leq s \\ (\gamma A\Gamma A^\top + I)^{-1}(\gamma A\Gamma H^\top \Lambda^{-1} f + y_{s+1,k}) & \text{otherwise} \end{cases}$
  $z_k = \frac{2}{s+1}(\pi_{1,k} + \cdots + \pi_{s+1,k}) - F_k$
  for $r = 1,\ldots,s+1$ **do in parallel**
    $y_{r,k+1} = y_{r,k} + \nu(z_k - p_{r,k})$
  $F_{k+1} = F_k + \frac{\nu}{2}(z_k - F_k)$
  $k = k+1$
until *convergence*

Let us now turn our attention to convex sets as described in (3.9). The main difficulty for applying PPXA is to find a partition $(S_r)_{1\leq r\leq s}$ for which the projections $(\mathsf{P}_{C_r})_{1\leq r\leq s}$ can be computed easily. To ensure this property, the partition can be defined in such a way that

$$(\forall r \in \{1,\ldots,s\})(\forall (i,i') \in S_r^2) \qquad i \neq i' \Rightarrow \#(\theta^{(i)} \cdot \theta^{(i')}) = 0, \tag{3.20}$$

where $\#(a)$ denotes the number of nonzero components in a vector a. Then, it is easy to see that the projection onto C_r with $r \in \{1,\ldots,s\}$ of a vector F can be decomposed as a finite number of projections of subvectors of dimensions $\left(\#(\theta^{(i)})\right)_{i\in S_r}$ extracted from F onto

weighted ℓ_α balls.

For a 4-connected lattice, when $(\forall i \in \{1, \dots, n\})$ $\theta^{(i)}$ is the i-th line of matrix $|A^\top|$, a decomposition of C can be performed by setting $s = 2$. The partition $(S_r)_{1 \leq r \leq 2}$ corresponds to two spatial disjoint sets each with a checkered pattern (also called red black checkerboard pattern). The projection onto C_r with $r \in \{1, 2\}$ for $\alpha = 2$ (resp. $\alpha = 1$ or $\alpha = +\infty$) reduces to simple projections onto hyperspheres (resp. hypercubes [211]).

For example, the decomposition on a small lattice when $\alpha = 2$ is illustrated by Figure 3.1.

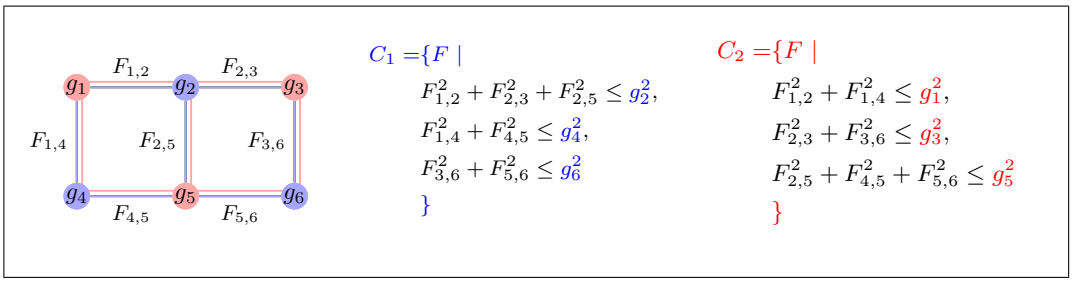
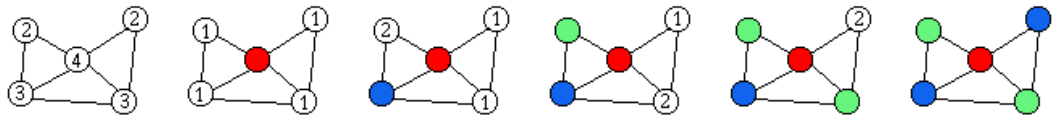


Figure 3.1: Decomposition of $C = C_1 \cap C_2$

Note also that, at each iteration $k \in \mathbb{N}$, the computation of $\pi_{s+1,k}$ requires a matrix inversion. In cases where H and Λ are (or can be well approximated by) circulant-block circulant matrices, and the graph is regular, this step can be performed efficiently [84, 96]. For this purpose, an equivalent formulation of matrix $(\gamma A \Gamma A^\top + I)^{-1}$ is computed using the matrix inversion lemma, also known as Woodbury's matrix identity. The new resulting linear system is then composed of a circulant block circulant matrix and is solvable efficiently using a Fast Fourier Transform.

Decomposing arbitrary graphs

In arbitrary graphs, the graph decomposition of Figure 3.1 in a checkerboard pattern is no longer valid, so we use a graph coloring algorithm for the decomposition. Coloring a graph with a minimum number of colors for the nodes is an NP-complete problem. However, there exist heuristic algorithms that find a non-optimal, but feasible solution in polynomial time. In our experiments, we employed the DSAT (Degree SATuration) algorithm [43], which is guaranteed to color triangulated graphs in linear time. An illustration is provided in Figure 3.2. Experimentally, this greedy algorithm typically decomposes of the convex set C in five sets or fewer.



(a) (b) (c) (d) (e) (f)

Figure 3.2: Coloring a graph using the DSAT algorithm [43].
(a) Sort nodes by decreasing degree. Initial labeling DSAT equal to node degrees. (b) Choose a node with maximum DSAT. Assign color number 1. Update DSAT: For each non-colored node v, if at least one neighbor of v is colored, DSAT(v) equals the number of different colors used in the neighborhood of v. (c) Choose a non-colored node of maximum DSAT. In case of conflict, choose a node of maximum degree. Assign the smallest possible color. Update DSAT. (d,e,f) Repeat the procedure in (c) until all nodes are colored.

3.3.2 M+SFBF algorithm

Starting from Problem (3.16), we deduce an alternative formulation of the energy to minimize. The formulation we obtain shall allow us to employ another parallel proximal algorithm, the Monotone + Skew Forward Backward Forward (M+SFBF) algorithm, which differs from PPXA. Interestingly, M+SFBF [46] does not require us to solve a linear system at each iteration of the algorithm. M+SFBF belongs to the rich class of primal-dual proximal methods which also include the methods by Chen and Teboulle [58], Chambolle and Pock [54] and Esser *et al.* [90].

Property 3. *Problem* (3.16) *admits the equivalent formulation*

$$\min_{\substack{x\in\mathbb{R}^n, a_1\in\mathbb{R}^m,\dots,a_s\in\mathbb{R}^m \\ a_1+\cdots+a_s=Ax}} \sum_{r=1}^{s} \sigma_{C_r}(a_r) + \frac{1}{2}(Hx-f)^\top \Lambda^{-1}(Hx-f) + \frac{\eta}{2}\|Kx\|^2. \tag{3.21}$$

Proof. Noticing that

$$\sigma_C^* = \iota_C = \sum_{r=1}^{s} \iota_{C_r}, \tag{3.22}$$

we deduce that

$$\sigma_C = \iota_{C_1}^* \square \cdots \square \iota_{C_s}^* = \sigma_{C_1} \square \cdots \square \sigma_{C_s} \tag{3.23}$$

where $\square$ denotes the infimal convolution. The infimal convolution of two convex functions φ and ψ from $\mathbb{R}^m$ to $]-\infty,+\infty]$ is the convex function defined as

$$(\forall a \in \mathbb{R}^m) \quad (\varphi \square \psi)(a) = \inf_{a'\in\mathbb{R}^m} \varphi(a') + \psi(a-a'). \tag{3.24}$$

We have then

$$(\forall a \in \mathbb{R}^m) \quad \sigma_C(a) = \inf_{\substack{a_1 \in \mathbb{R}^m, \dots a_s \in \mathbb{R}^m \\ a_1 + \cdots + a_s = a}} \sigma_{C_1}(a_1) + \cdots + \sigma_{C_s}(a_s). \tag{3.25}$$

This allows us to deduce that (3.16) can be reformulated as the following multivariate optimization problem:

$$\min_{\substack{x \in \mathbb{R}^n, a_1 \in \mathbb{R}^m, \dots, a_s \in \mathbb{R}^m \\ a_1 + \cdots + a_s = Ax}} \sum_{r=1}^{s} \sigma_{C_r}(a_r) + \frac{1}{2}(Hx - f)^\top \Lambda^{-1}(Hx - f) + \frac{\eta}{2}\|Kx\|^2.$$

□

Algorithm 6: M+S Forward-backward-forward algorithm solving (3.21)

Fix $\varepsilon \in]0, \big((s+1)\beta + 1\big)^{-1}[$ where $\beta = \max\{1, \|A\|\}$. Set $k = 0$.
Choose $(F_{r,0})_{1 \le r \le s+1}$ and $(a_{r,0})_{1 \le r \le s}$ in $\mathbb{R}^m$ and $x_0 \in \mathbb{R}^n$.
repeat
  $F_k = \frac{1}{s+1} \sum_{r=1}^{s+1} F_{r,k}$
  $\gamma_k \in [\varepsilon(s+1), (1-\varepsilon)/\beta]$
  for $r = 1, \dots, s+1$ **do in parallel**
    $$G_{r,k} = \begin{cases} F_{r,k} - \gamma_k a_{r,k} & \text{if } r \le s \\ F_{s+1,k} + \gamma_k A x_k & \text{otherwise.} \end{cases}$$
    $$b_{r,k} = \begin{cases} a_{r,k} + \gamma_k F_{r,k} & \text{if } r \le s \\ x_k - \gamma_k A^\top F_{s+1,k} & \text{otherwise.} \end{cases}$$
  $\Pi_k = \frac{1}{s+1} \sum_{r=1}^{s+1} G_{r,k}$
  for $r = 1, \dots, s+1$ **do in parallel**
    $$\pi_{r,k} = \begin{cases} b_{r,k} - \gamma_k \mathsf{P}_{C_r}(\gamma_k^{-1} b_{r,k}) & \text{if } r \le s \\ \big(\gamma_k(H^\top \Lambda^{-1} H + \eta K) + I\big)^{-1}(\gamma_k H^\top \Lambda^{-1} f + b_{s+1,k}) & \text{otherwise} \end{cases}$$
    $$G'_{r,k} = \begin{cases} \Pi_k - \gamma_k \pi_{r,k} & \text{if } r \le s \\ \Pi_k + \gamma_k A \pi_{s+1,k} & \text{otherwise.} \end{cases}$$
    $$b'_{r,k} = \begin{cases} \pi_{r,k} + \gamma_k \Pi_k & \text{if } r \le s \\ \pi_{s+1,k} - \gamma_k A^\top \Pi_k & \text{otherwise.} \end{cases}$$
    $F_{r,k+1} = F_{r,k} - G_{r,k} + G'_{r,k}$
    $$\begin{cases} a_{r,k+1} = a_{r,k} - b_{r,k} + b'_{r,k} & \text{if } r \le s \\ x_{k+1} = x_k - b_{s+1,k} + b'_{s+1,k} & \text{otherwise} \end{cases}$$
  $k = k + 1$
until *convergence*

We see that the proposed approach amounts to decomposing the gradient Ax of a candidate solution x in a sum of components $(a_r)_{1 \le r \le s}$ onto which a simpler regularization is

performed. In particular, as in the previous section, we choose the decomposition of C in such a way that each C_r is the product of convex sets in a lower-dimensional space, i.e. $\times_{i\in S_r} C_r^i$ (for example, C_r^i is a hypersphere of $\mathbb{R}^4$). In this case, $\sigma_{C_r^i}$ with $i \in S_r$ is a "local" sparsity measure, in the sense that the proximity operator of each $\sigma_{C_r^i}$ is a proximal thresholder [65]:

$$\text{prox}_{\sigma_{C_r^i}} a_r^i = 0 \Leftrightarrow a_r^i \in C_r^i. \tag{3.26}$$

This interpretation provides more insight in the choice for C.

Another interest of the formulation in (3.21) is that it allows us to propose an alternative parallel algorithm to solve the problem. More precisely, we propose to employ Algorithm 6 which is derived from the M+S Forward-backward-forward algorithm in [46][Prop. 4.4], recalled in Section 1.3.3.2.

One additional advantage of this primal-dual algorithm is that the convergence of the sequence $(a_{1,k}, \ldots, a_{r,k}, x_k)_{k\in\mathbb{N}}$ toward a solution to Problem 3.21 is guaranteed even in the limit case when $\eta = 0$.

3.4 Results

We now demonstrate the behavior of DCTV with respect to different choices of convex sets, weighting strategies, and graph constructions.Due to the generality of the proposed framework, we can employ DCTV in various applications such as image denoising, image deconvolution, image fusion, and mesh filtering.

3.4.1 Image Denoising

3.4.1.1 Local denoising

(a) Noisy image SNR = 16.4 dB | (b) TV SNR =19.1 | (c) Weighted TV SNR = 19.0 | (d) DCTV, $\alpha = \infty$ SNR = 18.8 | (e) DCTV, $\alpha = 2$ SNR = 19.3

Figure 3.3: Denoising an image with constant levels of additive Gaussian noise. (a) Noisy image, (b) TV [50] $\lambda = 0.06$, (c) Weighted TV on a 4-connected graph [89] , weights given by the gradient of the image, (d) PPXA norm ℓ_∞, (e) PPXA norm ℓ_2 ($\lambda = 0.22$, $\epsilon = 0.3$).

In our first application, we demonstrate the performance of DCTV with respect to weighted TV [89] for image denoising, and show that the contrast of images is better preserved using DCTV. In this section 3.4.1.1, the considered graph connects each node corresponding to an image value to its neigbors in a 4-connected local neighborhood.

(a) Original image | (b) Noisy SNR=20.9.dB | (c) Weighted TV SNR=24.8dB | (d) DCTV SNR=25.5dB

Figure 3.4: Denoising the 'man' image (zoom) corrupted with Gaussian noise of variance $\sigma^2 = 10$.

In Figure 3.3, we compare denoising results on a synthetic image, using different TV-based formulations. For a denoising task, where the noise is assumed to be zero-mean Gaussian and white. The general problem (4.1) is optimized using $\eta = 0$, $H = I$, $\Lambda = \lambda I$ with $\lambda \in]0, +\infty[$. The weights are set according to (3.12) with $\chi = 0.04$ for DCTV. The stopping criteria for the algorithms is $\|x_k - x_{k-1}\| < 5.10^{-3}\|x_k\|$. The results obtained with the weighted TV [89] look slightly better than using the classical, unweighted TV model [50]. Performing the DCTV optimization using convex sets with an ℓ_∞ norm results in more artifacts, however an improvement is obtained using ℓ_2-norm based convex sets of (3.11) with PPXA.

In a second experiment, we compare quantitatively DCTV with weighted TV. We used four standard test images that we corrupted with synthetic white zero-mean Gaussian noise with variance σ^2. The value of the image fidelity parameter λ was set according to an empirical rule depending on the variance of the noise, and ϵ was set to λ. Signal to Noise Ratio (SNR) is used as the performance measure in our quantitative evaluation. Table 3.1 reports SNR values for DCTV and weighted TV results obtained on each image corrupted

σ^2	5	10	15	20	25	50	100
SNR values obtained by optimizing weighted TV with Split Bregman							
house	32.5	28.8	27.0	25.7	25.0	21.6	17.3
man	28.4	24.8	23.0	21.7	20.6	17.2	12.8
lena	31.5	27.7	25.9	25.0	24.2	21.2	16.9
barbara	27.1	22.8	20.2	18.5	17.3	15.2	12.8
mean	**29.9**	**26.0**	**24.0**	**22.7**	**21.8**	**18.8**	**14.9**
SNR values obtained by optimizing DCTV							
house	32.6	29.2	27.3	25.7	25.1	21.6	16.9
man	28.9	25.5	23.4	21.9	20.8	17.0	12.6
lena	31.9	28.5	26.5	25.1	24.3	21.1	16.7
barbara	27.1	23.0	20.7	19.0	17.7	15.1	12.5
mean	**30.1**	**26.6**	**24.5**	**22.9**	**22.0**	**18.7**	**14.7**

Table 3.1: Quantitative denoising experiment on standard images corrupted with additive Gaussian noise with variance σ^2. The edges weights were set to The regularization parameter λ was chosen empirically to give the best results for both methods.

with different values of noise variance. Result examples are shown in Figure 3.4. Those experiments show that DCTV leads to improved results when the variance of the noise is lower than 50. Visually, DCTV results are sharper and feature better contrast than the weighted TV results. This may explain the slight degradation of performance in presence of heavy noise (variance ranging from 50 to 100). Geometrically, the improvement of DCTV over the weighted TV can be interpreted in the following way: the convex set we use for projection adapts itself to the local neighborhood, and this avoids oversmoothing as a result. We show, in the experiment of Figure 3.5, an example of the contrast preservation properties of DCTV.

In terms of computation time, DCTV is competitive with the most efficient weighted TV algorithm. Denoising the 512×512 Lena image corrupted with Gaussian noise ($\sigma^2 = 15$) requires 0.38 seconds for split Bregman, versus 0.7 seconds for PPXA to converge on an Intel Xeon 2.5GHz 8-core system.

3.4.1.2 Non local denoising

Rather than using locally connected graphs, non-local strategies [44, 47] have been shown to achieve denoising improvements. A non-local strategy may naturally be employed in the DCTV framework. The weights between non-neighbor nodes are computed following the main idea of [47]: for each pixel p of the image, the squared sum of differences (SSD) between the intensities in a block around p and all other blocks in a large neighborhood around that block is computed. Then, edges are added between p and the nodes producing the best

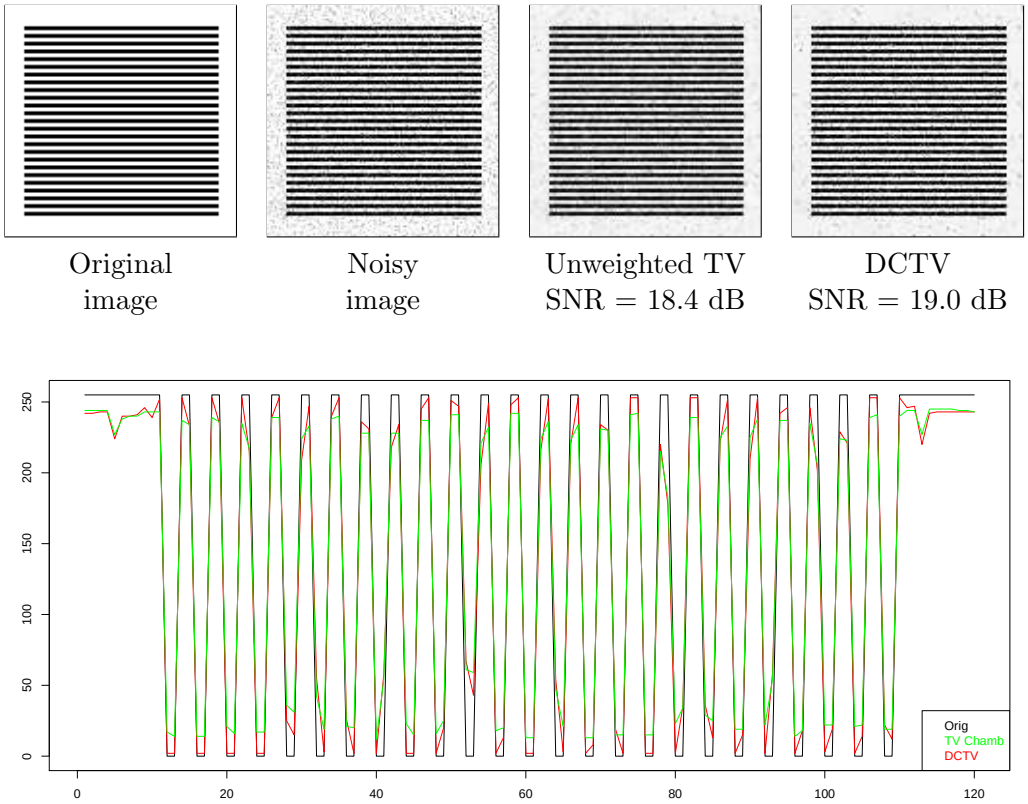
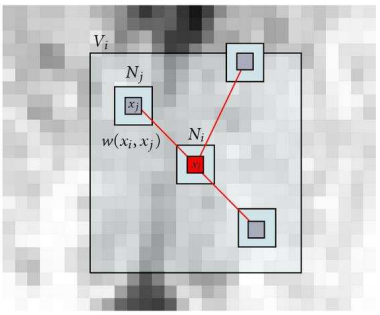


Figure 3.5: Synthetic denoising example demonstrating improved contrast preservation using DCTV.
The noise was AWGN with variance $\sigma^2 = 35$. The graphic shows the profile gray levels on one vertical line of the original image (in black), the TV result (in green), and DCTV result (in red).

matchs, with weights corresponding to the normalized SSD scores. As a result, the Laplacian – related to the matrix to be inverted at each iteration in PPXA – of this non-local graph is no longer circulant block circulant, resulting in loss of efficiency. We propose to employ the M+SFBF algorithm (Algorithm 6) in this case. Figure 3.6 presents a non local DCTV result obtained using this strategy compared with a non local TV result. About 15 iterations are necessary for M+SFBF to produce this result, taking about two seconds using a C parallel implementation on the same 2.5GHz 8-core Xeon system used previously. The projections on the different (9 in the example of Figure 3.6) convex sets are performed in parallel. In comparison, the split Bregman algorithm used for optimizing non local TV only takes four iterations to converge in about one second.

(a) Original image

(b) Noisy SNR=22.6dB



(c) Non-local graph (Figure P. Coupé, [68])

(d) Unweighted TV SNR=25.4dB

(e) Nonlocal TV SNR=26.1dB

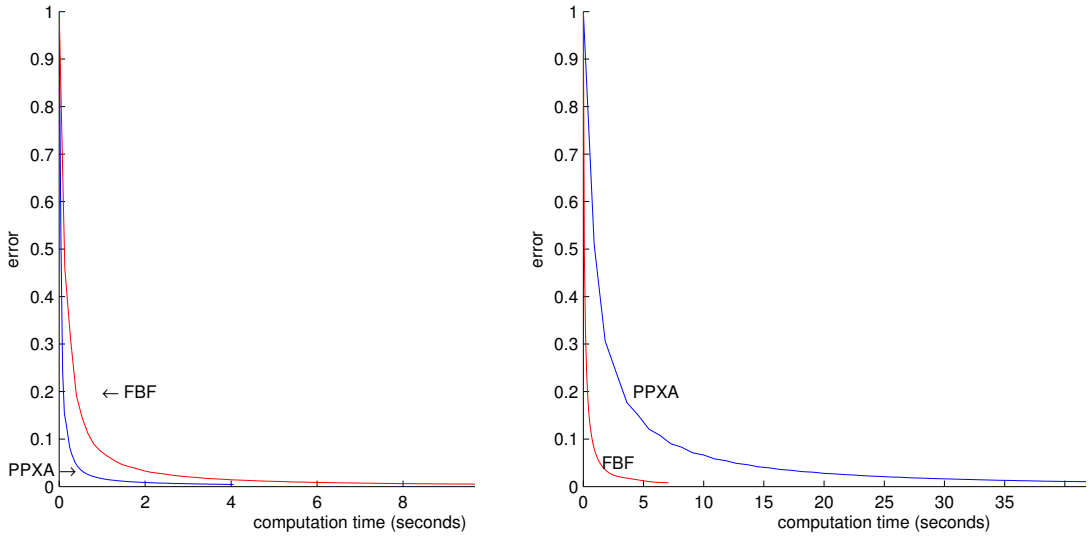
(f) Nonlocal DCTV SNR=26.1dB

Figure 3.6: Local and non-local denoising.
The nonlocal weights are computed using [44]. Regularization parameters used to obtain the best results: (d) $\lambda = 0.017$ (e) $\lambda = 0.015$ (f) $\lambda = 0.85$.

3.4.1.3 Comparison of applicability of PPXA and M+SFBF

The proximity operator associated with the data fidelity term in PPXA requires solving a linear system. In the case of an image recovery task using a local, regular graph (for example a 4-connected grid) as in the first part of Section 3.4.1, the special configuration of the matrix in the system may be taken into account. As this resulting matrix is circulant block-circulant, the system is efficiently solved using a Fast Fourier Transform. However, in the case of an arbitrary graph, for example a non-local graph as in Section 3.4.1.2, the matrix of the linear system is no longer circulant, so the system may be solved using a conjugate gradient method for instance.

When using the M+SFBF algorithm, the proximity operator of the function enforcing the data fidelity does not require us to solve a linear system. Instead, four matrix multiplications by the incidence matrix are performed per iteration. The speed of convergence using M+SFBF and PPXA are compared in Figure 3.7, for denoising an image using a local, and



(a) regular graph (b) non-regular graph

Figure 3.7: Comparison of convergence speed of the PPXA (blue) and M+SFBF (red) algorithms for denoising an image.
(a) using a regular, 4-connected lattice, (b) using a non-local graph.

a non-local graph. In the figure, only computation times are presented because the required number of iterations to reach convergence is the same for either algorithms. In a local graph, the matrix multiplications in M+SFBF make the algorithm slower than PPXA. Conversely, in a non-local graph, where the linear system of PPXA is solved using a conjugate gradient, M+SFBF converges much faster.

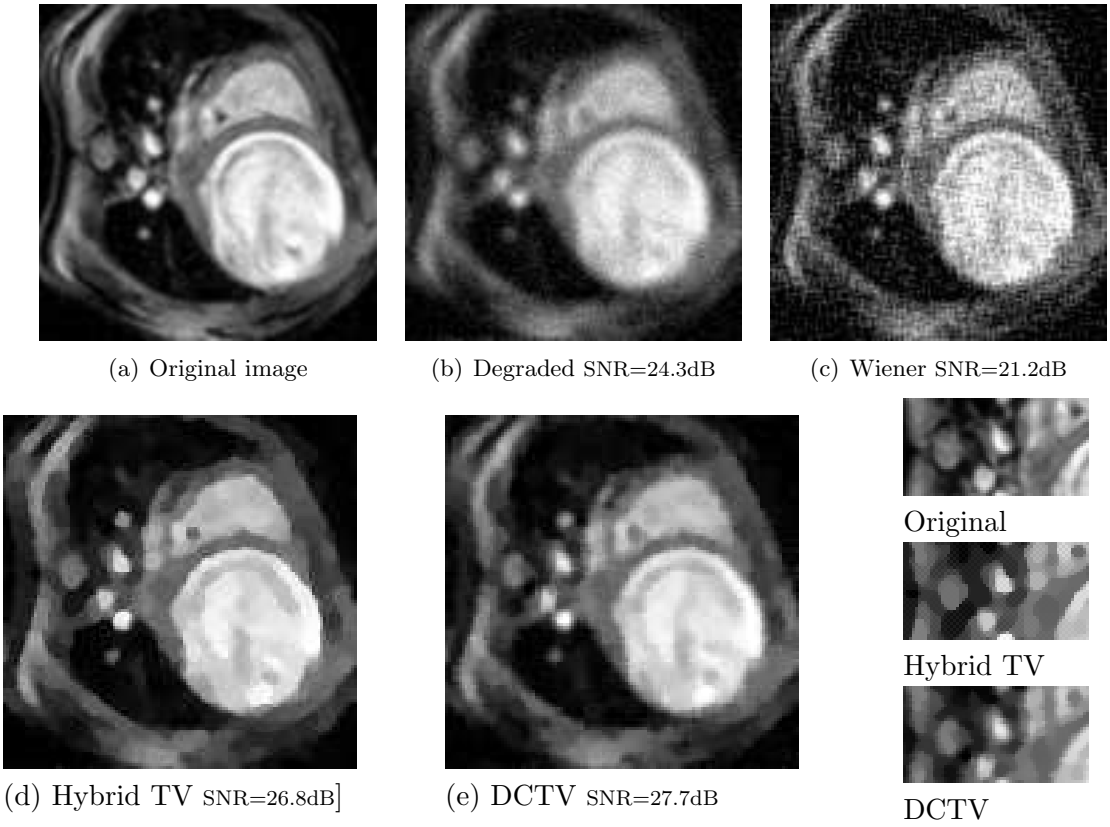
3.4.2 Image deconvolution

We now give some quantitative comparison results for joint denoising and deblurring tasks.

We compare the DCTV results to Wiener based deconvolution using the Matlab function ‘‘`deconvwrn`’’. The comparison also includes the hybrid TV/wavelet regularization method of Combettes and Pesquet [66]. We report in Table 3.4.2 the SNR values for restoration of images corrupted with additive white zero-mean Gaussian noise with variance $\sigma^2 = 5$, and $\sigma^2 = 10$, and convolved with uniform blur kernels of size 5×5 and 7×7. We observe that DCTV unsurprisingly outperforms the standard Wiener filter. More importantly, DCTV is competitive with a state-of-the-art method [66], both quantitatively in term of SNR, and qualitatively, without presence of checkerboard artifacts observed for the Hybrid TV method in Figure 3.8. Furthermore, the results are obtained twice faster, for the same number of iterations, using a Matlab implementations for both methods.

(σ^2, b)	(5,5)	(5,7)	(10,5)	(10,7)
SNR values obtained using **Wiener** deconvolution				
house	19.2	19.8	18.8	19.2
lena	18.2	18.9	18.1	18.8
barbara	14.7	14.5	14.2	14.0
man	15.0	15.6	15.2	15.9
mean	**16.8**	**17.2**	**16.5**	**17.2**
SNR values obtained using **Hybrid TV** [66]				
house	26.2	25.3	24.9	23.8
lena	24.9	23.8	23.9	22.9
barbara	16.4	15.7	15.8	15.4
man	21.6	20.6	20.6	19.6
mean	**22.3**	**21.4**	**21.3**	**20.4**
SNR values obtained by optimizing **DCTV**				
house	26.3	24.7	24.6	23.3
lena	25.5	23.9	24.0	22.8
barbara	16.9	15.8	16.0	15.5
man	22.1	20.6	20.8	19.6
mean	**22.7**	**21.3**	**21.4**	**20.3**

Table 3.2: Quantitative deconvolution experiment on standard images corrupted with additive white Gaussian noise with variance σ^2, and a Gaussian blur of kernel size $b \times b$. The DCTV weights have been set according to (3.12), where the estimate of the image $\overline{x}$ has been precomputed using DCTV with the original corrupted image f. This trick allows us to obtain an SNR improvements of about 0.5 dB, while computation times are still twice as fast than the hybrid TV method [66].



(a) Original image (b) Degraded SNR=24.3dB (c) Wiener SNR=21.2dB

(d) Hybrid TV SNR=26.8dB] (e) DCTV SNR=27.7dB

Figure 3.8: Denoising and deblurring an MRI image corrupted with synthetic Gaussian 5×5 blur and Gaussian noise ($\sigma^2 = 10$).
We observe in the TV hybrid result the presence of checkerboard artifacts due to the use of discrete filters for approximating the gradient. DCTV reduces the staircase effect of TV while preserving more details. Parameter used: Hybrid TV with regularization parameters [66]$\alpha = 0$ and $\beta = 0.025$, and DCTV with $\lambda = 0.005$ and $\eta = 0.04$.

3.4.3 Image fusion

We show here how to apply our Dual Constrained TV approach to image fusion. From N degraded images $f_1, \ldots, f_N$, the problem is to find a restored image x. The degradation model is the following:

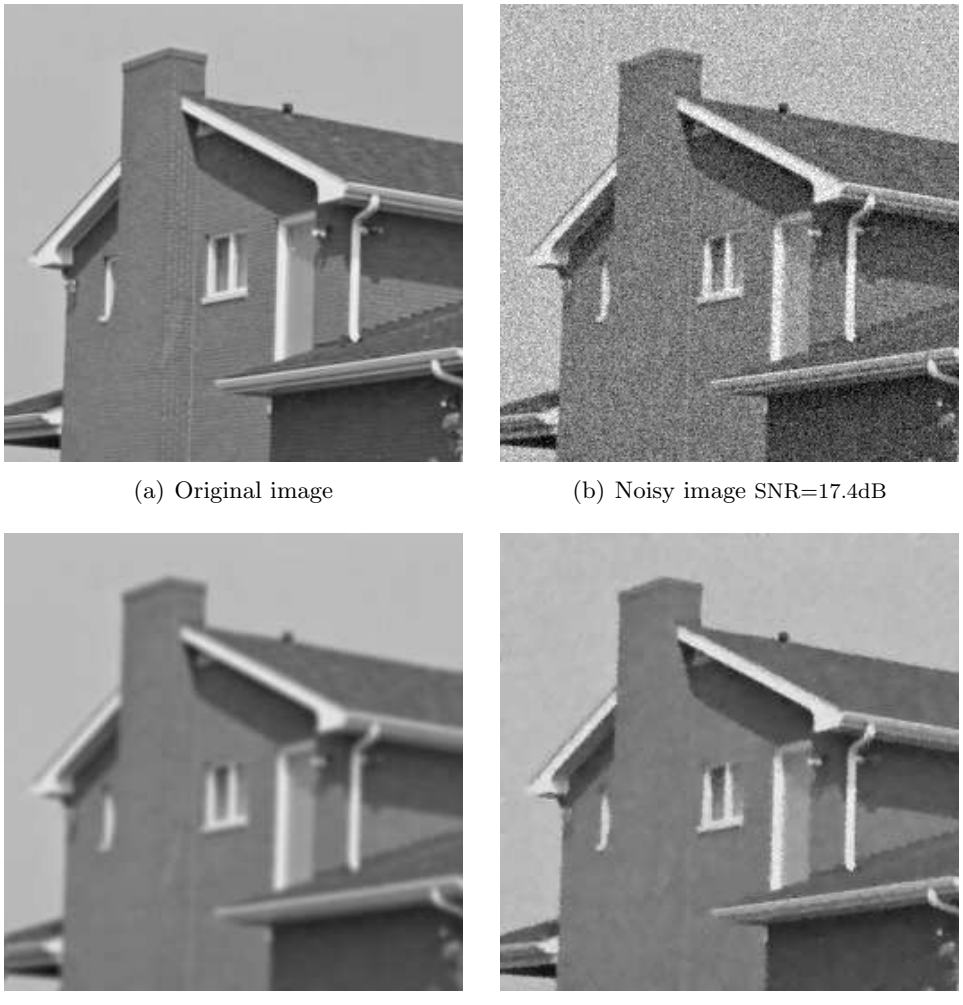
$$\begin{bmatrix} f_1 \\ \vdots \\ f_N \end{bmatrix} = \begin{bmatrix} H_1 \\ \vdots \\ H_N \end{bmatrix} x + \begin{bmatrix} b_1 \\ \vdots \\ b_N \end{bmatrix}, \tag{3.27}$$

where $H_1, ..., H_N$ are degradation matrices (i.e. generating blur), and $b_1, ..., b_N$ represent white additive zero-mean noises possibly with different variances, assumed to be uncorrelated.

The problem can be formulated as in (3.8) by setting

$$f = \begin{bmatrix} f_1 \\ \vdots \\ f_N \end{bmatrix}, \quad H = \begin{bmatrix} H_1 \\ \vdots \\ H_N \end{bmatrix}, \quad \Lambda = \begin{bmatrix} \lambda_1 I & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \lambda_N I \end{bmatrix} \tag{3.28}$$

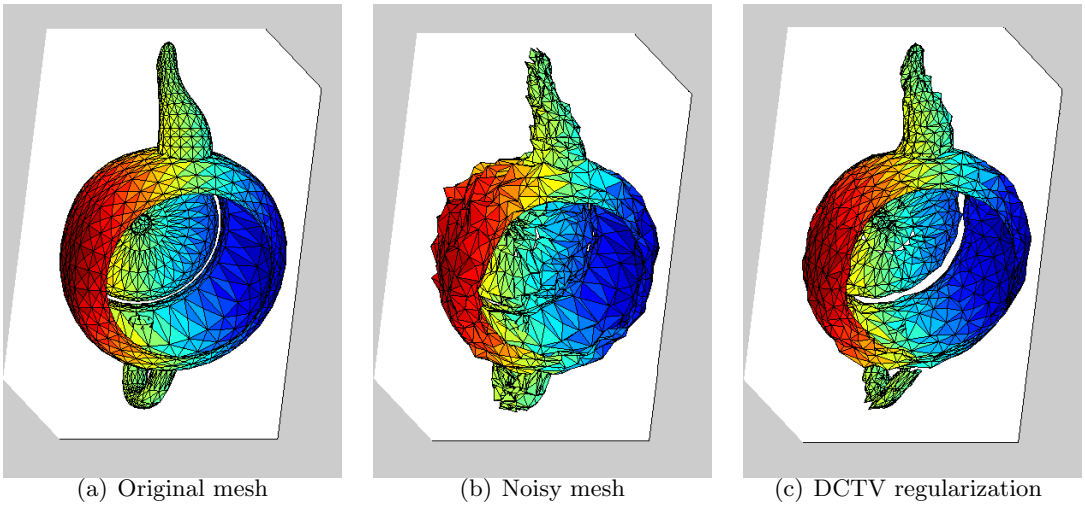
with $(\lambda_1, \dots, \lambda_N) \in]0, +\infty[^N$. We present in Figure 3.9 an example of fusion between a blurry image and a noisy image in a 4-connected lattice ($N = 2$ and $H_2 = I$).



(a) Original image (b) Noisy image SNR=17.4dB

(c) Blurry image SNR=23.9dB (d) DCTV SNR=26.3dB

Figure 3.9: Image fusion from an image corrupted with Gaussian noise with variance $\sigma_2^2 = 20$ (b) and a blurry image (c) (uniform blur kernel of size 5×5, with additive Gaussian noise with variance $\sigma_1^2 = 1$). (d) DCTV result with $\lambda_1 = 0.17$ and $\lambda_2 = 0.24$.



(a) Original mesh (b) Noisy mesh (c) DCTV regularization

Figure 3.10: Example of mesh denoising using DCTV on the spacial coordinates of the nodes. The M+SFBF algorithm was used to optimize DCTV with $\lambda = 0.25$.

On a rectangular grid, where the convex C is decomposed as described in Figure 3.1, the parallel proximal algorithm given in Algorithm 5 can be employed efficiently in this case.

3.4.4 Graph filtering

3.4.4.1 3D mesh filtering

The dual-constrained TV-based formulation being defined on arbitrary graphs, x is not limited to only represent image pixels. In Figure 3.10, we present an example of mesh denoising, where x is a vector composed of the spatial coordinates x_X, x_Y and x_Z of the mesh nodes. In this experiment, we added a randomly oriented with white zero-mean Gaussian magnitude noise vector to the original nodes of a mesh. This results in noisy mesh nodes of coordinates f_X, f_Y and f_Z. The degradation model is the following:

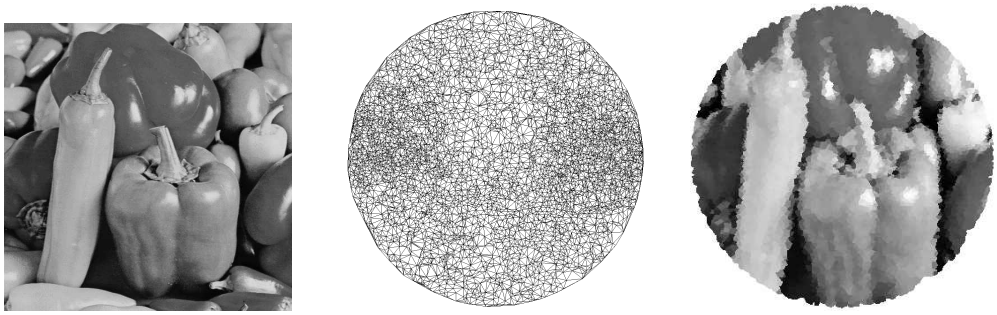
$$f = \begin{bmatrix} f_X \\ f_Y \\ f_Z \end{bmatrix} = \begin{bmatrix} x_X \\ x_Y \\ x_Z \end{bmatrix} + \sigma^2 \begin{bmatrix} b_X \\ b_Y \\ b_Z \end{bmatrix}, \tag{3.29}$$

where b_X, b_Y, b_Z represent vectors of additive noise with unit magnitude variance.

For this application, we used the M+SFBF algorithm, which is more efficient than PPXA as the graph is not regular. This application shows that the DCTV framework is well suited for regularizing various type of data.

3.4.4.2 Biologically sampled image filtering

This final example demonstrates the ability of DCTV to efficiently filter data on arbitrary graphs. In some situations, images are acquired from nonuniform samples. Specifically, most biological systems are known to acquire light nonuniformly. Following the work of [109], in Figure 3.11, we used the Graph Analysis Toolbox [116] that contains implementations of several filtering techniques in arbitrary graphs. From a graph representing the spacial resolution of the Bottlenosed dolphin visual system, the image (c) represents image resampled with the dolphin's mesh, corresponding to the input f of our method. It is then straightforward to run the M+SFBF algorithm, using the incidence matrix of the graph represented in (b).



(a) Original image

(b) Bottlenosed dolphin structure

(c) Sampled image

(d) Noisy sampled SNR = 22.1

(e) Taubin filtered result [208] SNR = 19.4

(f) DCTV result ($\lambda = 0.5$) SNR = 23.3

Figure 3.11: Filtering image data on a biologically sampled image [116].
Noise of variance $\sigma^2 = 10$ was added to the resampled values of the image (c) to produce (d).

Results obtained with DCTV are compared to results obtained using the mean filtering and lowpass filtering method of Taubin *et al.* [208]. We note that DCTV improves contrast preservation and thin objects rendering compared to Taubin's method.

3.5 Conclusion

In this chapter we have extended existing TV models by generalizing the constraint on the projection variable of the dual TV formulation. This new approach shows improved results compared with the weighted TV approach in image restoration applications. More generally, the proposed proximal algorithms make it possible to efficiently solve convex minimization problems involving the support function of an intersection of convex sets as a penalty term. It is also worth emphasizing that this approach can be applied on any graph data structures, in particular in those frequently employed in 3D modeling. By taking into account both edge and node weights, we have shown that the proposed framework brings improvements in denoising and deconvolution tasks. Its versatility also allows us to exploit extra information such as non local patch similarity into weight computation that makes it even more appealing.

Chapter 4

A unifying graph-based optimization framework: Power watershed

4.1 Introduction

The modern variations on graph-based segmentation algorithms are primarily built using a small set of core algorithms — graph cuts, random walker and shortest paths, which are reviewed shortly. Recently these three algorithms were all placed into a common framework that allows them to be seen as instances of a more general seeded segmentation algorithm with different choices of a parameter q [200]. In addition to these algorithms, the ubiquitous watershed segmentation algorithm [21] shares a similar seeding interface but only recently was a connection made between the watershed algorithm and graph cuts [4, 5]. In this chapter[1], we show how this connection between watershed and graph cuts can be used to further generalize the seeded segmentation framework of [200] such that watershed, graph cuts, random walker and shortest paths may all be seen as special cases of a single general seeded segmentation algorithm. Our more general formulation has several consequences which form our contributions.

1. This more general formulation reveals a previously unknown family of segmentation algorithms which we term *power watershed*. In this chapter, we give an algorithm for solving the energy minimization problem associated with the power watershed and demonstrate that this new algorithm has the speed of standard watershed but performing almost as well or better as all the other algorithms on our benchmark segmentation tests.

[1] Published in *Proc. of ICCV, 2009* [70], and *IEEE Trans. on PAMI, 2011* [72]

2. Placing watershed in the same framework as graph cuts, random walker and shortest paths allows us to easily incorporate data (unary) terms into conventional watershed segmentation.

3. By placing the watershed algorithm in the same generalized framework as graph cuts, random walker and shortest paths, it is possible to take advantage of the vast literature on improving watershed segmentation to also improve these other segmentation approaches.

4. Defining an energy function for the watershed optimization allows us to provide an MRF interpretation for the watershed.

5. By incorporating unary terms, we can push watershed beyond image segmentation into the area of general energy minimization algorithms which could be applied to any number of applications for which graph and MRF models have become standard.

Before proceeding to the exposition of our technique, we first review the graph-based segmentation literature in more detail. We refer to Chapter 1, and references therein for a more complete description.

4.2 A short review of graph-based segmentation

The algorithms that are reviewed in this section view the image as a graph with each pixel corresponding to a node and edges weighted to reflect changes in image intensity, color or other features.

Watershed: There exist many possible ways for defining a watershed [18, 77, 78, 166, 187, 216]. Intuitively, the watershed of a function (seen as a topographical surface) is composed of the locations from which a drop of water could flow towards different minima. The framework allowing the formalization and proof of this statement is the *optimal spanning forests relative to the minima* [76, 77]. For the purpose of seeded image segmentation, the gradient of the image can be considered as a relief map and, instead of minima, seeds may be placed by the user or found automatically to specify the segmentation of the image into desired regions. If the gradient is inverted, the maxima are considered instead of minima, and a thalweg is computed instead of watershed. A thalweg is the deepest continuous line along a valley. In the rest of the chapter, we use by convention the term "watershed" instead of "thalweg".

A maximum spanning forest (MSF) algorithm computes trees spanning all the nodes of the graph, each tree being connected to exactly one connected seed component, and the weight of the set of trees being maximum. If the seeds correspond to the maxima, the segmentation obtained by MSF is a watershed [77]. An optimal spanning forest can be computed by

Kruskal's or Prim's algorithm [144, 182] among others in quasi-linear time. In Kruskal's algorithm, the edges are sorted by decreasing edge weight, and chosen in that order to be added to the forest if they do not create cycles or join trees that are connected to different maxima.

Watersheds are widely used [168] in image segmentation because there exist numerous and efficient algorithms that are easy to implement. However, segmentation results from watershed may suffer from leaks and degeneracy of the solution on the plateaus of the weight function.

Graph cuts: The labeling produced by the graph cuts (GC) algorithm is determined by finding the minimum cut between the foreground and background seeds via a maximum flow computation. The original work on GC for interactive image segmentation was produced by Boykov and Jolly [32], and this work has been subsequently extended by several groups to employ different features [25] or user interfaces [151, 188]. Although GC is relatively new, the use of minimal surfaces in segmentation has been a common theme in computer vision for a long time [26, 99, 162] and other boundary-based user interfaces have been previously employed [61, 92, 112, 161]. Two concerns in the literature about the original GC algorithm are metrication error ("blockiness") and the shrinking bias. Metrication error was addressed in subsequent work on GC by including additional edges [34], by using continuous max flows [10] or total variation [210]. These methods for addressing metrication error successfully overcome the problem, but may incur greater memory and computation time costs than the application of maximum flow on a 4-connected lattice. The shrinking bias can cause overly small object segments because GC minimizes boundary length. Although some techniques have been proposed for addressing the shrinking bias [10, 34, 214], these techniques all require additional parameters or computation.

Random walker: The random walker (RW) algorithm [110] is also formulated on a weighted graph and determines labels for the unseeded nodes by assigning the pixel to the seed for which it is most likely to send a random walker. This algorithm may also be interpreted as assigning the unlabeled pixels to the seeds for which there is a minimum diffusion distance [62], as a semi-supervised transduction learning algorithm [86] or as an interactive version of normalized cuts [118, 197]. Additionally, popular image matting algorithms based on quadratic minimization with the Laplacian matrix may be interpreted as employing the same approach for grouping pixels, albeit with different strategies to determine the edge weighting function [148]. Diffusion distances avoid segmentation leaking and the shrinking bias, but the segmentation boundary may be more strongly affected by seed location than with graph cuts [200].

Shortest paths (geodesics): The shortest path algorithm assigns each pixel to the foreground label if there is a shorter path from that pixel to a foreground seed than to any background seed, where paths are weighted by image content in the same manner as with the

q \ p	0	finite	∞
1	Collapse to seeds	Graph cuts	MSF, watershed
2	ℓ_2 norm Voronoi	Random walker	Power watershed $q = 2$
∞	ℓ_1 norm Voronoi	ℓ_1 norm Voronoi	Shortest Path Forest

Table 4.1: Our generalized scheme for image segmentation includes several popular segmentation algorithms as special cases of the parameters p and q. The power watershed are previously unknown in the literature, but may be optimized efficiently with a maximum spanning forest calculation.

GC and RW approaches. This approach was recently popularized by Bai and Sapiro [16], but variants of this idea have appeared in other sources [6, 79, 91]. The primary advantage of this algorithm is speed and prevention of a shrinking bias. However, it exhibits stronger dependence on the seed locations than the RW approach [200], is more likely to leak through weak boundaries (since a single good path is sufficient for connectivity) and exhibits metrication artifacts on a 4-connected lattice.

All of the above models may be considered as addressing energies comprised of only unary and pairwise (binary) energy terms. However, recent literature has found that the addition of energy terms defined on higher-order cliques can help improve performance on a variety of tasks [138, 139]. Although we do not address higher-order cliques specifically in this work, we note that all recent progress in this area has been through an equivalent construction of pairwise terms. Therefore, our results could also be useful in that context. Despite the recent popularity of energies defined on higher order cliques, pairwise terms (and watershed) are still used ubiquitously in the computer vision literature and any improvement to these models can have a broad impact.

4.3 A unifying energy minimization framework

We begin our exposition by reviewing the unity framework of [200] before showing how to further broaden this framework to provide a general seeded segmentation scheme that includes the maximum spanning forest algorithm for watershed as a special case. Examination of the special cases of this general algorithm reveals a new class of watershed segmentation models. We prove several theoretical properties of this new class of watershed and then give an algorithm for minimizing the energy associated with this generalized watershed model.

4.3.1 A review of existing generalized segmentation framework

In this section, we review the segmentation framework introduced by Sinop and Grady in [200]. In image processing applications, each pixel is typically associated with a node and the nodes are connected locally via a 4 or 8-connected lattice. We recall that the weight of an edge e_{ij} is denoted by $w(e_{ij})$ or w_{ij}. We also denote w_{Fi} and w_{Bi} as the unary weights penalizing foreground and background affinity at node v_i. In the context of segmentation and clustering applications, the weights encode nodal affinity such that nodes connected by an edge with high weight are considered to be strongly connected and edges with a low weight represent nearly disconnected nodes. One common choice for generating weights from image intensities is to set

$$w_{ij} = \exp(-\beta(\nabla I)^2), \tag{4.1}$$

where ∇I is the normalized gradient of the image I. The gradient for a grey level image is $I_i - I_j$. Details on the parameters used are given in the experimental section. We use w to denote the vector of $\mathbb{R}^m$ that contains the weights w_{ij} of every edge e_{ij} in G.

The generalized energy proposed in [200] is given by as

$$\min_x \sum_{e_{ij}\in E} (w_{ij}|x_i - x_j|)^q + \sum_{v_i \in V} (w_i|x_i - y_i|)^q \tag{4.2}$$

where y represents a measured configuration and x represents the target configuration. In this equation w_{ij} can be interpreted as a weight on the gradient of the target configuration, such that the first term penalizes any unwanted high-frequency content in x and essentially forces x to vary smoothly within an object, while allowing large changes across the object boundaries. The second term enforces fidelity of x to a specified configuration y, w_i being weights enforcing that fidelity.

For an image segmentation in two classes, given foreground F and background B seeds, (4.2) may be included in the following algorithm

$$\begin{aligned} \text{Step 1: } x = \arg\min_x & \sum_{e_{ij}\in E} (w_{ij}|x_i - x_j|)^q + \\ & \sum_{v_i} (w_{Fi}|x_i|)^q + \sum_{v_i} (w_{Bi}|x_i - 1|)^q, \\ \text{s.t. } & x(F) = 1, \quad x(B) = 0, \\ \text{Step 2: } & s_i = 1 \text{ if } x_i \geq \frac{1}{2}, 0 \text{ if } x_i < \frac{1}{2}. \end{aligned} \tag{4.3}$$

In other words, we are looking for an optimum $x^\star$ of Eq. (4.3) that may be interpreted as a probability for a given pixel to belonging to either the foreground or the background, the final decision (hard segmentation) s giving the segmentation being taken by a threshold.

It was shown in [200] that graph cuts gives a solution to this model when $q = 1$, random walker gives the solution to this model when $q = 2$ and shortest paths (geodesics) give a solution to this model as $q \to \infty$. The case of this model with a fractional q was optimized in [199] via reweighted least squares and shown that intermediate values of q allowed for an algorithm which "interpolated" between the graph cuts, random walker or shortest paths algorithms.

In related work, Strang showed in [202] that minimization of the ℓ_p norm of the gradients of a potential field with boundary conditions (in continuous space with real-valued potentials) also leads to (continuous) max-flow (for an ℓ_1 norm of the gradients), the Dirichlet problem (for an ℓ_2 norm) and shortest paths (for an ℓ_∞ norm). Therefore, the framework of [200], which we now extend, may be seen as presenting similar ideas defined on an arbitrary graph, using the bridge between continuous PDEs and graph theory provided by discrete calculus [119].

4.3.2 Broadening the framework to watershed

We now broaden the segmentation algorithm in (4.3) to include watershed simply by separating the exponent on the weights and the variables. Specifically, we introduce parameter p to define a new segmentation model as

$$\begin{aligned} \text{Step 1: } x = \arg\min_x & \sum_{e_{ij}\in E} w_{ij}^p |x_i - x_j|^q + \\ & \sum_{v_i} w_{Fi}^p |x_i|^q + \sum_{v_i} w_{Bi}^p |x_i - 1|^q, \\ \text{s.t. } & x(F) = 1, \;\; x(B) = 0, \\ \text{Step 2: } & s_i = 1 \text{ if } x_i \geq \frac{1}{2}, 0 \text{ if } x_i < \frac{1}{2}. \end{aligned} \tag{4.4}$$

As before, the final segmentation s being chosen via a threshold.

We observe that (4.4) can be formulated in a general manner by rewriting it as the minimization of a general energy function $E_{p,q}(x)$ by introducing auxiliary nodes (See [120] for more details):

$$\boxed{\min_x \lambda \sum_{e_{ij}\in E} \underbrace{w_{ij}^p |x_i - x_j|^q}_{\text{smoothness terms}} + \sum_{v_i \in V} \underbrace{w_i^p |x_i - y_i|^q}_{\text{data fidelity terms}}} \tag{4.5}$$

For example, the unary term $w_{B_i}^p |x_i - 1|^q$ can also be rewritten as $w_i^p |x_i - y_i|^q$, where y_i is an auxiliary node and the signal at this auxiliary node is fixed at $y_i = 1$.

As with (4.3), when p is a small finite value, then the various values of q may be interpreted respectively as the graph cuts ($q = 1$), and random walker ($q = 2$) algorithms. When q and

p converge toward infinity with the same speed, then a solution to (4.4) can be computed by the shortest path (geodesics) algorithm. These three algorithms form the underpinning for many of the advanced image segmentation methods in the literature.

It was shown in [4, 5] that when $q = 1$ (graph cuts) and $p \to \infty$ then the solution of (4.4) is given by a maximum spanning forest algorithm. Said differently, as the power of the weights increases to infinity, then the graph cuts algorithm produces a segmentation corresponding to a segmentation by maximum spanning forest. Interpreted from the standpoint of the Gaussian weighting function in (4.1), it is clear that we may associate $\beta = p$ to understand that the watershed equivalence comes from operating the weighting function in a particular parameter range. An important insight from this connection is that *above some value of β we can replace the expensive max-flow computation with an efficient maximum spanning forest computation*. By raising $p \to \infty$ and varying the power q we obtain a previously unexplored family of segmentation models which we refer to as **power watershed**. An important advantage of power watershed with varying q is that the main computational burden of these algorithms depends on an MSF computation, which is extremely efficient [57]. In the next sections we explore two cases that are, to the best of our knowledge, unexplored. First, we show in Appendix A.2.1 that the case p finite, $q \to \infty$ corresponds to a Voronoi diagram computation from the seeds. Second, we prove that when q is finite, as $p \to \infty$ there exists a value of p after which any of the algorithms (regardless of q) may be computed via an MSF. We then give an algorithm to minimize (4.4) for any value of q when $p \to \infty$. Table 4.1 gives a reference for the different algorithms generated by various value of p and q.

4.3.3 The case q finite, $p \to \infty$ leading to watershed

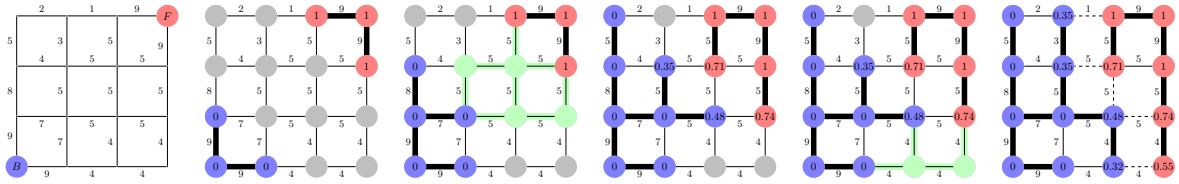
We now generalize the link between GC and MSF established by Allène *et al.* [4, 5] by proving that GC, RW, and generally all cuts resulting of the minimization of $E_{p,q}$ converge to MSF cuts as p tends toward infinity under the condition that all the maxima of the weight function are seeded.

The following properties are presented in the special case of segmentation into two classes, given two sets of labeled nodes F and B. However the following results generalize easily to multilabel segmentation.

Definition 4.3.1. *(q-cut, MSF, MSF cut)*

In a graph G, let F and B be two disjoint nonempty sets of nodes, p and q two real positive values, and s the segmentation result defined in Eq 4.4. The set of edges e_{ij} such that $s_i \neq s_j$ is a q-cut.

Let Y be a subgraph of G. We say that Y is an extension *of $F \cup B$ if each connected component of Y contains exactly one vertex of $F \cup B$ and each vertex of $F \cup B$ is contained in a connected component of Y. Consequently it is possible to define a* label *l on each vertex*



(a) (b) (c) (d) (e) (f)

Figure 4.1: Illustration of different steps in the proof of Thm. 1 for $q = 2$.
The values on the nodes correspond to x, their color to s. The bold edges represents edges belonging to an MSF. (a) A weighted graph with two seeds, all maxima of the weight function are seeded, (b) First step, the edges of maximum weight are added to the forest, (c) After several steps, the next largest edge set belongs to a plateau connected to two labeled trees, (d) Minimize (4.4) on the subset (considering the merged nodes as a unique node) with $q = 2$ (i.e., solution of the combinatorial Dirichlet problem), (e) Another plateau connected to three labeled vertices is encountered, and (f) Final solutions x and s obtained after few more steps. The q-cut, which is also an MSF cut, is represented in dashed lines.

of Y, 0 to the vertices connected to a vertex of B, and 1 to the vertices connected to a vertex of F.

Examples of extensions appear in Fig. 4.1, where F and B are displayed in (a), and two possible extensions in bold in (b) and (c), with their corresponding labels.

Let $\mathcal{F}$ be a subgraph of G. We say that $\mathcal{F}$ is a spanning forest (relative to $F \cup B$) *if:*
(i) $\mathcal{F}$ is an extension of $F \cup B$,
(ii) $\mathcal{F}$ contains no cycles, and
(iii) $V(\mathcal{F}) = V$ ($\mathcal{F}$ is spanning all vertices of G).

The weight $w_{\mathcal{F}}$ of a forest $\mathcal{F}$ for w *is the sum of the weight of all edges belonging to $\mathcal{F}$: $w_{\mathcal{F}} = \sum_{e_{ij} \in \mathcal{F}} w_{ij}$.*

We say that a spanning forest $\mathcal{F}$ is a maximum spanning forest (MSF) (relative to $F \cup B$) for w *if the weight of $\mathcal{F}$ is maximum, i.e. greater or equal to the weight of any other spanning forest (relative to $F \cup B$).*

Let $\mathcal{F}$ be an MSF for w, and l its associated label. An MSF cut for w *is the set of edges e_{ij} such that $l_i \neq l_j$.*

We call a subgraph M a maximum *of w if M is connected, all the edges of M have the same weight w_M, and the weight of any edge adjacent to M is strictly lower than w_M.*

Finally, a plateau *is a subgraph of G consisting of a maximal set of nodes connected with edges having the same weight.*

These definitions are compatible with the watershed cut framework of [77]. We may now introduce a general link between the Maximum Spanning Forest segmentation result and the solution of the optimization of (4.4) when the power of the weights converges toward infinity.

Theorem 4.3.1. *Let M be the subgraph of G composed of the union of all maxima of the weight function w. If every connected component of M contains at least a vertex of $B \cup F$, and $q \geq 1$, then any q-cut when $p \to \infty$ is an MSF cut for w.*

Proof. The proof is based on the construction of a set of edges that belong to the q-cut when $p \to \infty$. During the construction, we consider the edges of E in decreasing order, following Kruskal's algorithm for maximum spanning forest construction. At the end of the construction, the q-cut obtained is an MSF cut for w. The successive steps of the proof are illustrated on an example on Fig. 4.1.

At each step, we consider the set $E_{\max}$ of edges of maximum weight $w_{\max}$. We normalize all the weights by dividing them by $w_{\max}$, to obtain all the weights between 0 and 1 with the normalized weight of $E_{\max}$ equal to 1. The energy to minimize is also

$$\sum_{e_{ij} \in E} \left(\frac{w_{ij}}{w_{\max}} \right)^p |x_i - x_j|^q, \text{ s.t. } \begin{cases} x(F) = 1, \\ x(B) = 0. \end{cases} \tag{4.6}$$

As all maxima of the weight function contain seeds, each connected component of $E_{\max}$ has at least one labeled vertex. For every connected component $C_{\max}$ of $E_{\max}$, two cases are possible:

If $C_{\max}$ contains no vertices of different labels, the edges of weight $w_{\max}$ can not be a part of the minimum q-cut energy when p tends toward infinity because all the other normalized weights converge toward 0 and so does any finite sum of these weights. Choosing $x_i = x_j$ for all edges $e_{ij} \in C_{\max}$ is the only possibility to eliminate the terms of maximum weight of (4.6). The edges of $C_{\max}$ are not included in the q-cut, and also do not belong to the MSF cut as they have to be merged to labeled nodes to form an MSF (e.g., Figure 4.1(b)).

If $C_{\max}$ contains vertices of different labels, any labeling can be done on the plateau, because adding edges of $C_{\max}$ to the q-cut or not will always give an MSF cut on the plateau (e.g., Figures 4.1(c) and 4.1(d)).

Repeating the steps recursively until all the vertices are labeled, we find that in building a q-cut, we are also building an MSF cut for w in exactly the same manner as with Kruskal's algorithm. □

In theorem 4.3.1, the condition for seeds to be the maxima of the weight function is necessary as specified in Appendix A.2.2.

We can note that if the weights are all different, the MSF cut is unique and Thm. 4.3.1 is also true without the condition for seeds to be the maxima of the weight function. We can also note that in the case $q = 1$, the authors of [5] explicitely provide a lower bound for p such that the min-cut corresponds to an MSF-cut.

The next property states that when the power on the neighboring node differences is strictly greater than one, the minimization of $E_{p,q}$ admits a unique solution, and the proof is given in Appendix A.2.3.

Property 4. *If q is a real number such that $1 < q < \infty$, then the solution x to problem (4.4) is unique.*

An interpretation of the minimization of our general energy as a maximum *a posteriori* approximation is presented in Appendix A.2.5.

We now introduce an algorithm to optimize $E_{p,q}$ when $p \to \infty$, and show that the threshold s of that solution produces an MSF cut.

4.4 Algorithm for optimizing the case q finite, $p \to \infty$

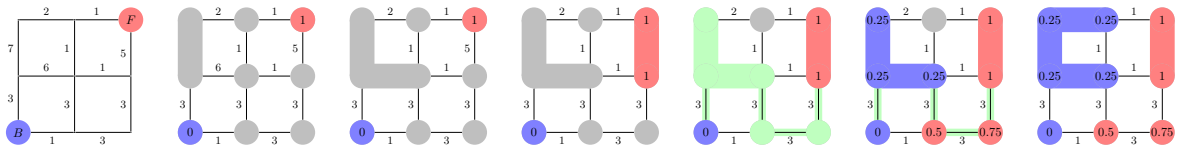


Figure 4.2: Example of behavior of the power watershed algorithm for $q = 2$ with the formation of a plateau that was not present in the original graph.
(a) Initialization: A weighted graph with two seeds. (b),(c),(d) First steps: The nodes of edges of maximum weight are merged , (e) The next largest edge set belongs to a plateau connected to two different labels, (f) Minimize (4.4) on the subset with $q = 2$ (i.e., utilize the random walker algorithm on the plateau), and (g) Final segmentation obtained after one more step.

The algorithm proposed in this section may be seen as Kruskal's algorithm for maximum spanning tree with two main differences — a forest is computed in place of a tree, and the optimization

$$\min_{x} \sum_{e_{ij} \in \text{ plateau}} |x_i - x_j|^q \tag{4.7}$$

is performed on the plateaus (the maximal set of nodes connected with edges of same weight). The power watershed algorithm is detailed in Alg. 1, and an illustration of different steps on an example is given in Fig. 4.2.

In Algorithm 7, the **merge** operation of a set of nodes S consists of removing the nodes in S from the graph and replacing these nodes with a single node such that any edge spanning a node in S to nodes in $\overline{S}$ now connects the merged node to the same nodes in $\overline{S}$. Additionally, in the above algorithm, the unary terms in (4.4) are treated as binary terms connected to

Algorithm 7: power watershed algorithm, optimizing $p \to \infty, q \geq 1$

Data: A weighted graph $G(V, E)$ and a set of foreground F and background B seeds
Result: A potential function x and a labeling s associating a label to each vertex.
Set $x_F = 1$, $x_B = 0$ and all other x values as unknown.
Sort the edges of E by decreasing order of weight.
while *any node has an unknown potential* **do**
  Find an edge (or a plateau) E_{MAX} in E of maximal weight; denote by S the set of nodes connected by E_{MAX}.
  if *S contains any nodes with known potential* **then**
    Find x_S minimizing (4.4) (using the input value of q) on the subset S with the weights in E_{MAX} set to $w_{ij} = 1$, all other weights set to $w_{ij} = 0$ and the known values of x within S fixed to their known values. Consider all x_S values produced by this operation as known.
  else
    Merge all of the nodes in S into a single node, such that when the value of x for this merged node becomes known, all merged nodes are assigned the same value of x and considered known.
Set $s_i = 1$ if $x_i \geq \frac{1}{2}$ and $s_i = 0$ otherwise.

phantom seeds v_F and v_B, i.e.,

$$\sum_{v_i} w_{Fi}^p |x_i - 0|^q + \sum_{v_i} w_{Bi}^p |x_i - 1|^q = \sum_{v_i} w_{Fi}^p |x_i - x_B|^q + \sum_{v_i} w_{Bi}^p |x_i - x_F|^q. \quad (4.8)$$

We prove in the next section that the labeling x obtained by Algorithm 7 optimizes (4.4).

An illustration for this section is given in Figure 4.3. The segmentation was performed with progressively larger values of p, keeping $q = 2$ and shows that the segmentation result converges to the result given by the above algorithm for the power watershed with $q = 2$. The value $q = 2$ was employed for this example since it is known that $q = 2$ forces a unique minimum to (4.4) regardless of the value of p.

An implementation of Algorithm 7 when $q = 2$ can be downloaded from sourceforge [1]. Some implementation details are given in Appendix A.2.6.

4.4.1 Justification of the power watershed algorithm

We now prove that the algorithm we propose optimizes the energy presented in our framework when $q > 1$ and $p \to \infty$.

Let us define the labeling x^* as the solution $x^* = \arg\min_x E_{p,q}(x)$ defined in (4.4) subject to the boundary constraints. We note the labeling obtained by Algorithm 7 by $\bar{x}$.

The two following theorems 4.4.1 and 4.4.2 state that the energy of the solution computed by the power watershed algorithm converges to the energy which minimizes $E_{p,q}$ when $p \to \infty$.

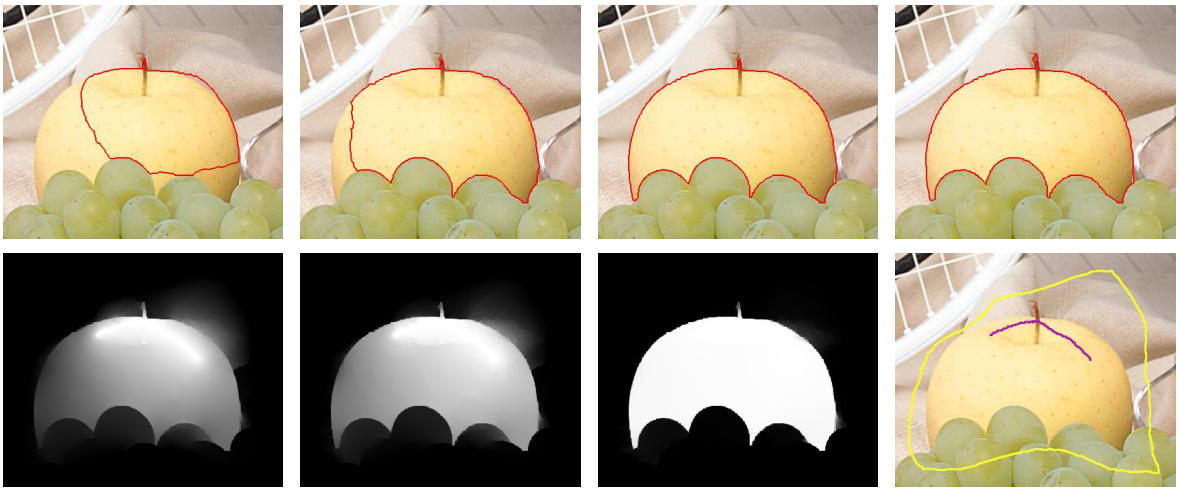


Figure 4.3: Illustration of progressive convergence of the random walker result to the power watershed result as $p \to \infty$, using $q = 2$.
Top row: Segmentation results obtained by random walker with weights at the power $p = 1$, $p = 8$, $p = 25$ and finally by the power watershed algorithm. Bottom row: Corresponding potentials for $p = 1$, $p = 8$, $p = 25$ and the input seeds.

Theorem 4.4.1. *Let p, q be real positive numbers. Let w_M be the maximum weight of the graph G. For every $\delta > 0$, there exists a real k such that if $p > k$,*

$$0 \leq \frac{E_{p,q}(\bar{x})}{{w_M}^p} - \frac{E_{p,q}(x^*)}{{w_M}^p} \leq \delta. \tag{4.9}$$

The proof of this theorem is given in Appendix A.2.4.

Theorem 4.4.2. *If $q > 1$, the potential x^* obtained by minimizing the energy of (4.4) subject to the boundary constraints converges toward the potential $\bar{x}$ obtained by Algorithm 7. as $p \to \infty$.*

Proof. We prove that by optimizing (4.4) we are performing the same steps as Alg. 7. As in Thm. 4.3.1, at each step, we consider a set of connected edges of maximum weight $E_{\max}$ of E, and we normalize all the weights, minimizing also (4.6).

If $E_{\max}$ contains no vertices of different labels, then the weights $w_{\max}$ can not be a part of the minimum energy when p tends toward infinity, because all the other normalized weights converge toward 0 and so does any finite sum of these weights. Choosing $x_i = x_j$ for every edge $e_{ij} = e_{\max} \in E_{\max}$ is the only possibility to eliminate the only term(s) of maximum weight of (4.6). This choice of $\bar{x_i} = \bar{x_j}$ is also performed by Algorithm 7 by the "merge" operation. From the standpoint of energy minimization, having $x_i = x_j$ in the graph G may be brought back to having one unique node instead of v_i, v_j, and e_{ij}. We can also replace v_i

and v_j by a unique node.

If $E_{\max}$ contains vertices of a different label, as the weights of $E_{\max}$ are arbitrarily greater than the weights of the unprocessed edges, minimizing (4.6) boils down to minimizing

$$\sum_{e_{ij} \in E_{\max}} |x_i - x_j|^q, \tag{4.10}$$

with boundary conditions given by already labeled nodes. It is exactly what is performed by Algorithm 7 in the "If" part.

Repeating the steps recursively until all the vertices are labeled, we find that the Algorithm 7 procedure agrees with the energy minimization of (4.5). □

We can note that even if Algorithm 7 minimizes the energy $E_{p,q}$ in the case $p \to \infty$, several solutions $\bar{x}$ are possible when $q = 1$.

Property 5. *For any $q \geq 1$, the cut C defined by the segmentation s computed by Algorithm 7 is an MSF cut for w.*

Proof. At each step of Algorithm 7, we consider a set of connected edges of maximum weight $E_{\max}$.

If $E_{\max}$ contains no vertices of different labels, Algorithm 7 chooses $x_i = x_j$ for the edges $e_{ij} \in E_{\max}$. The edges of $E_{\max}$ are not included in C, and also do not belong to the MSF cut as they have to belong to an MSF since their weight is maximum.

If $E_{\max}$ contains vertices of different labels, any labeling can be done on the plateau, because adding edges of $E_{\max}$ to the q-cut or not will always give an MSF cut on the plateau.

Repeating the steps of Algorithm 7 recursively until all the vertices are labeled, we find that we are building an MSF cut for w.

□

4.4.2 Uniqueness of the solution

Most of the energy minimization problems in our framework, *i.e.* the cases optimized by graph cuts, shortest path forests and maximum spanning forest algorithms (and watershed in general [21, 23, 75, 93, 187]) have the problem that the optimum solution may not be unique, for example on plateaus. This implies that the result of each one of these algorithms depends on the implementation.

To remove such dependency, two approaches have been proposed:

- A classical approach is to compute a geodesic distance on the plateau [187], and to use that distance as a way to distinguish between points of the plateau. Generally, the cut is located on the "middle of the plateau", but other locations are possible according to the application [75, 166].

- Another proposal is the tie-zone watershed [12]; it takes into account all the possible solutions derived from a shortest-path based watershed to generate a unique solution: when the multiple solutions disagree with each other on the segmentation result of a region (i.e., the label to be assigned), the region is included to the tie-zone and a specific tie value is assigned to each node, corresponding to the probability of assigning a label to the node according to the number of all possible assignments. A major drawback of that tie-zone approach is that nodes with equal probability of belonging to different label classes can appear.

In contrast to that approaches, the power watershed compute a probability map (consisting of x in (4.4)) minimizing a global energy function, and whenever q is finite and $q > 1$, the solution is unique.

4.5 Results

4.5.1 Generality of the framework

4.5.1.1 Adding unary terms

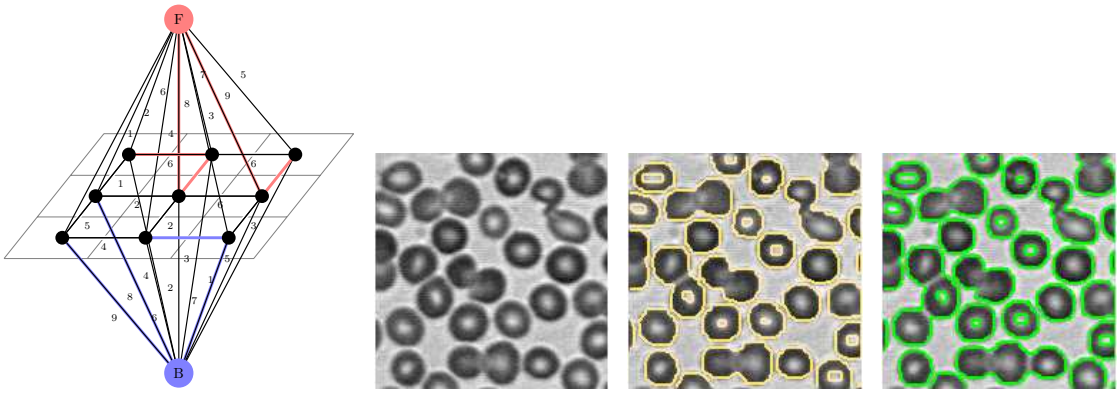
We now present an application of the framework to unseeded segmentation. Unary terms were first employed with graph cuts in [120]. Since this initial work, many other applications have used graph cuts with unary terms. Gathering watershed and graph cuts into the same framework allows us to employ unary terms for watershed computation.

The unary terms in (4.4) are treated as binary terms connected to phantom seeds v_F and v_B as in (4.8).

For the example of image segmentation with two labels, the weights w_{Bi} between v_B and v_i can be fixed to the absolute difference of the pixel v_i intensity with the mean of the gray scales plus the variance, and w_{Fi} to the absolute difference of the pixel v_i intensity with the mean of the gray scales minus the variance. An example of such a weighted graph is given in Fig. 4.4. With this construction, we can apply any of the algorithms in our framework to the resulting graph. An example of result is shown at Fig. 4.4 for purposes of segmenting blood cells. Note that those examples show how to add two phantom seeds, but this idea is extendable to more than two labels as explained in the next section. To the best of our knowledge, this is the first time that the watershed algorithm has been used as an unseeded segmentation method (*i.e.*, without markers or seeds).

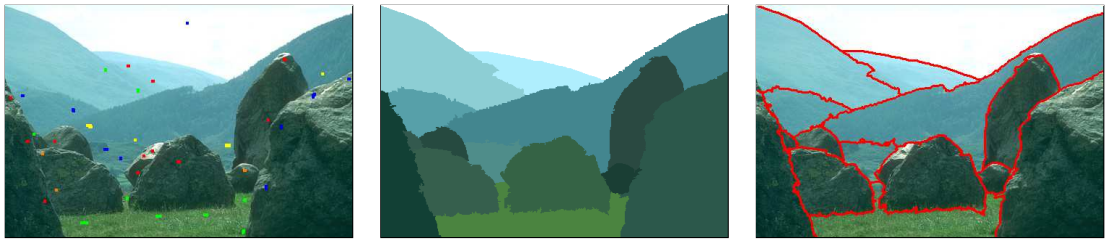
4.5.1.2 Multilabel segmentation

Minimizing exactly the energy $E_{1,1}$ is possible by using the graph cuts algorithm in the case of two labels, but is generally NP-hard if constraints impose more than two different labels.



(a) (b) (c) (d)

Figure 4.4: Unseeded segmentation using unary terms.
(a) Example of the unseeded segmentation of a 3×3 image computed with a maximum spanning forest (watershed). (b) An image of blood cells, (b) graph cuts result(c) Maximum Spanning Forest (watershed) result.



(a) (b) (c)

Figure 4.5: Example segmentations with more than two labels.
(a) Seeds, (b, c) power watershed result ($q = 2$).

However, the other algorithms presented in our framework can perform seeded segmentation with as many labels as desired efficiently.

We detail the method of multilabel segmentation in the case of the power watershed algorithm. Let N represent the number of different labels $l = 1, 2..., N$. Instead of computing an x solution of the Foreground/Background as it is done for the two-labels segmentation, N solutions x^l have to be computed. In order to perform N-labels segmentation, we may define seeds at a node i by setting $x_i^l = 1$ for a given label l and $x^{\bar{l}} = 0$ for any other label than l.

The segmentation result is obtained by affecting each node v_i to the label which x_i^l is maximum:

$$s_i = \arg\max_l x_i^l \tag{4.11}$$

An example of result is shown on Fig. 4.5.

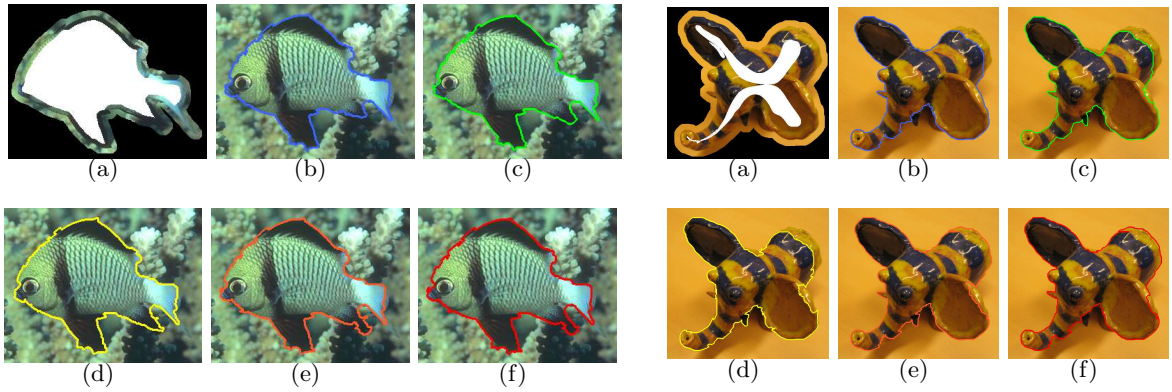


Figure 4.6: Example segmentations using the provided (left images) and skeletonized (right images) set of seeds on the Grabcut database images
(a) Seeds, (b) Graph cuts, (c) Random walker, (d) Shortest path, (e) Maximum spanning forest (standard watershed), and (f) Power watershed ($q = 2$).

4.5.2 Seeded segmentation

We now demonstrate the performance of power watershed with respect to the other seeded image segmentation algorithms. In the introduction we discussed how many of the leading graph-based segmentation algorithms (e.g., Grabcut, lazy snapping, closed-form matting) have graph cuts, random walker, shortest paths or watershed as an underlying component. Consequently, we will not compare the Power Watershed to any of the complete segmentation systems listed above, but rather against the comparable (component) algorithms of graph cuts, random walker, shortest paths and watershed. Additionally, to simplify the comparison we will not employ unary terms in our segmentations.

4.5.2.1 Quantitative assessment

Our experiments consist of testing five algorithms embodying different combinations of p and q, consisting of graph cuts, (GC), random walker (RW), shortest path (SP), watersheds/maximum spanning forest (MSF), and power watershed using the power $q = 2$. As before, we chose to employ the power watershed algorithm with $q = 2$ due to the uniqueness of the solution to (4.4) for this setting.

We used the Microsoft 'Grabcut' database available online [188], which is composed of fifty images provided with seeds. However, the seeds provided by the Grabcut database are generally equidistant from the ground truth boundary. To remove any bias from this seed placement on our comparative results, we produced an additional set of seeds by significantly eroding the original foreground seeds. The weights are set for all algorithms according to equation (4.1) with the value of β hand-optimized to provide the best results independently

for each algorithm. As only the order of the weights is taken into account in the MSF and power watershed algorithms, those two algorithms are independent of β. We used the color gradient given by $\sqrt{\max((R_i - R_j)^2, (G_i - G_j)^2, (B_i - B_j)^2)}$ for a color image of red, green, and blue components R, G, B. The normalization is achieved by dividing the gradient by the maximum value of the gradient over every edge in the graph G. Example seeds and segmentations for the five algorithms with the first seeding strategy are shown at the top of Figure 4.6(a) and with the second seeding strategy at the bottom of Figure 4.6(a).

	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 watershed ($q = 2$)	2.87	0.971	0.0245	0.210	**3.25**
Graph cuts	3.12	0.970	0.0249	0.212	**5**

Table 4.2: Mean errors computed between the segmentation masks and the ground truth images from the GrabCut database. Symmetrically eroded ground truth are used as seeds. The weight parameter β was set to 600 for Graph cuts, 700 for Random walker, and 900 for Shortest paths in order to maximize the performances of each algorithms.

	BE	RI	GCE	VoI	**Average rank**
Graph cuts	4.70	0.953	0.0380	0.284	**1**
Power watershed ($q = 2$)	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**

Table 4.3: Mean errors computed between the segmentation masks and the ground truth images from the GrabCut database. Asymmetrically eroded ground truth are used as seeds.

Tables 4.2 and 4.3 display the performance results for these algorithms. We quantify the error in the results using four different standard segmentation measures used in [219] , namely Boundary Error (BE), Rand Index (RI), Global Consistency Error (GCE), and Variation of Information (VoI). Good segmentation results are associated with low BE, high RI, low GCE and low VoI.

When segmenting with the first seeding strategy (the seeds contained in the Grabcut database) , the shortest path algorithm is the best performer because this algorithm do well when the seeds are placed roughly equidistant from the desired boundary [200] as they are with the first set of seeds.

The experiment on the second set of seeds shows that shortest paths is not robust to the seeds number and centering, as it is with this set of seeds the worst performer. Graph cuts

performs the best under this second seeding strategy but was the worst performers on the first one. Power watershed is in second position under the second seeding strategy, showing a good robustness to both seed quantity and location. It is interesting to note that with the first set of seeds, power watershed and maximum spanning forest results are quite similar, but with the asymmetrically eroded seeds, the power watershed results outperform the standard maximum spanning forest (watershed) results. The second set of seeds contained many areas where several contours could possibly be found, given the seeds. The merging operation of the power watershed reassembles undetermined areas and in performing the random walker on ambiguous regions, often generates a better labeling than the arbitrary labeling produced by Prim's or Kruskal algorithms when computing the maximum spanning forest (watershed).

4.5.2.2 Computation time

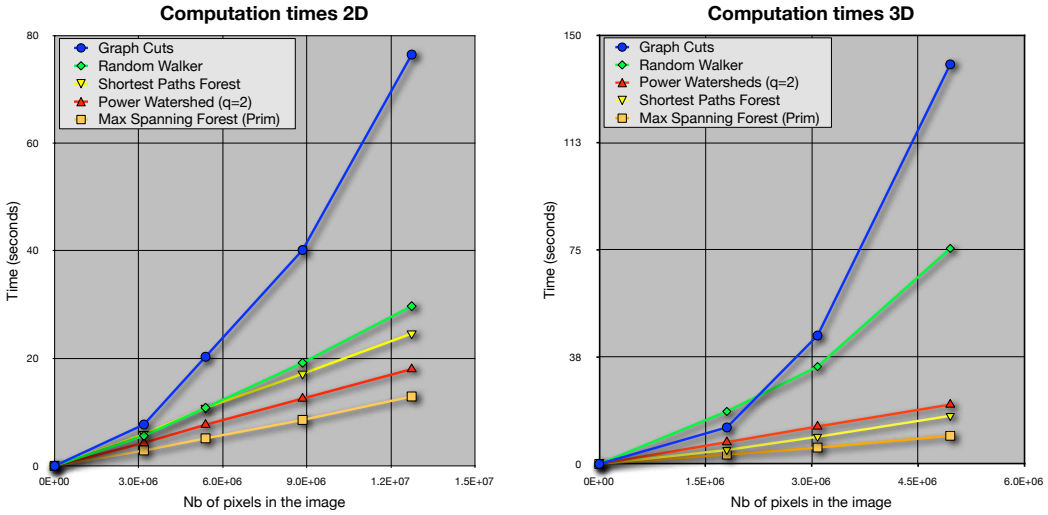


Figure 4.7: Computation time for 2D and 3D seeded image segmentation. For each dimension, the times were generated by segmenting the same image scaled down.

For all MSF algorithms, including the power watershed algorithm, only the order of the weights in taken in consideration for the segmentation. Also, there is no parameter choice to make for β, and no exponential to take in the weight function, so it is possible to sort the weights with a linear sort algorithm.

The worst-case complexity of the power watershed algorithm (obtained if all the edges weights are equal) is given by the cost of optimizing (4.4) for the given q. In best-case scenario (all weights have unique values), the power watershed algorithm has the same asymptotic complexity as the algorithm used for MSF computation, that is to say quasi-linear.

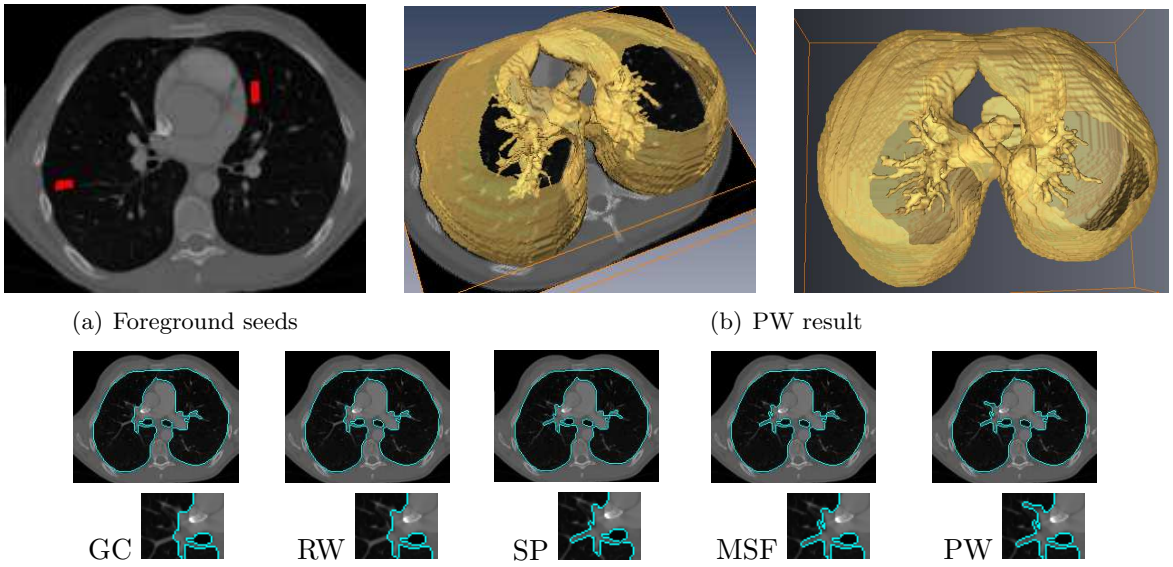


Figure 4.8: Example of 3D image segmentation. The foreground seed used for this image is a small rectangle in one slice of each lung, and the background seed is the frame of the image.

In practical applications where the plateaus are have size less than some fixed value K, then the complexity of the power watershed algorithm matches the quasi-linear complexity of the standard watershed algorithm. In our experiments in Section 4.5 with practical image segmentation tasks, the dependence of the computation time on image size of the power watersheds is very similar to the dependence in standard watersheds. For generating the computation time for the graph cuts algorithm, we used a software package provided at http://www.cs.ucl.ac.uk/staff/V.Kolmogorov/software.html and described in [33]. Our implementation of the shortest path is performed with a Fibonacci heap using double precision weights. For the implementation of Prim's algorithm, we used weights with integer precision, and red black tree as a sorting data structure. Finally, the random walker algorithm was implemented following the multigrid method described in [111] for 2D image segmentation, and a conjugate gradient descent method for 3D image segmentation.

4.5.2.3 Qualitative assessment

Unlike most watershed algorithms, the power watershed algorithm (with $q = 2$) has the property of providing a unique segmentation. Fig. 4.9 shows the behavior of the algorithm of our framework in presence of a plateau. Additionally, the power watershed (with $q = 2$) is not subject to the same shrinking bias exhibited by graph cuts segmentation. Fig. 4.10 compares the results of graph cuts and the power watershed on an example in which the

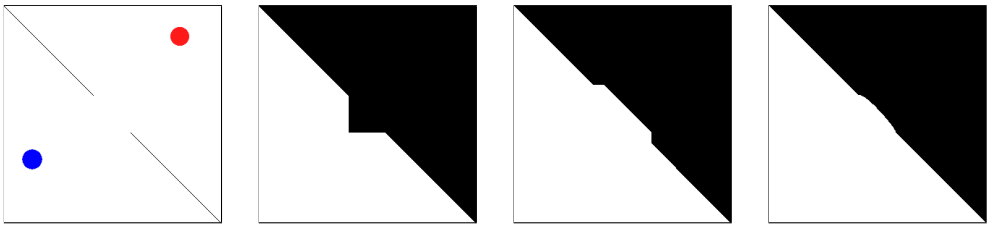


Figure 4.9: Segmentation of an artificial image.
(a) Image with foreground (red) and background (blue) seeds, (b) A segmentation obtained with graph cuts, (c) Segmentation obtained with Prim's algorithm for maximum spanning forest, and with a shortest path algorithm, (d) Segmentation obtained with random walker as well as power watershed with $q = 2$.

shrinking bias could substantially affect the result.

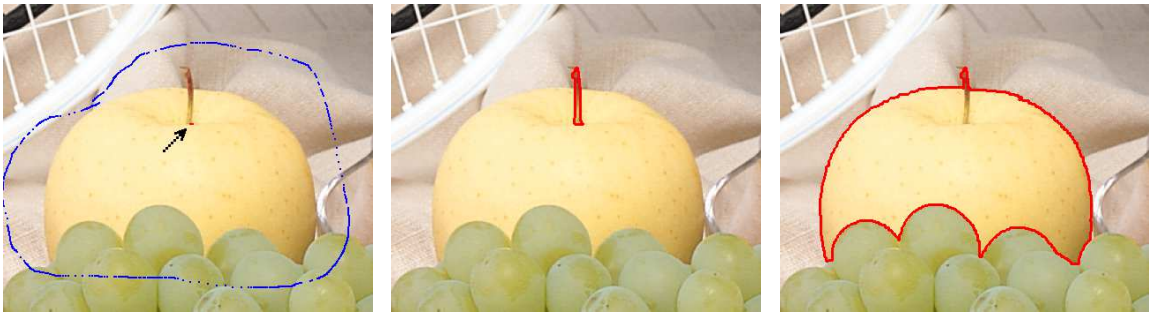


Figure 4.10: Example comparison of Graph cuts and Power watershed faced to a weak foreground seeds quantity.
(a) Seeds (the foreground seed is in red, indicated by an arrow), (b) Graph cuts segmentation result, (c) Power watershed ($q = 2$) result.

The power watershed is an MSF and therefore it inherits the standard properties of MSF, among others contrast invariance and scale invariance [5]. The contrast invariance property means that if a strictly monotonic transformation is applied to the weights of the graph, then the algorithm produces exactly the same result. This property is due to the fact that only the order or the weights is used to build a maximum spanning forest. The scale invariance property means that if we extend the image or graph in a way that does not change the relative ordering of weights, for example by linear interpolation, the result is invariant.

We summarize the performance of the algorithms of the framework:

- GC is a good fit for 2D image segmentation into two labels when the seeds are far away from the boundary (asymmetric seeding), but is too slow to be used for 3D

segmentation.

- SPF (geodesics) may be used if the object to segment is well centered around foreground and background seeds.
- The RW is efficient and performs well for both seeding strategies (equidistant seeds and strongly asymmetric seeds).
- Maximum Spanning Forest (watershed) algorithms provide better segmentations than SPF when seeds are not centered, and their fast computation time makes the algorithm suitable for 3D segmentation.
- The power watershed algorithm when $q = 2$ has the additional property of a well-defined behavior in presence of plateaus improving also the quality of the segmentation compared to standard MSF. As an MSF, it is still sensitive to leaking, but less so than traditional algorithms due to the random walk behavior. The computational speed of the power watershed is higher than all of the algorithms except the pure MSF.

4.6 Conclusion

In this chapter we clarified, simplified and extended the recent work connecting graph cuts and watershed [4, 5]. Extending the framework of [200], we have proposed a general framework encompassing graph cuts, random walker, shortest-path segmentation and watersheds. This connection allowed us to define a new family of optimal spanning forest for watershed segmentation algorithms using different exponents, which we termed the "power watershed". We produced an algorithm for computing the power watershed and our experiments showed that the power watershed with $q = 2$ retains the speed of the MSF algorithm while producing improved segmentations. In addition to providing a new image segmentation algorithm, this work also showed how unary terms could be employed with a standard watershed algorithm to improve segmentation performance.

Viewed as energy minimization algorithms, graph cuts, random walker and shortest paths have found many different applications in the computer vision field that go beyond image segmentation, such as stereo correspondence, optical flow and image restoration (e.g., [147, 196, 205]). By placing the optimal spanning forest algorithm for watersheds in the same energy minimization framework as these other algorithms, watershed algorithms may find new uses and applications within the computer vision field beyond its traditional domain of image segmentation. Due to the relative speed of the optimal spanning forest algorithms, we believe that it may be an attractive alternative to current systems in these other applications of energy minimization. The next chapter follows that idea, presenting three original applications of power watershed in computer vision.

Chapter 5

Power watershed applied to filtering, stereovision and surface reconstruction

In this chapter, we present three applications of the power watershed method to image restoration [1], stereovision and surface reconstruction from noisy sets of points [2]. This study paves the way for using the power watershed as a useful general-purpose minimizer in many different computer vision contexts.

5.1 Anisotropic diffusion

Not all problems in computer vision are convex: For instance, L_0 norm optimization such as seen in compressive sensing is not. In Chapter 4, a novel discrete framework encompassing many known segmentation methods was proposed: power watershed. We are interested in exploring the possibilities of this minimizer to solve problems other than segmentation, in particular with respect to unusual norm optimization. In this section we reformulate the problem of anisotropic diffusion as an L_0 optimization problem, and we show that power watersheds are able to optimize this energy quickly and effectively.

5.1.1 Introduction

The most common assumption about image noise is that the noise exhibits high frequency. Therefore, a natural choice of image filter is a lowpass filter which dampens power in the high-frequency range. Unfortunately, a lowpass filter has the undesirable effect of *blurring* object

[1] Published in *Proc. of ICIP 2010* [71]
[2] Published in *Proc. of ISMM 2011* [69]

boundaries, which also feature high frequencies. This dilemma was recognized early in the image processing literature and various approaches were proposed to solve it, with the goal of smoothing the image content *internal* to an object, but preserving image discontinuities across boundaries.

An approach to discontinuity-preserving filtering was proposed by Perona and Malik (PM) [176] who modeled image filtering as an anisotropic diffusion process that smoothed image intensities inside an object while preserving the intensity discontinuity between objects. Later, the anisotropic diffusion approach was shown by Black *et al.* [24] to be interpreted as a gradient descent method for optimizing a robust error function model which had been proposed previously. Anisotropic diffusion has been a very successful filtering algorithm, in part because it is easy to implement. However, this algorithm does have the problem of setting two parameters, the robust estimator parameter and the diffusion time. Practical usage of anisotropic diffusion requires a choice between long computation times or blurry boundaries.

Although the power watershed technique was introduced in the context of image segmentation in Chapter 4, the method can be used as an optimization method for some functionals. In this chapter we show that PW is well-suited to address the robust estimator filtering model and therefore provides an alternative to anisotropic diffusion for the optimization of this model. There are several advantages of using PW as compared to anisotropic diffusion in the context of optimizing the robust estimator filtering model. First, we do not require the robust estimator parameter. Second, we preserve discontinuities without blurring images. Third, the PW method is a very fast optimization procedure. Fourth, no time step needs to be determined (eliminating the risk of a divergent solution). Therefore, our optimization via PW removes the unfortunate compromise presented by anisotropic diffusion between fast optimization and sharp object boundaries.

Other models for discontinuity-preserving filtering are also well-known, including total variation [189], the Mumford-Shah functional [162] and the piecewise constant Mumford-Shah (Chan-Vese) model [213]. The total variation model is convex, allowing for a relatively efficient solution [50, 82] while the Mumford-Shah and Chan-Vese models are more difficult to optimize. However, recent work [45, 113] has provided more efficient optimization methods. Despite this progress in the optimization of other discontinuity-preserving filtering methods, very little work has been done to improve the speed/blur tradeoff that is necessary in the anisotropic diffusion algorithm. Rather than promoting one discontinuity-preserving model over another, we simply show here how the PW may be used to very efficiently optimize the robust estimator model underlying anisotropic diffusion such that we no longer have to choose between fast optimization and blurred boundaries. Our optimization will be achieved at the cost of a few iterations of the efficient PW, which is simple to implement.

5.1.2 Formulation

Black *et al.* showed that anisotropic diffusion could be viewed as the minimization of a robust estimator. We employ this same robust estimator formulation of anisotropic diffusion, but show that the power watershed may be used to iteratively optimize the formulation.

Given the classical continuous form

$$\frac{dx}{dt} = \nabla \cdot (g(\nabla x)\nabla x), \tag{5.1}$$

the anisotropic diffusion algorithm is defined using the discrete calculus notations as

$$\frac{dx}{dt} = A^\top g(Ax)Ax, \tag{5.2}$$

where x is the image intensities of a filtered image, and x at time 0 equals the input, unfiltered image, and A is, as usual, the incidence matrix of a graph. The function $g(x)$ is used to prevent blurring over edges and may be of any decreasing form. One possibility for $g(x)$ suggested in [176] is

$$g(x) = \exp\left(-\alpha x^2\right), \tag{5.3}$$

where α is a free parameter.

The typical method for solving the anisotropic diffusion equation in (5.2) is via a forward Euler method in which the iteration

$$x^{k+1} = x^k + dtA^\top g(Ax^k)Ax^k,$$

is applied, using a time step dt which is designed to obey the CFL conditions and ensure stability of the solution. In practice, the number of iterations applied to produce the filtered image is a free parameter which is set to define the desired level of smoothing.

Black *et al.* [24] showed that the anisotropic diffusion equation given by (5.2) could be viewed as the gradient of the energy

$$E(x) = \sigma(Ax), \tag{5.4}$$

where $\sigma(x)$ is a *robust estimator* or an *M-estimator*. A gradient of the Black *et al.* energy in (5.4) is given by

$$\frac{dE}{dx} = A^\top \sigma'(Ax)Ax. \tag{5.5}$$

Therefore, when $\sigma'(Ax) = g(Ax)$, then the gradient of the Black *et al.* energy, (5.5) is the same as the anisotropic diffusion equation in (5.2). In other words, *anisotropic diffusion is gradient descent minimization of* (5.4). Further, the $\sigma(z)$ corresponding to the PM weighting

function in (5.3) is given by the Welsch function $\sigma(z) = 1 - \exp\left(-\alpha z^2\right)$.

Black *et al.* explored the effect of different robust estimator forms of $\sigma(z)$. However, a common characteristic of all robust estimators is that they exhibit linear or quadratic growth near $z = 0$, but provide a nearly constant output as $z \to \infty$. The transition between quadratic growth and constant growth is controlled by the parameter α. If $\alpha \to \infty$, then the robust estimator ultimately achieves its constant growth phase for any $z \neq 0$. In this way, the $\sigma(z)$ function may be viewed as a differentiable approximation to the $||\cdot||_0$ norm in which $||0||_0 = 0$ and $||z||_0 = 1$ for any $z \neq 0$. Consequently, the anisotropic diffusion filtering algorithm may be viewed as gradient descent on an energy functional which is an approximation to the $||\cdot||_0$ norm.

Previous filtering algorithms that explicitly formulated filtering from the standpoint of optimizing the $||\cdot||_0$ norm took the form

$$E(x) = \sigma(Ax) + \lambda h(x, f), \tag{5.6}$$

in which f represents the intensities of the input unfiltered image, $h(x, f)$ represents a loss function enforcing data fidelity, and λ is a free parameter. When $h(x, f) = ||x - f||_2^2$, the gradient of (5.6) becomes $\frac{dE}{dx} = A^\top \sigma'(Ax)Ax + \lambda\,(f - x)$, which reaches a stable point when

$$\left(A^\top \sigma'(Ax)A + \lambda I\right) x = \lambda f. \tag{5.7}$$

Since (5.7) may be viewed as a *backward Euler* solution for the anisotropic diffusion equation (5.2) when $\lambda = \frac{1}{dt}$, then the anisotropic diffusion algorithm may be seen as the optimizing a robust estimator of the image gradient balanced against a loss function of the form $||x - f||_2^2$ where the tradeoff between gradient smoothness and data fidelity is governed by $\lambda = \frac{1}{dt}$. Therefore, by setting a fixed *time* for the anisotropic diffusion, we may view the solution obtained for this time as a steady-state optimization of our second energy functional (5.6) with a corresponding λ parameter. Viewing the time parameter as a loss function for the fidelity of the filtered image with the original, we may freely alter the loss function.

5.1.2.1 Anisotropic diffusion and L_0 norm

As said in the previous section, it is possible to alter the loss function. For example, we could optimize the fully robust energy

$$E(x) = \sigma(Ax) + \lambda\sigma\left(x - f\right),$$

with gradient given by

$$\frac{dE}{dx} = 2\alpha A^\top \sigma'(Ax)Ax + 2\alpha\lambda\sigma'(x-f)x.$$

If, at any iteration, we fix the values of x inside the robust error function, then we have

$$\frac{dE}{dx^{k+1}} = A^\top \sigma'(Ax^k)Ax^{k+1} + \lambda\sigma'(x^k - f)x^{k+1}. \tag{5.8}$$

This energy may be written as the steady-state optimization of the energy functional

$$E_{k+1} = {x^{k+1}}^\top A^\top \sigma'(Ax^k)Ax^{k+1} + \lambda {x^{k+1}}^\top \sigma'(x^k - f)x^{k+1},$$

that may also possibly be written

$$E_{k+1} = \sum_{e_{ij}} \sigma'(Ax^k)\left(x_i^{k+1} - x_j^{k+1}\right)^2 + \lambda \sum_{v_i} \sigma'(x^k - f)\left(x^{k+1} - f\right)^2. \tag{5.9}$$

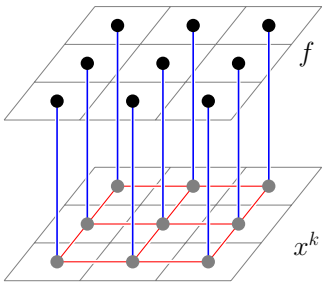


Figure 5.1: Graph necessary to build for applying the power watershed diffusion algorithm to a 3×3 image.
The unary weights in blue and pairwise weights in red are updated at each step.

Algorithm 8: Anisotropic diffusion using PW

Data: An image f, an initial solution x^0, $\lambda \in \mathbb{R}_+^*$
Result: A **filtered image** x^k
Set $k = 0$. Build the graph described in Figure 5.1.
repeat
  Generate the pairwise weights $\exp -(Ax^k)^2$, and unary weights $\exp -(x^k - f)^2$.
  Use PW with $y = f$ to optimize (5.10) to obtain x^{k+1}.
  $k = k + 1$;
until $||x^{k+1} - x^k||_2 < \epsilon$

This expression for the energy to compute a minimum step is of a form that may be optimized by the power watershed (PW) if the parameter $\alpha \to \infty$ for $g(x)$.

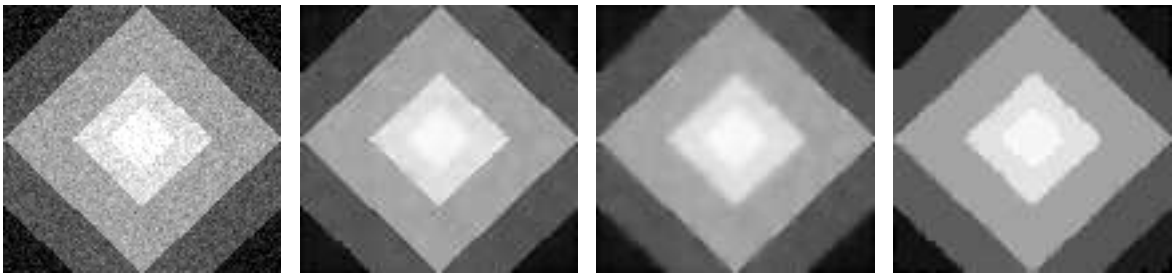
We recall that the generalized PW energy is given by

$$\min_x \sum_{e_{ij} \in E} {w_{ij}}^p |x_i - x_j|^q + \sum_{v_i \in V} {w_i}^p |x_i - y_i|^q \tag{5.10}$$

where y represents a measured configuration and x represents the target configuration. In this equation, the first term essentially forces x to vary smoothly within an object, and the second term enforces data fidelity. We can see that Eq. (5.9) is of the same form as Eq. 5.10, with $q = 2$, $p = \alpha$.

Therefore, we suggest to employ the filtering algorithm given in Algo. 8.

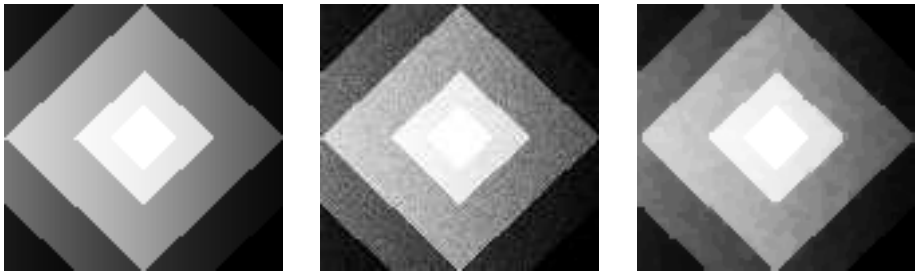
This technique can be initialized by the image f, or, in the case of very noisy images by a smoother version of the image, for instance by the application of a Gaussian or a median filter.



(a) Noisy image, PSNR = 24.24dB (b) PM, PSNR = 34.03dB (c) PM, PSNR = 30.46dB (d) PW, PSNR = 31.54dB

Figure 5.2: Comparison of Perona-Malik(PM), and power watershed(PW) algorithms for denoising a synthetic image.
(b) PM used with 80 iterations $\alpha = 0.0015$, leading to a good PSNR but with a few remaining isolated noisy pixels. (c) PM, best compromise found for this image to remove the isolated pixels with 50 iterations and $\alpha = 0.0005$. (d) Any median filtered image as initialization and $\lambda = 0.975$ results in a better PSNR while removing isolated noisy pixels.

In the initialization step of the algorithm, in image filtering applications, we need to build a graph in linking each pixel node with its neighbors, for example its 4 neighbors for 4-connectivity. The values of those nodes will be the x^k. Noting N the number of pixels in the image, we add N nodes to this graph in linking each pixel node v_i by an edge to the new node v_{i+N}. Those additional nodes are set to constant values of f. A description of the graph on a example is shown in Figure 5.1. The algorithm using power watershed for anisotropic diffusion is summarized in Alg. 8.



(a) Original image (b) Noisy image (c) PW result

Figure 5.3: Example of piecewise smooth image denoising using the power watershed algorithm.

		Perona-Malik		PW
Fig. 5.2, 104×100	Nb iter.	50	80	5
	Time (s)	0.19	0.30	0.17
Fig. 5.5, 250×300	Nb iter.	50	80	6
	Time (s)	1.38	2.14	1.78
Fig. 5.4, 299×364	Nb iter.	50	80	6
	Time (s)	1.95	3.08	2.43

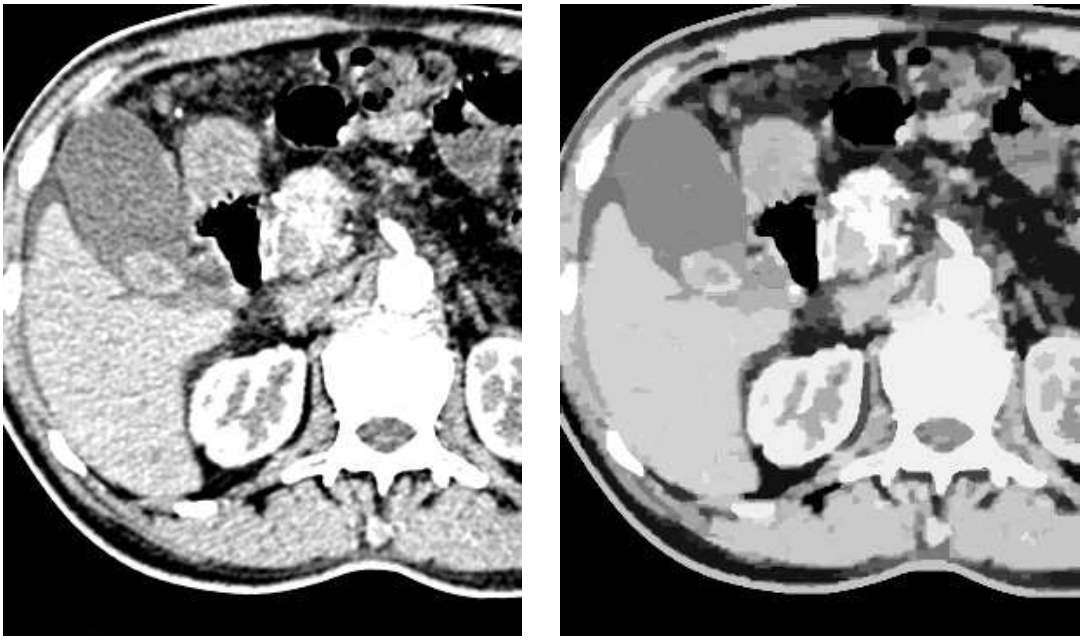
Table 5.1: Number of iterations of Perona-Malik, and power watershed algorithms on the image of Fig. 5.2, Fig. 5.5, and Fig. 5.4

5.1.3 Results

We now demonstrate the performance of the power watershed algorithm for anisotropic diffusion in presenting results on synthetic and real images.

A first, the test in Figure 5.2 is performed on a synthetic image corrupted with a Gaussian noise of standard deviation $\sigma^2 = 16$. For each result, we may compute the peak signal-to-noise ratio (PSNR) relatively to the original image. Typical values for the PSNR in denoising lie between 20 and 40 dB where higher is better. In comparison with PM algorithm, the power watershed algorithm tends to produce piecewise constant results. The result obtained at Figure 5.2(d) shows a good compromise between noise removing and edge preservation. However there is a choice to make for PM algorithm (Figure 5.2(b) and Figure 5.2(c)) between complete denoising and good contrast edge restoration.

Examples on real images in Figure 5.4 and Figure 5.5 show that the power watershed algorithm may be useful as a filtering step before segmentation. In terms of computation time, each iteration of the power watershed algorithm for anisotropic diffusion operates in quasi-linear time in practice. We noticed empirically that the convergence is very fast (fewer than 10 iterations). The result obtained at the first iteration is generally close to the final



(a) Original image (b) PW result

Figure 5.4: Filtering of a liver image by power watershed.
Here noise and small vessels are both removed, leading to a result which may be used as a first step before segmentation.

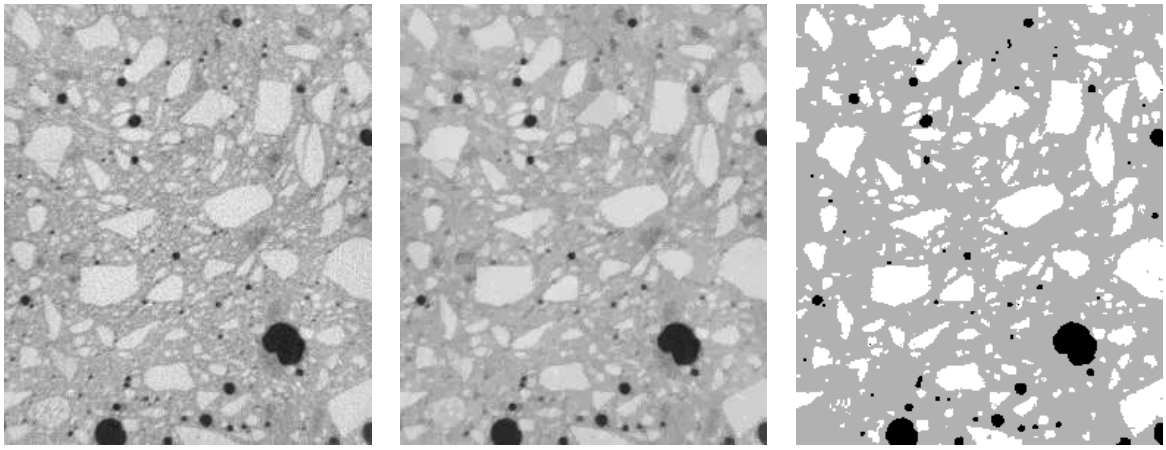
result as shown in Figure 5.5. We give some timing with necessary number of iterations to reach convergence at Table 5.1.

5.1.4 Conclusion

The power watershed (PW) algorithm was originally presented as a technique applied to image segmentation. In this section, we showed that PW provides an optimization procedure that allows us to optimize robust error measures which are operated in a particular parameter range.

Black *et al.* showed how the anisotropic diffusion method could be viewed as optimization of a robust error filtering model. We showed that PW could be used to optimize the same robust error filtering model, leading to an alternative optimization procedure.

An aspect of the PW optimization in this context is that it applies to the optimization of the robust error filtering model when operated in a particular parameter range which effectively models the image as piecewise constant. For cases in which this model was reasonable (such as the synthetic images), the algorithm exhibited strong denoising capabilities. When this denoising algorithm was applied to real images, the effect of imposing a piecewise con-



(a) Original image (b) PW result (6 iter.) (c) Segmentation

Figure 5.5: Segmentation of an image of concrete filtered using PW.
The segmentation in three classes (bubbles in black, stones in light grey) is useful for the study of the material's mechanical properties. The segmentation is obtained by two thresholds of the filtered image by PW.

stant model was to effectively quantize images into a small number of grayscale levels (Figure 5.4 and 5.5). This transformation of an image into a piecewise constant form may be helpful as a preprocessing step for segmentation or recognition [56].

5.2 Stereovision

Dense stereo is a technique for estimating a depth map of a scene using the point of view of several images. In effect, as illustrated in Figure 5.6, the disparity, which is the distance between two corresponding points of the same object in two different views, is a function of the depth.

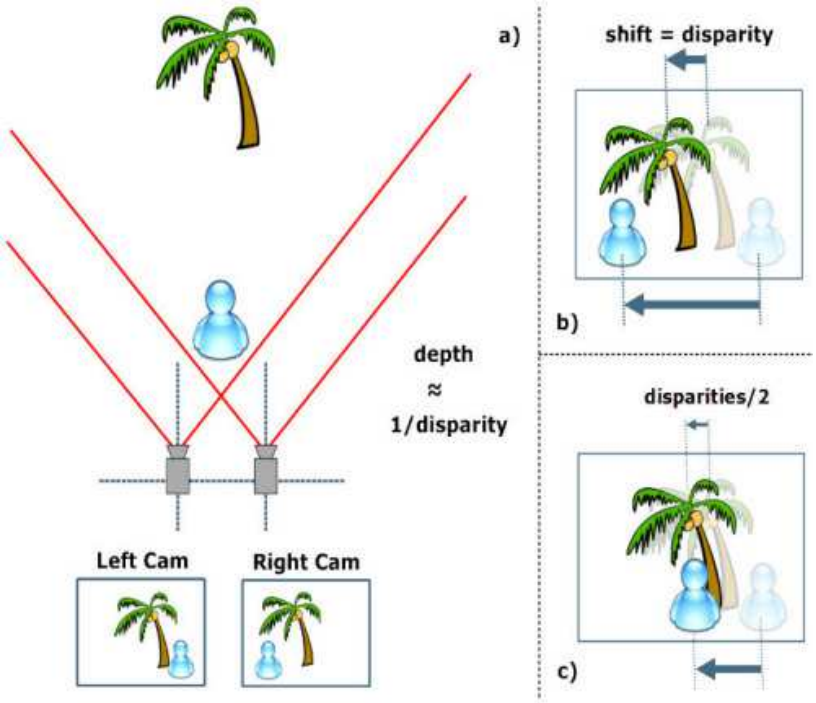


Figure 5.6: Illustration of the relationship between disparity between two views and scene depth estimation.
Figure from the IMEC's website (http://www.imec.be)

Among the large diversity of two-frame dense stereo techniques [192], global optimization methods such as graph cuts [218] and total variations minimization methods [223] have been quite successful. We present in this section an example showing the ability of power watershed to perform dense stereo as well.

In order to compute the disparity map from two aligned images, we can build a graph as illustrated in Figure 5.7. The nodes of the graph correspond to the disparity labels x to be recovered.

The nodes marked $l_1, ..., l_k$ correspond to labeled nodes, with values $0, ...k-1$ corresponding to distances (in pixels) for matching pixels. The unary weights w_{i,l_j} between a nodes v_i and a node l_j are given by similarity coefficients between blocks centered in the position i in the first image and blocks in position i shifted of l_j in the second image. The pairwise

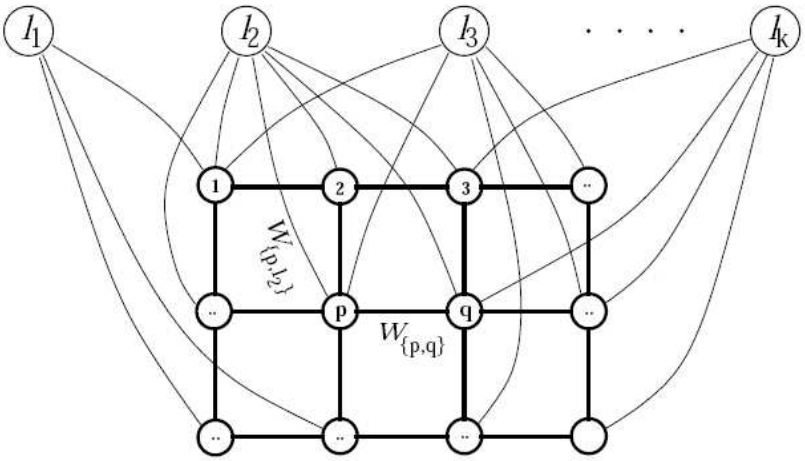


Figure 5.7: Possible graph construction for the stereo application.
The values of disparity x are computed over the nodes of the grid, given the labeled nodes on the top. Figure from O. Veksler.

weights have been chosen equal to a constant.

The result obtained by the power watershed algorithm is given in Figure 5.8.

Figure 5.8: Two images and the stereo reconstruction using a maximum spanning forest algorithm. This preliminary result is obtained using a different graph than the one described in Fig. 5.7, and is composed of different levels, each level corresponding to a different disparity.

This prospective result shows that power watershed is suitable to dense stereo estimation. Future work will determine if the global energy optimized with power watershed brings benefits to stereo applications. Much work remains to be done, exploiting the multilabel character of the power watershed.

5.3 Surface reconstruction

Surface reconstruction from a set of noisy point measurements has been a well studied problem for several decades. Recently, variational and discrete optimization approaches have been applied to solve it. These methods demonstrate good robustness to outliers thanks to a global energy minimization scheme. In this chapter, we use the power watershed algorithm, which is fast and robust to markers placement to produce smooth surfaces. Experiments show that the algorithm compares favorably in terms of speed, memory requirement and accuracy with existing algorithms.

5.3.1 Introduction

This chapter develops a watershed-based algorithm providing a global optimal solution to the surface reconstruction problem from a set of scattered points, under the hypothesis that the surface to recover is a close surface. Surface fitting is a challenging problem when dealing with data containing sparse noise, gaps, and outliers. The initial set of points may be for example acquired by several scans of an object (range scanning). In this context, regularization-based methods have been shown to be robust when the points lack connectivity, ordering information, and may be contaminated by noise. While there exist numerous explicit surface extraction techniques that estimate the exact positions of surface points, in this work we will focus on implicit surface representation. Implicit surfaces may be represented by level sets (*e.g.* [226]) or binary partitions (*e.g.* [146]).

Local methods for surface reconstruction including the MPU method [38] are sensitive to noise, as shown in the experiments of [131]. Among the recent global approaches, the Poisson method [135] and FFT-method [136] are more robust to noise, however they require orientation information.

The method of Jalba and Roerdink [131] makes it possible to avoid having to estimate orientation information. This is achieved by computing approximations of Coulomb potentials in a grid as an input for the convection method of [226]. We propose a different approach that allows us to perform a global optimization of the surface reconstruction problem without requiring the computation of such field as Coulomb potentials.

A generic optimization-based regularization formulation for shape fitting minimizes the total variation functional weighted by the distance function from the set of points P. More generally, given two positive numbers p and q, we consider an object indicator partition u

solution of

$$\begin{aligned} &\min_{u\in[0,1]} \int_{\Omega} w(z)^p |\nabla u(z)|^q \mathrm{dz} \\ &\text{subject to } u(z) = 0 \quad \forall z \in \Omega_{\text{in}}, \\ &\text{and } u(z) = 1 \quad \forall z \in \Omega_{\text{out}}, \end{aligned} \tag{5.11}$$

where Ω_{in} is the set of labels inside the surface and Ω_{out} is the set of labels outside the surface. The weight function w is defined at every point z in a grid as $w(z) = d_P(z)$, where $d_P(z)$ is the distance map from the points. When p is finite and $q = 1$, Eq. (5.11) leads to a binary solution u [203]. A solution can be deduced in the discrete setting using *e.g.* the network flow technique [97], also known as Graph cuts [36]. Augmenting path max flow implementations are fast and efficient in 2D but memory consuming as the connectivity increases, for instance in 3D. Lempitzky and Boykov [146] have overcome this problem by limiting the size of the search for a solution in the grid while still guaranteeing a global optimum. However their solution is based on restrictive assumptions assuming that the data points are provided with an estimate of the surface orientation. Furthermore, at low resolution, results exhibit metrication artifacts and look blocky, so a high resolution is essential for getting smooth results using the Graph cuts method.

We propose using the power watershed approach in order to provide a way to quickly obtain smooth surfaces at a high resolution without any need to pre-estimate the surface orientation. The power watershed method can be seen as an anisotropic discretization of (5.11) with $p \to \infty$ and $q = 2$. Although this technique was introduced in the context of image segmentation, as described in previous chapters, it may be employed as an optimization method for various functionals shown in the previous sections 5.1 and 5.2. In the present section we show that the power watershed method is well-suited to address the surface reconstruction problem. The idea of using watersheds for surface reconstruction from a set of points is quite natural. It can be seen as a an extension of the classical "coffee bean" segmentation example, where a watershed is applied on a filtered distance function to separate overlapping convex objects [20].

For further comparison, we also propose to examine a well-known weighted isotropic discretization of (5.11) (for $q = 1$) known as "Total Variation" (TV). We mention that there exist other TV-based approaches developed to overcome metrication artifacts, *e.g.* [107, 221]. Further comparison with these techniques will be the subject of future work. Finally, we will also show that our algorithm compares favorably with existing surface reconstruction algorithms [7, 38, 131, 135, 136].

5.3.2 Method

For solving the surface reconstruction problem, we first place the points cloud onto a regular grid. Our method aims to label the nodes of the grid as an indicator of the object to reconstruct. Since the power watershed is defined on a graph, we begin by casting the surface reconstruction problem formulation in discrete terms.

Given foreground F and background B node values (also called seeds), and p, q two real positive values, the energy presented for binary segmentation in [70] is a discretization of (5.11)

The definition of the weights for surface reconstruction from a set of points P is based on the construction of a discrete distance map d_P, in a grid bounding the set of points.

$$w_{ij} = \min\left(d_P(i), d_P(j)\right), \tag{5.12}$$

where $d_P(i)$ is the discrete Euclidean distance between the node i and the set of points P. We recall that exact Euclidean discrete distance map may be obtained in linear time using the algorithm of Hirata [125], and that high-quality ordered algorithms also exist [186] . The background seeds may simply correspond to the frame, or bounding box of the lattice. The foreground seeds can be given by the maxima of the distance function that are not connected to the frame. The distance map may be previously filtered, for example using an attribute filter to obtain more robust markers [41, 165].

As we illustrate in the remainder, the energy defined in (5.10) essentially forces x to remain smooth within the object, while allowing it to vary quickly close to point clusters near the boundary of the object. The data constraints enforce fidelity of x to a specified configuration, taking the values zero and one as the reconstructed object indicator. Observe that the values of x may not necessarily be binary when the value of q is strictly greater than one, which is a positive point for the surface reconstruction problem as we will further explain in this work.

The different values of p and q lead to different algorithms for optimizing the energy. When the power of the weight, p, is finite, and the exponent $q = 1$, we recover the Graph cuts energy which can be optimized by a max flow algorithm. When $q = 2$, we obtain a combinatorial Dirichlet problem also known as the Random walker problem [110]. As described in [70, 72], when the exponent p tends toward infinity, the cut obtained when minimizing the energy is a watershed cut [77], which has been shown to be equivalent to Maximum Spanning Forests (MSF). Furthermore, an algorithm is presented to optimize the unique watershed that optimizes the energy for $q = 2$ and $p \to \infty$.

This set of parameters $q = 2$ and $p \to \infty$ is particularly interesting :

1. The power watershed algorithm has a well-defined behavior in the absence or lack of

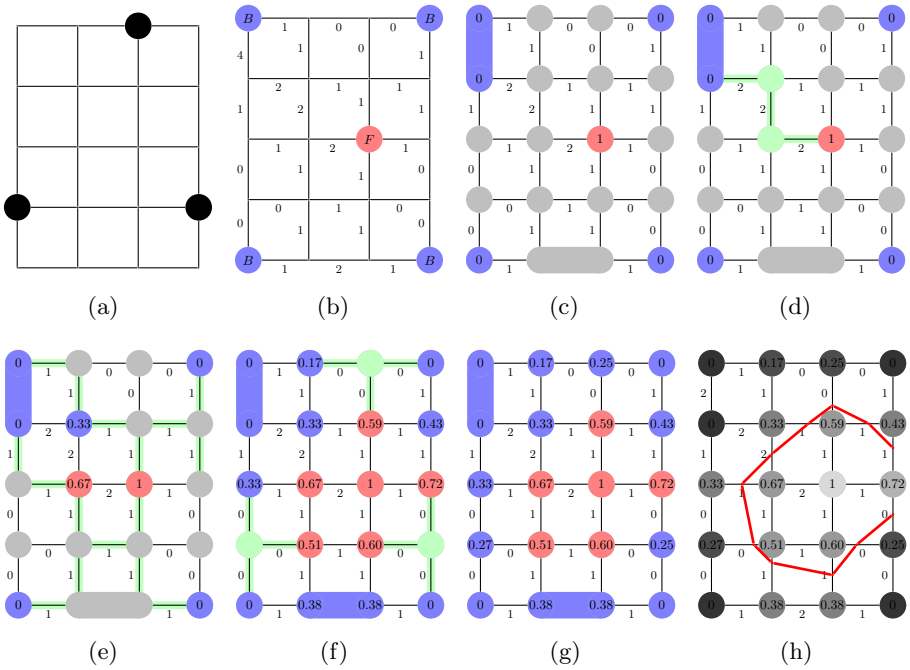
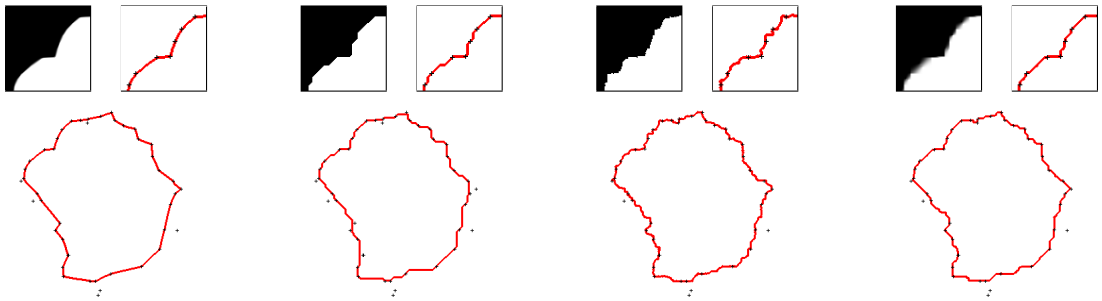


Figure 5.9: Illustration of surface extraction from dots using PW on a small graph. (a) Three dots in a 4×5 lattice. (b) Associated lattice weighted by a distance map from the dots according to (5.12) and squared, with Foreground and Background seeds. Note that only the ordering of the weights counts in the power watershed algorithm, thus the squared distance map can be used directly and produces the same solution as if the weights in (5.12) were used. (c) First steps of the power watershed algorithm to optimize (5.10) in the case $q = 2$ and $p \to \infty$. Nodes having a maximum weight are merged. (d) A plateau of weight 2 (in green) including different seeded nodes is encountered. The Random walker algorithm is applied to label the nodes on the plateau. (e,f) New plateaus of weight 1 and 0 are encountered, the Random walker algorithm is applied, (g) Final labeling x solution of (5.10). (h) The isocontour is represented in red.

weight information (presence of plateaus). An example is shown at Figure (5.9).

2. The worst-case complexity of the power watershed algorithm in the case $p \to \infty$ is given by the cost of optimizing (5.10) for the given q. In the best-case scenario (all weights have unique values), the power watershed algorithm has the same asymptotic complexity as the algorithm used for a MSF computation (quasi-linear) (See [57] for more details). In practical applications where the plateaus have a size less than some fixed value K, then the complexity of the power watershed algorithm matches the quasi-linear complexity of the standard watershed algorithm.

Although the PW algorithm is fast, it can be further accelerated for the specific case of weight defined according to a distance map (5.12). Given the foreground and background seeds, we define a narrow band $\mathcal{S}$ as the set obtained by thresholding the distance map d_P

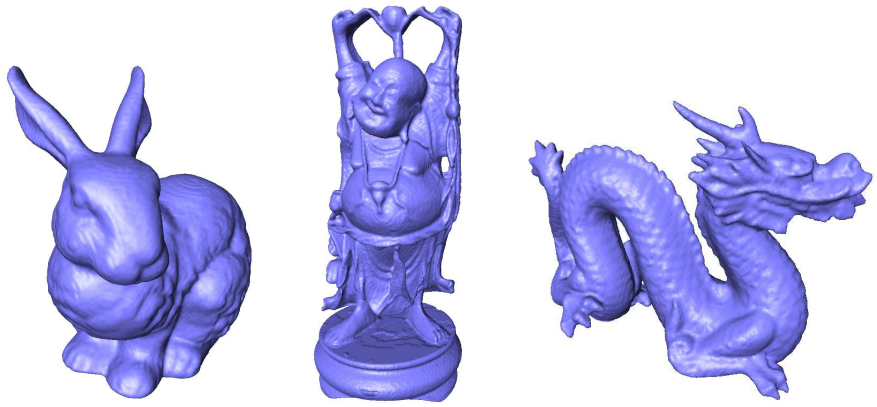


(a) Total variation result (b) Graph cuts result (c) Max Spanning Forest result (d) Power watershed result

Figure 5.10: Comparison of surface reconstruction from a set of points in 2D, using the total variation method, Graph cuts, a maximum spanning forest algorithm and finally the power watershed algorithm.

with the the smallest threshold T_S such that the connected components are divided between at least an interior (foreground) and an exterior (background). We then compute the PW only on this incomplete distance map, which saves both time and memory. In practice, it is possible to avoid computing an exact distance map on the full grid, using for instance an ordered algorithm propagating from the point cloud [186].

Recall that the exterior seed is connected to the frame of the image, so this is a simple unambiguous connectivity criterion. Applying PW on $\mathcal{S}$ is guaranteed to provide the same (unique) global optimizer of the energy as the solution on the full grid, because the connectivity criterion ensures this computation yields a Jordan curve (2D) or surface (3D) separating foreground and background seeds and passing through only already-computed distance values. Performing the PW computation on the full map would not change this result because no distance weight would be lowered, and so no new surface with a lower weight could be found on this full map. Conversely, thresholding the distance map at a lower value than T_S *would* change the result, as any Jordan curve or surface separating foreground and background seeds and computed on this map would necessarily cross some nodes where the distance map had not been computed. Note this does not mean that this thresholding criterion is necessary and sufficient for optimal computation and least memory usage, as an adaptive threshold depending on the local point density could be used instead. However, we leave it as future work to find a better criterion than the simple global threshold.



(a) (b) (c)

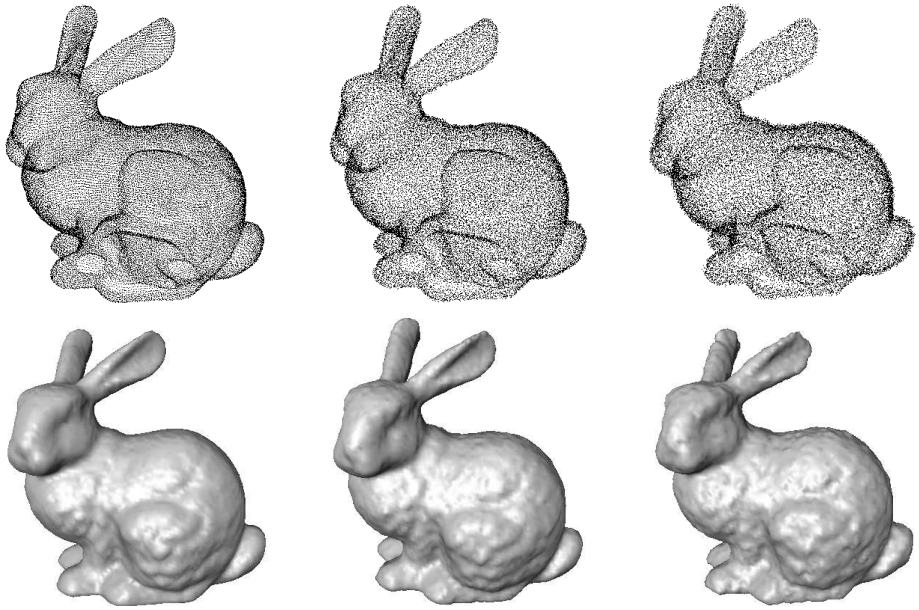
Figure 5.11: Power watershed results obtained from points clouds.
(a) Total size of scans : 362272 points, Grid size : $234 \times 297 \times 301$. (b) Number of scans used : 341072 points, Grid size : $275 \times 276 \times 668$ (c) Total size of scans : 2748318 points, Grid size $382 \times 270 \times 171$.

5.3.3 Results and comparative evaluation

5.3.3.1 Comparison with Graph cuts and Total Variation

We now demonstrate the performance of the power watershed algorithm for surface reconstruction with respect to two graph-based methods discretizing the energy defined in equation (5.11), namely the weighted total variation (TV) and the Graph cuts (GC) method.

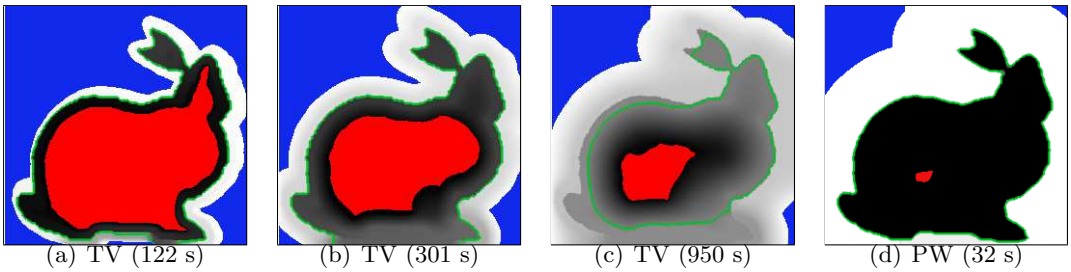
Our first experiment consists of finding a contour fitting sparse and noisy dots in a 2D plane. Figure 5.10 compares the result of TV minimization, the Graph cuts result, a maximum spanning forest result, and the result obtained with power watershed algorithm ($q = 2$, $p \to \infty$). We observe that all resulting contours are excluding outliers. The Graph cuts results demonstrate that this algorithm is less sensitive to noise, but the contours are blocky because the obtained object indicators are binary. We note that a post-processing step for producing a smooth isosurface from such a binary object reconstruction has recently been proposed by Lempitzky [145]. The TV contour is the smoothest one, which may be an unwanted effect for rendering details in surfaces. Thus, the TV method requires us to adapt the parameter p in the exponent of the weights to the desired smoothness of the surface. The maximum spanning forest result passes through most points and the contour looks noisy. The power watershed result demonstrates good performance for fitting the dots, while avoiding both blocky contours and noise. This good performance is due to the presence of interwoven plateaus in the distance map. During the execution of the algorithm, the Random walker (RW) algorithm is called several times around the dots, resulting in a smooth output x. An isocontour or isosurface computation at the 0.5 level is thus providing smooth contours



$\sigma = 0$ Error $= 0$ | $\sigma = 0.003$ Error $= 2.7 \times 10^{-3}$ | $\sigma = 0.005$ Error $= 4.0 \times 10^{-3}$

Figure 5.12: Power watershed results obtained from noisy points clouds, corrupted by Gaussian noise of variance σ.

The error was computed as the average distance between the obtained isosurface points to the original point cloud. The error is given in percentages of the diagonal of the bounding box of the data points.



(a) TV (122 s) (b) TV (301 s) (c) TV (950 s) (d) PW (32 s)

Figure 5.13: Robustness to seed quantity.
This figure shows slices of solutions obtained for reconstructing the bunny surface, obtained on a grid of size $251 \times 248 \times 195$, using different seeding strategies. The interior seed is colored in red, and the background seed in blue, and resulting isolines in green. (a,b,c) : Results obtained with TV minimization, and (d) Result obtained with Power Watershed.

compared to the binary results obtained with Graph cuts or maximum spanning forests.

Figure 5.11 shows surfaces reconstructed from noisy scanned dot sets using the power watershed (PW) algorithm. We used the coordinates of points acquired from scans of several 3D shapes (bunny, Buddha) from the Stanford database available online [2] In our experiments, we embedded those points in 6-connected grids. Quantitative comparisons for the fitting quality are difficult because not all data points are required to be part of the surface. However, we show in Figure 5.12 that the power watershed method is producing reasonable results even if the point cloud is corrupted by Gaussian noise. We also compared our results to the results obtained using TV minimization and Graph cuts at Figure 5.14. We can observe that the surface obtained with the Graph cuts method is quite blocky. Using the same rendering method to render the output x minimizing the power watershed energy, the power watershed algorithm obtains a smoother surface showing significantly more details. Figure 5.13 shows that the power watershed method performs well even with a small amount of seeds. In contrast, the total variation method requires a large amount of seeds placed close to the searched surface.

5.3.3.2 Computation times and memory requirements

In our comparisons, we used the C++ software library of Lemptitzky and Boykov available online which implements the touch-expand algorithm that minimizes the GC energy. For that purpose, the touch-expand algorithm calls a max flow algorithm on a partial graph, a band around the points which is extended when the solution touches the boundary of the band. We also compare to a fast TV-based solver often called split-Bregman algorithm [108] but identical in principle to the Alternating Direction Method of Multipliers [104, 195]. This TV solver is implemented in C and may be called from Matlab. Finally, we also implemented the

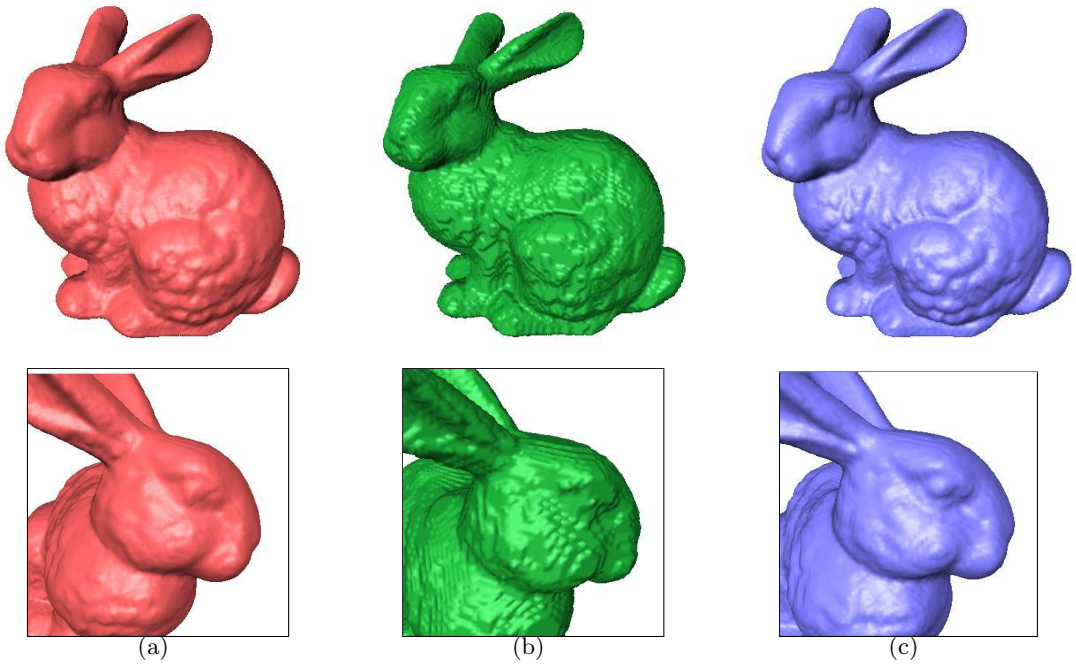


Figure 5.14: Comparison of TV, GC, and PW resulting surfaces from the Stanford bunny dataset.
Grid size : $234 \times 297 \times 301$. (a) Total variation minimization result, (b) Graph cuts result, (c) power watershed result. Isosurfaces at 0.5 have been extracted on all results, and were downsampled by 2 to render the surfaces.

PW method in C. Computation times of the compared algorithms for the bunny and Buddha data sets are provided on Table 5.2. The three methods have large memory requirements when applied to the full grid. However, the graph cuts and power watershed banded methods are less memory intensive. For the bunny dataset, the touch-expand algorithm optimizing the Graph cuts energy only needs to allocate 3.6% of the full grid size. The solution space for the bunny dataset is much larger for our power watershed algorithm and reaches 33% of the full grid, because some points are widely spaced out. Thus, the touch expand algorithm is faster than the power watershed method. However, considering the ratio time per number of grid node, the power watershed approach is faster. The TV approach has currently no guarantee to produce a global optimum when called on a banded graph, so the TV algorithm was implemented only on a full grid. On the Buddha dataset for example, given a $207 \times 505 \times 207$ grid, the TV-based method requires 3G of RAM.

Method	Grid size	Grid used	Peak memory	Time	Time / Nb voxels
GC	$234 \times 257 \times 301$	3.6%	212	9	1.4×10^{-5}
PW	$234 \times 257 \times 301$	31%	1180	51	9.1×10^{-6}
TV	$195 \times 248 \times 251$	100%	900	122	1.0×10^{-5}

Table 5.2: Timing experiments (in seconds) for reconstructing the bunny surface using an PC with a 3GHz Intel dual-core processor and 2G of RAM. The memory requirements are given in Mega Octets, and the computation times in seconds. Note that the seeds used are different for the three methods. For TV and PW, the seeds are imposed as hard constraints and are located far from the point cloud (see Figure 5.13). The GC touch expand method uses soft constraints seeds computed from the normals given in input with the point cloud. We observe that Graph cuts are faster than PW and TV. However, note that the solution is computed on only 3.6% of the full grid, ten times less than PW. On the contrary, the ratio time per voxel shows that the PW could be obtained in a short time using a similar graph reduction.

5.3.3.3 Comparison with some other approaches

Following the quantitative information given in [131], we compare the performances of power watershed for the Stanford bunny reconstruction with different methods [7, 38, 131, 135, 136]. All these methods, including the power watershed, reconstruct a surface close to the Bunny set of dots, with an error comprised between 2×10^{-4} and 6×10^{-4}. In terms of computation time, the Power Crust method [7] is about 10 times slower, and the Poisson method [135] 3 times slower than our PW approach. Although the FFT [136], MPU [38] and Hoppe *et al* [127] methods are fast, the FFT method suffers from large memory requirements limiting the grid resolution, and Hoppe *et al* and MPU methods produces artifacts in the presence of noise (See [131]). The method of Jalba and Roerdink [131], based on Coulomb potentials, uses 4 times less memory than our power watershed method, but is 5 to 10 times slower on a CPU and is still slower using a GPU. Furthermore our PW is more flexible in the choice of the markers. In our experiments, the amount of markers is not very large as shown in Figure 5.13, but a strategy using larger seeds could be employed to reduce the size of the solution space and the computation time.

5.3.4 Conclusion

The power watershed method can be used to efficiently produce surfaces fitting noisy measurements. Contrary to standard watershed algorithms and the Graph cuts approach, the unique solution provided by the power watershed is not binary, resulting in the reconstruction of both smooth and detailed surfaces. In addition, this method is fast and not limited by large memory requirements, when using a restriction of the solution space. Finally, in com-

parison with other methods, our power watershed is robust to seed placement, and requires fewer parameters to be set. In practice, when using the power watershed method, close-fitting markers are not as mandatory as in other methods. Future work will follow several directions: memory and computation times improvements are still achievable, in particular by improving the touch-expand idea used in the Graph cuts optimization and adapting it for power watershed. The power watershed energy could also be modified to add some priors to the reconstructed surface, such as local orientation. Finally, we hope to demonstrate the efficiency of the power watershed technique for solving related problems such as multiview reconstruction.

We used the energy minimization aspect of PW to expand the traditional use of watersheds from a segmentation algorithm to image filtering, stereovision, and surface reconstruction algorithms. Although our algorithm optimizes the filtering objective or disparities as a real-valued optimization problem, we showed that we still retain the speed of the watershed algorithm. However, the limit of PW as an energy optimization strategy remains unclear — What energy functions can be minimized via power watersheds? Future work will address this issue and continue to demonstrate applications which can benefit from the speed of power watershed.

Chapter 6

Conclusion

In this thesis we have proposed several graph-based variational approaches and have provided theoretical links of significant importance between existing and new methods. We summarize in Table 6.1 the diverse regularization terms studied and developed in this work. It is not exhaustive but it illustrates the different energies from a unified point of view. The methods we developed were successfully applied to such diverse problems as image segmentation, restoration, stereovision and surface reconstruction.

6.1 Contributions

Contributions – We have presented a new combinatorial formulation of the continuous maximum flow problem and a solution using an interior point primal-dual algorithm. This formulation allows us to optimize the maximum flow problem as well as its dual for which we provide an interpretation as a minimal surface problem. This new combinatorial expression of continuous max-flow avoids blockiness artifacts compared to graph cuts. Furthermore, the formulation of CMF on a graph reveals that it is actually the fact that the capacity constraints are applied to point-wise norms of flow *through nodes* that allows us to avoid metrication errors, as opposed to the conventional graph cut capacity constraint *through individual edges*. Additionally, contrary to graph cuts, we showed that CCMF accurately estimates the true boundary length of objects. Finally, unlike finite-element based methods such as AT-CMF, the CCMF formulation is expressed for arbitrary graphs, and thus can be employed in a large variety of tasks such as classification.

We have also compared our algorithm to known weighted combinatorial TV minimization methods. We have shown that our results are generally better for segmentation than TV-based methods, and that our implementation on CPU can be much faster and more predictable. Specifically, the number of iterations required for convergence does not vary much for CCMF, whereas it can vary of more than a factor of ten for the fastest TV-based methods.

Method	Energy formulation	Details
Max flow / Graph cuts	$\underset{x\in\{0,1\}^n}{\arg\min}\, w^\top \lvert Ax\rvert + \mathcal{D}(x)$	See Sect. 1.2.2
Combinatorial Continuous Max Flow	$\underset{\substack{\nu\in\{\delta,1+\delta\}\\ \lambda>0,\ \delta\in\mathbb{R}_+}}{\arg\min}\ \lambda^\top g^2 + (\frac{1}{4}\cdot/(\lvert A\rvert\lambda))^\top (A\nu)^2 + \mathcal{D}(\nu,\lambda)$	See Sect. 2.2.3
Combinatorial Total Variations	$\underset{x}{\arg\min}\, g^\top (\lvert A\rvert^\top (Ax)^2)^{\frac{1}{2}} + \mathcal{D}(x)$	See Sect. 2.3.3
Dual Constrained Total Variations	$\underset{x}{\arg\min}\, \underset{F\in C}{\max}\, F^\top (Ax) + \mathcal{D}(x)$	See Sect. 3.2
Combinatorial Dirichlet	$\underset{x}{\arg\min}\, (Ax)^\top \mathrm{diag}(w)(Ax) + \mathcal{D}(x)$	See Sect. 1.2.3
Shortest Path Forest (geodesics)	$\lim_{q\to\infty} \underset{x}{\arg\min}\, w^{\top q} \lvert Ax\rvert^q + \mathcal{D}(x)$	See [200]
Power watershed	$\lim_{p\to\infty} \underset{x}{\arg\min}\, w^{\top p} \lvert Ax\rvert^q + \mathcal{D}(x)$	See Sect. 4.3.2
ℓ_0-norm anisotropic diffusion	$\lim_{\alpha\to\infty} \underset{x}{\arg\min}\, \sigma_\alpha(Ax) + \mathcal{D}(x)$	See Sect. 5.1.2.1

Table 6.1: The different combinatorial regularization terms studied and proposed in this thesis. The solution x is a labeling defined on a graph with incidence matrix A, edge weights w and node weights g that are assumed to be positive. The data fidelity term is noted $\mathcal{D}$. More details about each approach are given in the designated sections and references.

From a theoretical view point, the deep study of the relationships between CCMF and combinatorial TV reveals that, in contrast with expectations from their duality under restrictive assumptions in the continuous domain, their duality relationship is only weak in the combinatorial setting.

Dual Constrained TV-based regularization for image restoration – Inspired from the CCMF formulation, the energy formulation has been adapted and extended to multi-label problems. Constraining the dual projection variable in the primal dual total variation problem provides a more flexible way of penalization of variations. Fast parallel proximal optimization tools have been tested and adapted for the optimization of this problem. Experimental results show an improvement of the approach in term of restoration quality and contrast preservation.

Power watershed: A unifying graph-based optimization framework – We have introduced a framework that generalizes several state-of-the-art graph-based segmentation algorithms, namely graph cuts, random walker, shortest paths, and watershed. This generalization allowed us to exhibit a new family of watershed algorithms using different exponents, which we term "Power watershed". The algorithm produced for computing power watershed is a novel, globally optimal optimization method. Our experiments showed that the case $q = 2$ retains the speed of efficient watershed algorithms while providing improved segmentations. Furthermore, this case exhibits the interesting properties of uniqueness of the solution, and possible application for fast, exact global optimization of multi-label problems.

Placing the maximum spanning forest for watershed in the same energy minimization framework than state-of-the-art optimization algorithms – used in a large variety of applications in computer vision – opens the way to new use of watershed within the computer vision field. Indeed, we have shown that it may be used beyond its traditional domain of image segmentation. Specifically, we have demonstrated how to use power watershed as a solver for image filtering optimizing an L_0 norm energy. We also presented an example of stereo-reconstruction using power watershed, and performed fast and smooth surface reconstruction from a noisy cloud of 3D points.

6.2 Future work

The work presented in this thesis offers many avenues for further investigation.

Flow based methods – The current optimization of the CCMF problem – performed by an interior point method – is faster than the existing continuous methods. However, a drawback of our current implementation of the interior point method is that the resolutions of linear systems can be expensive for 3D tasks. Future improvements could involve the implementation of preconditioned conjugate gradient for the linear system resolution, or the use of fast first order methods. In the context of restoration using DCTV, diverse formulations of the constraints on the projection variable remain to be explored.

Power watersheds – Future work develop along several directions. One direction is the further improvement of image segmentation algorithms using power watersheds as a component to larger systems in a similar manner as graph cuts, random walker and shortest paths have been used. Additionally, we hope to use the common framework for these algorithms to leverage existing ideas from the watershed literature into these other algorithms. In particular, hierarchical schemes [11, 121, 163, 164, 167] look like an interesting topic that can

take advantage of the power watershed uniqueness. A second direction for future work will also include the characterization of the class of energy that may possibly be optimized in this framework. For example, the possibility of using negative weights could be explored, with potential applications in texture analysis, or segmentation with a curvature constraint [88].

Ultimately, we hope to employ power watersheds as a fast, effective alternative to the energy minimization algorithms that currently pervade the wide variety of applications in computer vision. The efficiency of the power watershed for large datasets is suitable to many problems in vision (and beyond), such as stereo-vision, inpainting, video segmentation, optical flow, and more.

Finally, further developments may be achieved by using extra information to obtain better results. For example shape constrained segmentation [212], co-segmentation methods [126], and segmentation of moving objects. Extra information coming from machine learning could bring benefits. For example, the convolutional networks of Lecun *et al.* [220] simulate the learning of a neuronal network by mixing thousands of simple convolution operations, leading to and effective real-time object recognition and classification. However, there are still possible improvements for these methods, which provide only rough estimates of objects contours. It will be highly interesting to study the potential benefit of graph-based approaches for image recognition in the learning process.

Appendix

A.1 Combinatorial continuous max flow

A.1.1 Proof of Property 1

In a transport graph G with m edges, n nodes, we define a vector c of $\mathbb{R}^m$, composed of zeros except for the element corresponding to the source/sink edge which is -1. Let λ and ν be two vectors of $\mathbb{R}^n$. The CCMF problem

$$\begin{aligned} \max_{F\in\mathbb{R}^m} \quad & -c^\top F, \\ \text{s. t.} \quad & A^\top F = 0, \\ & |A^\top| F^2 \leq g^2. \end{aligned} \tag{1}$$

has for dual

$$\begin{aligned} \min_{\lambda\in\mathbb{R}^n,\ \nu\in\mathbb{R}^n} \quad & \lambda^\top g^2 + \tfrac{1}{4}\Big(\mathbf{1}^n \cdot /(|A|\lambda)\Big)^\top \Big((c + A\nu)^2\Big), \\ \text{s. t.} \quad & \lambda \geq 0, \end{aligned} \tag{2}$$

equivalently written in (2.10), and the optimal solution (F^*, λ^*, ν^*) verifies

$$\max_F c^\top F = c^\top F^* = 2{\lambda^*}^\top g^2, \tag{3}$$

and the n following equalities

$$\lambda^* \cdot |A^\top| \Big((c + A\nu^*) \cdot /(|A|\lambda^*)\Big)^2 = 4\lambda^* \cdot g^2. \tag{4}$$

Proof. The Lagrangian of (1), whose objective is equivalent to $\min_F c^\top F$, is

$$L(F, \lambda, \nu) = (c^\top + \nu^\top A^\top)F + \lambda^\top |A^\top| F^2 - \lambda^\top g^2,$$

with $\lambda \in \mathbb{R}^n$ and $\nu \in \mathbb{R}^n$ two Lagrange multipliers. We have to find F in the dual function is such as $\nabla_F L(F, \lambda, \nu) = 0$.

$$\nabla_F L(F, \lambda, \nu) = c + A\nu + 2|A|\lambda \cdot F = 0 \;\Leftrightarrow\; F = (c + A\nu) \cdot /(-2|A|\lambda). \tag{5}$$

Substituting F in the Lagrangian function, we obtain

$$L(\lambda, \nu) = (c^\top + \nu^\top A^\top)\Big((c + A\nu) \cdot /(-2|A|\lambda)\Big) + \lambda^\top |A^\top| \Big((c + A\nu) \cdot /(-2|A|\lambda)\Big)^2 - \lambda^\top g^2.$$

The Lagrangian may be simplified as:

$$L(\lambda, \nu) = -\Big(\mathbf{1}^n \cdot /(4|A|\lambda)\Big)^\top \Big((c + A\nu)^2\Big) - \lambda^\top g^2.$$

The dual of (1) is also

$$\begin{aligned} \min_{\lambda,\nu} \quad & \lambda^\top g^2 + \tfrac{1}{4}\Big(\mathbf{1}^n \cdot /(|A|\lambda)\Big)^\top \Big((c + A\nu)^2\Big), \\ \text{s. t.} \quad & \lambda \geq 0. \end{aligned} \tag{6}$$

Furthermore, the KKT optimality conditions give

$$\lambda^* \cdot \Big(|A^\top| F^{*2} - g^2\Big) = 0. \tag{7}$$

Using the expression of F^* from equation (5), and substituting it in (7), we obtain

$$\lambda^* \cdot |A^\top| \Big((c + A\nu^*) \cdot /(|A|\lambda^*)\Big)^2 = 4\lambda^* \cdot g^2.$$

Taking the sum of right hand side and left hand side,

$$(c + A\nu^*)^2 \cdot /(|A|\lambda^*) = 4{\lambda^*}^\top g^2. \tag{8}$$

Finally, if we substitute (8) in the dual expression (2), we have

$$\max_F \; c^\top F = c^\top F^* = 2{\lambda^*}^\top g^2.$$

□

A.1.2 Weak duality of CCMF and CTV

Even if CCMF is not dual with CTV, the two problems are weakly dual.

In the combinatorial setting the weak duality property is given by

$$||F|| \leq g \Rightarrow F^\top(Au) \leq g^\top ||Au||. \tag{9}$$

The next property shows that the norm $||F|| = |A^\top|F^2$ verifies weak duality.

Property 6. *Let G be a transport graph, F a flow in G verifying the capacity constraint $\sqrt{|A^\top|F^2} \leq g$. Let u be a vector of $\mathbb{R}^n$ (defined on nodes of G). Then*

$$F^\top Au \leq g^\top \sqrt{|A^\top|(Au)^2}.$$

Proof. Since we know that $\sqrt{|A^\top|F^2} \leq g$, the following statement is true

$$\sqrt{|A^\top|F^2}^\top \sqrt{|A^\top|(Au)^2} \leq g^\top \sqrt{|A^\top|(Au^2}.$$

We can now show that

$$F^\top Au \leq \sqrt{(|A^\top|F^2)}^\top \sqrt{|A^\top|(Au)^2}.$$

Using summation notation, then by the Cauchy-Schwartz inequality,

$$\sum_{i\in V}\sum_{e_{ij}\in E} F_{ij}(u_i-u_j) \leq \sum_{i\in V}\sqrt{\left(\sum_{e_{ij}\in E} {F_{ij}}^2\right)\left(\sum_{e_{ij}\in E}(u_i-u_j)^2\right)}.$$

We conclude that

$$F^\top Au \leq \sqrt{(|A^\top|F^2)}^\top \sqrt{|A^\top|(Au)^2} \leq g^\top \sqrt{|A^\top|(Au)^2}.$$

□

In terms of energy value, this proposition means that the CCMF energy, that is to say the flow, is always smaller than the combinatorial total variation.

Property 7. *Let F be a compatible flow verifying the constraints in (2.3), and u a vector of $\mathbb{R}^n$ such as $u_s - u_t = 1$. Then,*

$$F_{st} \leq g^\top \sqrt{A^\top (Au)^2}. \tag{10}$$

Proof. In the combinatorial setting, the Green formula gives

$$F^\top Au = u^\top A^\top F.$$

Let a be a vector of $\mathbb{R}^n$ defined for each node v_i as

$$a_i = \begin{cases} -1 & \text{if } v_i \text{ belongs to the sink,} \\ 1 & \text{if } v_i \text{ belongs to the source,} \\ 0 & \text{otherwise.} \end{cases} \tag{11}$$

We can provide the following equivalent formulation of the CCMF problem (2.3) using an

incidence matrix A of the transport graph deprived of the sink-source edge:

$$\begin{aligned} &\max F_{st}, \\ &\text{s.t.} \quad A^\top F = F_{st} a, \\ &|A^\top| F^2 \leq g. \end{aligned} \tag{12}$$

As F verifies the divergence free constraint $A^\top F = F_{st} a$,

$$u^\top A^\top F = u^\top F_{st} a,$$

and as the equation $u_s - u_t = 1$ may be written equivalently $a^\top u = 1$, we conclude that

$$u^\top F_{st} a = F_{st},$$

and by weak duality (Property 6),

$$F_{st} \leq g^\top \sqrt{A^\top (Au)^2}.$$

□

A.1.3 Algorithms for solving the CCMF problem

A.1.3.1 Parallel proximal method for solving the primal problem

It is possible to reformulate the problem (2.6) into a single objective function optimization problem. We note D the divergence convex set

$$D = \{F \in \mathbb{R}^m \mid A^\top F = 0\}, \tag{13}$$

and C the capacity convex set

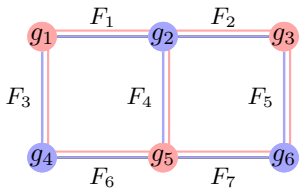
$$C = \{F \in \mathbb{R}^m \mid |A^\top| F^2 \leq g^2\}. \tag{14}$$

The problem (2.6) may also be written as

$$\min_F -c^\top F + i_C(F) + i_D(F). \tag{15}$$

In order to employ the parallel proximal algorithm of Section 1.3.3.1, we need to compute the proximal operator of $\imath_C$ and $\imath_D$, corresponding to P_C and P_D by definition. Projecting directly on C and D is analytically intractable so we split the two convex sets into several parts. We decompose $\imath_C(F)$ into the sum of several indicator function on convex sets C_k, $k = 1...p$. Given a decomposition of the set of nodes $V = \cap_{k=1}^p S_k$, we define

$$C_k = \{F \in \mathbb{R}^m |\ \ \forall i \in S_k \ \ |A^\top|_i F^2 \leq g_i^2, \} \tag{16}$$



$$C_1 = \{F \in \mathbb{R}^m |$$
$$(F_1^2 + F_2^2 + F_4^2)^{\frac{1}{2}} \leq g_2^2$$
$$(F_3^2 + F_6^2)^{\frac{1}{2}} \leq g_4^2$$
$$(F_5^2 + F_7^2)^{\frac{1}{2}} \leq g_6^2\}$$

$$C_2 = \{F \in \mathbb{R}^m |$$
$$(F_1^2 + F_3^2)^{\frac{1}{2}} \leq g_1^2$$
$$(F_2^2 + F_5^2)^{\frac{1}{2}} \leq g_3^2$$
$$(F_4^2 + F_6^2 + F_7^2)^{\frac{1}{2}} \leq g_5^2\}$$

Figure 1: Decomposition of $C = C_1 \cap C_2$

For a 4-connected lattice, this decomposition may be achieved with $p = 2$, by defining a partition S_k composed of two disjoint sets having a checkered pattern. An example of such a decomposition for small graph is given at Fig. 1.

This decomposition of C allows to perform efficient projections on the C_k in parallel, that reduce to projections on hyperspheres. Following the same idea, we decompose $\imath_D(F)$ into the sum of several indicator function on convex sets D_k, $k = 1...q$. In a 4-connected lattice, the decomposition may be achieved with $q = 3$

$$D_{k=1,2} = \{F|\ \ \forall i \in S_k \ \ A_i^\top F = 0\}, \quad D_3 = \{F|\ \ A_s^\top F = 0 \text{ and } A_t^\top F = 0\}, \tag{17}$$

where s denotes the source node, and t the sink node. Similarly to the case of C, the projections on convex sets D_i are given by simple projections onto hyperplanes.

We obtain the following problem

$$\min_F \sum_{k=1}^{p} i_{C_k}(F) + \sum_{k=1}^{q} i_{D_k}(F) - c^\top F, \tag{18}$$

optimized by the parallel proximal algorithm given in Algo 9.

Algorithm 9: Parallel proximal algorithm optimizing CCMF (15)

Data: A transport graph, a metric $g \in \mathbb{R}^n$
Result: A maximum flow $F \in \mathbb{R}^m$
For $i = 1, ..., p+q$, $y_i \in \mathbb{R}^m = 0$
Set $F = 0$, choose $\nu \in]0, 2[$

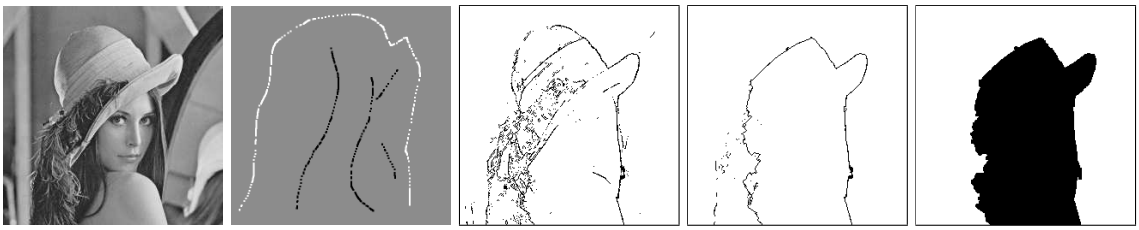
Repeat until convergence
- For (in parallel) $i = 1, \ldots, p+q+1$
  - $p_i = \begin{cases} P_{C_i}(y_i) & \text{if } i \leq p \\ P_{D_{i-p}}(y_i) & \text{if } p < i \leq p+q \\ y_i + c & \text{otherwise} \end{cases}$
- $p_k = \frac{1}{p+q+1} \sum_{i=1}^{p+q+1} p_i$
- For (in parallel) $i = 1, \ldots, p+q+1$
  - $y_i = y_i + \nu(2p - F - p_i)$
- $F = F + \nu(p - F)$

From the optimal flow vector F, one can obtain a map of the saturated nodes (See Def. 2.2.1), as displayed in Figure 2, but the convergence is very slow, and subject to large oscillations, making it difficult to find a stopping criterium. Furthermore, the value of F is not sufficient to obtain the optimal value of ν and λ that can not be recovered analytically. With some patience, after about one hour (on CPU) for the image of Fig. 2, a segmentation can be obtained, by propagation of the seeds into the saturation map.

The most important issue is that this algorithm only gives a value of the flow, and the final deduced segmentation is not very robust when the optimal value of flow is not precise enough.

A.1.3.2 First order method for solving the dual problem

In order to obtain directly the optimal labeling, we perform a direct optimization of the CCMF dual problem. Our goal is to optimize



(a) Image (b) Seeds (c) Saturated nodes 130 s. (d) Saturated nodes 1 h. (e) Segmentation

Figure 2: Segmentation of a 256 by 256 image using PPXA.
(c) Saturated vertices after 130 seconds (4000 iterations) (d) Saturated vertices after about one hour (120000 iterations). In this case, the obtained map is precise enough to obtain, by propagation of the marker, the segmentation in (e).

$$\min_{\lambda\in\mathbb{R}^n,\ \nu\in\mathbb{R}^n} E(\lambda,\nu) = \quad \lambda^\top g^2 + \tfrac{1}{4}\Big(\mathbf{1}^n \cdot /(|A|\lambda)\Big)^\top \Big((c+A\nu)^2\Big), \qquad (19)$$
$$\text{s. t.} \qquad \lambda \geq 0.$$

The derivatives with respect to λ and ν are given by

$$\nabla_\lambda(E) = -\frac{1}{4}|A|^\top \Big((c+A\nu)\cdot /(|A|\lambda)\Big)^2 + g^2,$$

$$\nabla_\nu(E) = \frac{1}{2}A^\top \Big((c+A\nu)\cdot /(|A|\lambda)\Big)$$

A conjugate gradient descent has been tested for the optimization of (20). The constraint of positivity for λ was enforced by setting to neglectable but positive values the negative values of λ at each step. A second strategy was to employ a method alternating between minimizations in function of λ and minimization in function of ν. Both approaches were converging toward the optimal solution until a certain point but the precision required was rarely reached.

Further possible tests would include implementations of Beck and Teboule's FISTA algorithm, and projected gradient method (i.e [19]).

A.2 Power watershed

A.2.1 The case p finite, $q \to \infty$: Voronoi diagram

Intuitively, we see that when the power over the neighboring differences tends toward infinity, the weights become negligible so that the problem obtained from (4.4) is a Voronoi diagram of the seeds.

A proof showing that solving the minimization problem (4.4) when $p = q$ and $q \to \infty$ can be achieved by shortest path computations is given in [200]. Here we use the same idea to prove that the problem (4.4) in the case p finite, $q \to \infty$ is equivalent to a Voronoi diagram problem.

As $\sqrt[q]{\cdot}$ is monotonic, minimizing $E_{p,q}$ is equivalent to minimizing $\sqrt[q]{E_{p,q}}$.

First we may factorize the objective function of our problem (4.4)

$$\sqrt[q]{\sum_{e_{ij}\in E} w_{ij}^p |x_i - x_j|^q} = \sqrt[q]{\sum_{e_{ij}\in E} \left(w_{ij}^{\frac{p}{q}} |x_i - x_j| \right)^q} \tag{20}$$

Taking the limit $\lim_{q\to\infty} \sqrt[q]{\sum_i X_i^q}$ of a q-norm yields the maximum norm $\max_i X_i$.

Therefore, our objective function may be written

$$\lim_{q\to\infty} \sqrt[q]{\sum_{e_{ij}\in E} w_{ij}^p |x_i - x_j|^q} = \lim_{q\to\infty} \max_{e_{ij}\in E} w_{ij}^{\frac{p}{q}} |x_i - x_j|. \tag{21}$$

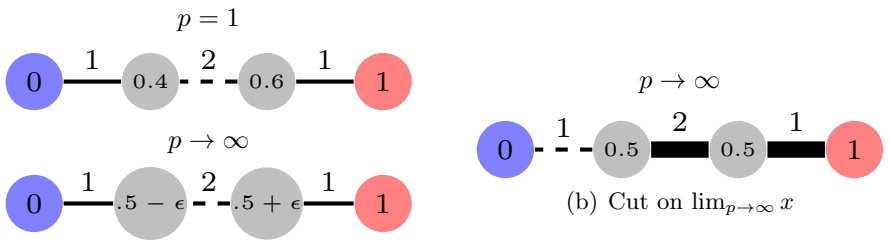
The minimization problem can be written as

$$\begin{aligned} &\min_x \max_{e_{ij}\in E} \lim_{q\to\infty} w_{ij}^{\frac{p}{q}} |x_i - x_j|, \\ &\text{s.t. } \; x(F) = 1, \;\; x(B) = 0. \end{aligned} \tag{22}$$

When $q \to \infty$ and p is finite, $\frac{p}{q}$ converges toward 0, so $w_{ij}^{\frac{p}{q}}$ converges toward 1 for every edge of E. Also the case p finite, $q \to \infty$ can be brought back to the case $p = 0$, $q \to \infty$ which solution is a Voronoi diagram with an ℓ_1 norm (due to the assumed 4-connectivity of the lattice).

A.2.2 Illustration of Theorem 4.3.1

In theorem 4.3.1, the condition for seeds to be the maxima of the weight function is necessary as illustrated in the following Figure 3.



(a) q-cuts for different p

(b) Cut on $\lim_{p\to\infty} x$

Figure 3: Example of seeded graph where the q-cut does not correspond to an MSF cut. Let $x = \arg\min E_{pq}$. When the maxima of the weight function are not seeded, the threshold of $\lim_{p\to\infty} x$ can be different of the limit $\lim_{p\to\infty}$ of the threshold of x. (a) Labeling $x = \arg\min E_{pq}$ and corresponding q-cut ($q = 2$) when the weights are at the power $p = 1$, and below for an arbitrary big value of p. The q-cut in dashed line remains in the center. (b) Labeling $x = \arg\min \lim_{p\to\infty} E_{pq}$ and cut (dashed) corresponding to the threshold of x. In this example, the q-cut is not an MSF cut, justifying the condition in Thm. 4.3.1. Note that the threshold of $\lim_{p\to\infty} x$ is an MSF cut, as stated later in Prop. 5.

A.2.3 Proof of Property 4

If q is a real number such that $1 < q < \infty$, then the solution x to problem (4.4) is unique.

Proof. Let A be the incidence matrix of the graph G, and x a vector of $\mathbb{R}^n_+$. We note by $|\cdot|$ the elementwise absolute value operator. The function $g : x \to Ax$ is convex. The function $h : x \to |x|^q$ is convex and non-decreasing. The function $f : x \to w^\top x$ is also convex and non-decreasing. Note that $E_{p,q}(x)$ can be written the following way:

$$E_{p,q}(x) = f \circ h \circ g(x) = w^{p\top} |Ax|^q. \tag{23}$$

As h is a non-decreasing convex function, and g is convex, $h \circ g$ is convex. As f is a non-decreasing convex function, $E_{p,q}$ is convex.

If $1 < q < \infty$, the function $h \circ g$ is strictly convex, so $E_{p,q}$ is a strictly convex function, thus the minimization of $E_{p,q}$ subject to the boundary constraints is achieved by a unique x. □

A.2.4 Proof of Theorem 4.4.1

The energy of the solution computed by the power watershed algorithm converges to the energy which minimizes $E_{p,q}$ when $p \to \infty$.

Let p, q be real positive numbers. Let w_M be the maximum weight of the graph G. For every $\delta > 0$, there exists a real k such that if $p > k$,

$$0 \leq \frac{E_{p,q}(\bar{x})}{{w_M}^p} - \frac{E_{p,q}(x^*)}{{w_M}^p} \leq \delta. \tag{24}$$

Proof.

$$\frac{E_{p,q}\,(x)}{{w_M}^p} = \sum_{e_M} |x_i - x_j|^q + \sum_{e_{ij} \neq e_M} \left(\frac{w_{ij}}{w_M}\right)^p |x_i - x_j|^q.$$

$$\begin{aligned} \frac{E_{p,q}(\bar{x})}{{w_M}^p} - \frac{E_{p,q}(x^*)}{{w_M}^p} = \sum_{e_M} |\bar{x_i} - \bar{x_j}|^q - \sum_{e_M} |x_i^* - x_j^*|^q + \\ \sum_{e_{ij} \neq e_M} \left(\frac{w_{ij}}{w_M}\right)^p |\bar{x_i} - \bar{x_j}|^q - \sum_{e_{ij} \neq e_M} \left(\frac{w_{ij}}{w_M}\right)^p |x_i^* - x_j^*|^q. \end{aligned} \tag{25}$$

The first part of (25) is bounded by 0,

$$\sum_{e_M} |\bar{x_i} - \bar{x_j}|^q - \sum_{e_M} |x_i^* - x_j^*|^q \leq 0, \tag{26}$$

because the energy obtained with $\bar{x}$ can not be greater than the one obtained by the optimal solution x^*. More precisely, if there are no plateaus with different labels, the x_i^*, x_j^* computed on the edges e_M with Algorithm 7 are equal, leading to a sum equal to 0. Else (if there are plateaus with different labels), $\sum_{e_M} |x_i - x_j|^q$ subject to the boundary constraints is minimized on the plateaus, so the solution is optimal.

The last part of (25) is also negative,

$$-\sum_{e_{ij} \neq e_M} \left(\frac{w_{ij}}{w_M}\right)^p |x_i^* - x_j^*|^q \leq 0. \tag{27}$$

It only remains to bound the middle part of (25)

$$\sum_{e_{ij}\neq e_M}\left(\frac{w_{ij}}{w_M}\right)^p|\bar{x_i}-\bar{x_j}|^q\leq\sum_{e_{ij}\neq e_M}\left(\frac{w_{ij}}{w_M}\right)^p\leq M_2\left(\frac{w_{M_2}}{w_M}\right)^p, \tag{28}$$

with M_2 the number of edges of weight inferior to w_M, and w_{M_2} the second maximum weight.

Thus we have

$$\frac{E_{p,q}(\bar{x})}{{w_M}^p}-\frac{E_{p,q}(x^*)}{{w_M}^p}\leq M_2\left(\frac{w_{M_2}}{w_M}\right)^p. \tag{29}$$

$$p\geq k=\frac{\log\frac{\delta}{M_2}}{\log\frac{w_{M_2}}{w_M}}. \tag{30}$$

□

A.2.5 Interpretation as a Markov Random Field

An optimum $x^\star$ of Eq. 4.5 may be interpreted as a probability for each pixel to belong to the object (as opposed to background). More rigorously, as shown in [198], this segmentation model can be viewed as an estimation of a continuous valued MRF. By linking the watershed algorithm to this framework, it becomes possible to view the watershed algorithm as the MAP estimation of an MRF. However, note that this analysis allows us to interpret the watershed as the MAP estimate for a particular MRF, which is in contrast to previous efforts to link a probabilistic framework with the watershed (such as [9], who use random placements of seeds to define the most probable locations of the watershed lines).

In this section, we follow the development of [198], with modifications to incorporate the power watershed.

In the interpretation as an MRF, we define the binary segmentation label s_i for node v_i as a Bernoulli random variable (i.e., $s_i=1$ if v_i is foreground and $s_i=0$ if v_i is background), in which the variable x_i denotes the success probability for the distribution of s_i, i.e., $p(s_i=1|x_i)$. In this case, the success probability may be written as

$$p(s_i=1|x_i)=\max\{\min\{x_i,1\},0\}=\begin{cases}1 \text{ if } x_i>1,\\ x_i \text{ if } 0\leq x_i\leq 1,\\ 0 \text{ if } x_i<0.\end{cases} \tag{31}$$

However, the generalized mean value theorem in [198] guarantees that the optimal solution

$x^\star$ to (4.5), assuming that the auxiliary nodes have values comprised between 0 and 1, takes its values between 0 and 1 when the weights are all positive valued. Consequently, in our context, we may simply set $p(s_i = 1|x_i) = x_i$ without concern that x_i will be outside the interval $[0, 1]$.

Our goal is now to infer the hidden variables x_i from the image content I. The hidden variables may be estimated in a Bayesian framework by considering the posterior model

$$p(x, s|I) \quad \propto \quad p(x)p(s|x)p(I|s) \quad = \quad p(x) \prod_{v_i \in V} p(s_i|x_i) \prod_{v_i \in V} p(I_i|s_i), \quad (32)$$

in which $p(x)$ models how the parameters of the Bernoulli variables vary spatially. The spatial smoothness prior is parameterized by

$$p(x) \propto \exp\left(-\lambda \sum_{e_{ij} \in E} w_{ij}^p |x_i - x_j|^q\right), \quad (33)$$

where $\lambda > 0$ and the weights are strictly positive.

We can estimate the marginalized MAP criterion to obtain the optimum x^* by setting

$$x^* = \arg\max_x p(x)p(I|x) = \arg\max p(x) \sum_s p(I|s)p(s|x). \quad (34)$$

Unfortunately, $\sum_s p(I|s)p(s|x)$, is not straightforward to estimate. Therefore, we assume that we can parameterize $p(I_i|x_i)$ as

$$p(I|x) \propto \exp\left(-\sum_{v_i \in V} w_{i0}^p |x_i - 0|^q - \sum_{v_i \in V} w_{i1}^p |x_i - 1|^q\right), \quad (35)$$

where $w_{i0} \geq 0$ and $w_{i1} \geq 0$, and these terms act to bias the parameters x_i toward 0 and 1. Similarly, these terms can be used to encode the user interaction (seeding) by setting a foreground seed v_i to have weights $(w_{i0}, w_{i1}) = (0, \infty)$ and a background seed to have weights $(w_{i0}, w_{i1}) = (\infty, 0)$. With this parameterization, then the MAP estimate described in (34) is equal to our energy minimization problem from (4.5).

While the use of binary variables (the s variable in our formulation) is more common in recent work which applies MRFs to image segmentation, our focus on estimating a real-valued parameter or variable is far from unique in the computer vision literature. For example, in Gaussian MRFs the variables each have a Gaussian distribution and the goal is often to

estimate the (real-valued) parameters of these variables (i.e., mean and/or variance). These kind of MRFs have been applied in image segmentation and other pattern recognition applications [8, 134, 206]. Beyond Gaussian MRFs, anisotropic diffusion has been interpreted as a continuous-valued MRF [142, 159] and MRFs requiring continuous-valued estimations have appeared in both early work on computer vision [30, 98, 100, 149] and also recently [150, 191, 193].

A.2.6 Using mathematical morphology for an efficient preprocessing step

One difficulty in Algorithm 7 is dealing with the set of merged nodes. More precisely, when solving (4.7) we need to keep track of which nodes have merged (with some nodes merged multiple times). If we look informally at the "emergence" process underlying the algorithm, it will help us to locate those maximal merged nodes. Using topographical references, we view the weights as the surface of a terrain, with the weight of an edge corresponding to its altitude. If the surface were completely covered by water, and the level of water slowly decreases, then islands (regional maxima) would appear that grow and merge. At a given level, when an island that does not contain a seeded pixel meets an island containing one, we can give a value to the (maximal) merged node. Indeed, we can see that any merged node consists of a connected component of a upper-level set of the weights. More precisely, let $\lambda \in \mathbb{R}^+$ and w be the weight function defined on E. We define

$$w[\lambda] = \{e \in E | w(e) \geq \lambda\}. \tag{36}$$

The graph induced by $w[\lambda]$ is called a *section* of w. A connected component of a section $w[\lambda]$ is called a *component of w (at level λ)*.

The components of w can be used to find merged nodes.

Property 8. *Any maximal merged node corresponds to a component of w that:*

- *does not contain any seed;*
- *and is not contained in a larger unseeded component of w.*

Conversely, any component of w satisfying these two properties corresponds to a maximal merged node in Algorithm 7.

The components of w, ordered by the inclusion relation, form a tree called the *max-tree* [190] or the *component tree* [42, 122, 123]. Several efficient algorithms exist to compute the

component tree, some quasi-linear [165] (based on union-find [207]) and some parallelized [155, 217]. From Prop. 8, it is easy to see how to use this tree in Algorithm 7. Note that such a tree, which keeps track of all components, can be used when one wants to improve a given segmentation result by adding extra seeds.

Another tool from mathematical morphology has been used as a preprocessing step for watershed segmentation with markers. It is called *geodesic reconstruction from the markers* [21, 157], and is given as a function w_R such that, for every edge e, we set $w_R(e)$ to be equal to the level δ of the highest component of w containing e and at least one seed node. Note that any component of w_R contains at least one seed.

Property 9. *Any maximal merged node corresponds to a connected set of edges e_{ij} that belong to a plateau of w_R and that satisfy $w_{ij} > w_R(e_{ij})$. The converse is also true.*

Prop. 9 also suggests that geodesic reconstruction can be used as a preprocessing in Alg. 7. Note that there exist some very efficient and easy to implement algorithms to compute a geodesic reconstruction [101, 167, 215]. Both the component tree and the geodesic reconstruction have the same theoretical complexity, so either approach could be used profitably to reduce the bookkeeping necessary to keep track of merged nodes.

Prop. 9 also suggests links between our framework and the classical watershed-based segmentation framework [21, 157, 158]. The framework of watershed cuts [77, 78] allows us to make a precise statement about this connection. The cut provided by a maximum spanning forest with one different seed for every maxima is called a *watershed cut*. Since geodesic reconstruction removes all maxima which are not connected to a seed, then we can state the following:

Property 10. *Any q-cut is a watershed cut of the reconstructed weights.*

This statement ties the cuts produced by our power watershed framework to the concept of watershed cuts in the morphology literature.

List of Figures

Index

References

[1] http://sourceforge.net/projects/powerwatershed/. 79

[2] Stanford 3D scanning repository. http://graphics.stanford.edu/data/3Dscanrep/. 109

[3] M. V. Afonso, J. M. Bioucas Dias, and M. A. Figueiredo. An augmented Lagrangian approach to the constrained optimization formulation of imaging inverse problems. *Inverse Problems*, 20(3):681–695, March 2011. 48, 53

[4] C. Allène, J.-Y. Audibert, M. Couprie, J. Cousty, and R. Keriven. Some links between min cuts, optimal spanning forests and watersheds. In *7th international symposium on mathematical morphology ISMM'07*, volume 2, pages 253–264. INPE, 2007. 69, 75, 89

[5] C. Allène, J.-Y. Audibert, M. Couprie, and R. Keriven. Some links between extremum spanning forests, watersheds and min-cuts. *Image and Vision Computing*, 28:1460–1471, 2009. 69, 75, 77, 88, 89

[6] C. V. Alvino, G. B. Unal, G. Slabaugh, B. Peny, and T. Fang. Efficient segmentation based on Eikonal and diffusion equations. *Int. J. Comput. Methods*, 84(9):1309–1324, 2007. 11, 72

[7] N. Amenta, S. Choi, and R. K. Kolluri. The power crust. In *Proceedings of the sixth ACM symposium on Solid modeling and applications*, SMA '01, pages 249–266, New York, NY, USA, 2001. ACM. 103, 111

[8] A. Anandkumar, L. Tong, and A. Swami. Detection of Gauss-Markov random field on nearest-neighbor graph. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'07*, volume 3, 2007. 130

[9] J. Angulo and D. Jeulin. Stochastic watershed segmentation. In *Proceedings of the 8th International Symposium on Mathematical Morphology, ISMM'07*, pages 265–276, 2007. 128

[10] B. Appleton and H. Talbot. Globally minimal surfaces by continuous maximal flows. *IEEE Trans. on PAMI*, 28(1):106–118, 2006. 19, 20, 21, 24, 26, 35, 36, 37, 71

[11] P. A. Arbeláez and L. D. Cohen. A metric approach to vector-valued image segmentation. *Int. J. Comput. Vis.*, 69(1):119–126, 2006. 115

[12] R. Audigier and R. Lotufo. Uniquely-determined thinning of the tie-zone watershed based on label frequency. *J. Math. Imaging Vision*, 27(2):157–173, 2007. 82

[13] J.-F. Aujol. Some first-order algorithms for total variation based image restoration. *J. Math. Imaging Vision*, 34:307–327, 2009. 47

[14] D. Auroux. Fast algorithms for image processing and data assimilation. Habilitation à Diriger des Recherches (Habilitation Thesis), University of Toulouse, France, November 2008. 8

[15] L. Bagnato, P. Frossard, and P. Vandergheynst. Optical flow and depth from motion for omnidirectional images using a TV-ℓ_1 variational framework on graphs. In *Proceedings of the 16th IEEE International Conference on Image Processing*, ICIP'09, pages 1453–1456, Cairo, Egypt, 2009. 47

[16] X. Bai and G. Sapiro. A geodesic framework for fast interactive image and video segmentation and matting. In *Proceedings of International Conference of Computer Vision (ICCV)*, 2007. 11, 72

[17] F. Benmansour and L. D. Cohen. Tubular structure segmentation based on minimal path method and anisotropic enhancement. *Int. J. Comput. Vision*, 92:192–210, April 2011. 10

[18] G. Bertrand. On topological watersheds. *J. Math. Imaging Vision*, 22(2-3):217–230, 2005. 70

[19] D. P. Bertsekas. Projected newton methods for optimization problems with simple constraints. *SIAM J. Control and Optimization*, 1982. 124

[20] S. Beucher and C. Gratin. *Micromorph reference manual, applications and solutions.* Ecole des Mines de Paris, 1989. 103

[21] S. Beucher and F. Meyer. The morphological approach to segmentation: The watershed transformation. In E. R. Dougherty, editor, *Mathematical Morphology in Image Processing*, pages 433–481. CRC, 1993. 69, 81, 131

[22] A. Bhusnurmath and C. J. Taylor. Graph cuts via ℓ_1 norm minimization. *IEEE Trans. on PAMI*, 30(10):1866–1871, October 2008. 12, 22, 31

[23] A. Bieniek and A. Moga. An efficient watershed algorithm based on connected components. *Pattern Recognition*, 33(6):907 – 916, 2000. 81

[24] M. J. Black, G. Sapiro, D. H. Marimont, and D. Heeger. Robust anisotropic diffusion. *IEEE Trans. Image Process.*, 7(3):421–432, March 1998. 8, 92, 93

[25] A. Blake, C. Rother, M. Brown, P. Perez, and P. Torr. Interactive image segmentation using an adaptive GMMRF model. In *Proceedings of European Conference on Computer Vision (ECCV)*, LNCS 3021, pages 428–441, 2004. 71

[26] A. Blake and A. Zisserman. *Visual Reconstruction*. MIT Press, 1987. 71

[27] J. Bolz, I. Farmer, E. Grinspun, and P. Schröder. Sparse matrix solvers on the GPU: Conjugate gradients and multigrid. In *ACM Transactions on Graphics*, volume 22 of *SIGGRAPH*, pages 917–924, July 2003. 30, 38

[28] O. Borůvka. O jistém problému minimálním, 1926. 11

[29] S. Bougleux, A. Elmoataz, and M. Melkemi. Discrete regularization on weighted graphs for image and mesh filtering. In *Proceedings of International Conference on Scale Space and Variational Methods in Computer Vision*, SSVM '07, pages 128–139, Ischia, Italy, May 30 - June 2 2007. 7, 48, 49

[30] C. Bouman and K. Sauer. A generalized Gaussian image model for edge-preserving MAP estimation. *IEEE Trans. Image Process.*, 2(3):296–310, 1993. 130

[31] S. Boyd and L. Vandenberghe. *Convex Optimization*. CUP, 2004. 12, 14, 29

[32] Y. Boykov and M.-P. Jolly. *Interactive graph cuts* for optimal boundary & region segmentation of objects in N-D images. In *Proceedings of International Conference of Computer Vision (ICCV)*, pages 105–112, 2001. 7, 41, 71

[33] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. on PAMI*, 26:359–374, 2001. 8, 87

[34] Y. Boykov and V. Kolmogorov. Computing geodesics and minimal surfaces via graph cuts. In *Proceedings of International Conference of Computer Vision (ICCV)*, volume 1, 2003. 71

[35] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. on PAMI*, 26(9):1124–1137, 2004. 10

[36] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. on PAMI*, 23(11):1222–1239, Nov. 2001. 20, 103

[37] J. P. Boyle and R. L. Dykstra. A method for finding projections onto the intersection of convex sets in hilbert spaces. *Advances in order restricted statistical inference (Iowa City, Iowa, 1985), Lecture Notes in Statist.*, 37:2847, 1986. 52

[38] I. Braude, J. Marker, K. Museth, J. Nissanov, and D. Breen. Contour-based surface reconstruction using mpu implicit models. *Graphical models*, 69, 2007. 102, 103, 111

[39] K. Bredies, K. Kunisch, and T. Pock. Total generalized variation. *SIAM J. Imaging Sci.*, 3(3):492–526, 2010. 48

[40] K. Bredies, K. Kunisch, and T. Valkonen. Properties of l^1 - TGV^2 : The one-dimensional case, 2011. Submitted for publication. 48

[41] E. Breen and R. Jones. Attribute openings, thinnings and granulometries. *Graphical Models and Image Processing Journal*, 64(3):377–389, 1996. 104

[42] E. J. Breen and R. Jones. Attribute openings, thinnings, and granulometries. *Comput. Vis. Image Underst.*, 64(3):377–389, 1996. 130

[43] D. Brélaz. New methods to color the vertices of a graph. *Communications of the ACM*, 22:251–256, April 1979. 54, 55, 134

[44] X. Bresson. A short note for nonlocal TV minimization, 2009. http://www.cs.cityu.edu.hk/ xbresson/ucla/papers/a%20short%20note%20for%20nonlocal%20TV%20minimization.pdf. 7, 59, 61

[45] X. Bresson, S. Esedoglu, P. Vandergheynst, J.-P. Thiran, and S. Osher. Fast global minimization of the active contour/snake model. *J. Math. Imaging Vision*, 28(2):151–167, 2007. 92

[46] L. Briceño Arias and P. L. Combettes. A monotone+skew splitting model for composite monotone inclusions in duality. arXiv:1011.5517, 29 Nov 2010. 15, 16, 17, 55, 57

[47] A. Buades, B. Coll, and J.-M. Morel. A non-local algorithm for image denoising. In *Proceedings of IEEE Computer Vision and Pattern Recognition*, volume 2 of *CVPR'05*, pages 60–65, San Diego, CA, USA, 2005. 6, 23, 48, 59

[48] J. Carter. *Dual Methods for Total Variation-based Image Restoration*. PhD thesis, UCLA, Los Angeles, CA, USA, 2001. 47

[49] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contours. *Int. J. Comput. Vis.*, 22(1):61–79, 1997. 24, 26

[50] A. Chambolle. An algorithm for total variation minimization and applications. *J. Math. Imaging Vision*, 20(1-2):89–97, 2004. 7, 20, 32, 47, 48, 57, 58, 92

[51] A. Chambolle, V. Caselles, M. Novaga, D. Cremers, and T. Pock. An introduction to Total Variation for Image Analysis, 2010. 7

[52] A. Chambolle, D. Cremers, and T. Pock. A convex approach for computing minimal partitions. Tech. Rep. 649, Ecole Polytechnique CMAP, 2008. 21

[53] A. Chambolle, S. E. Levine, Bradley, and J. Lucier. Upwind and multiscale finite-difference methods for total variationbased image smoothing. *SIAM J. Imaging Sci.*, 4:277–299, 2011. 7

[54] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *J. Math. Imaging Vision*, 40(1):120–145, 2011. 7, 21, 26, 34, 37, 48, 55

[55] T. F. Chan, G. H. Golub, and P. Mulet. A nonlinear primal-dual method for total variation based image restoration. *SIAM J. Sci. Comput.*, 20(6):1964–1977, 1999. 12, 47, 49

[56] C. Chaux, A. Jezierska, J.-C. Pesquet, and H. Talbot. A spatial regularization approach for vector quantization. *Accepted for publication in JMIV*, 2010. 99

[57] B. Chazelle. A minimum spanning tree algorithm with inverse-ackermann type complexity. *J. ACM*, 47:1028–1047, November 2000. 11, 75, 105

[58] G. Chen and M. Teboulle. A proximal-based decomposition method for convex minimization problems. *Math. Program.*, 64:81–101, 1994. 55

[59] I. Ciril and J. Darbon. Image denoising with a constrained discrete total variation scale space. In *Proceedings of Discrete Geometry for Computer Imagery*, pages 465–476, Nancy, France, 2011. Springer. 48

[60] L. D. Cohen. Minimal paths and fast marching methods for image analysis. *Mathematical Models in Computer Vision: The Handbook, Nikos Paragios and Yunmei Chen and Olivier Faugeras Editors*, 2005. 10

[61] L. D. Cohen and R. Kimmel. Global minimum for active contour models: A minimal path approach. *Int. J. Comput. Vis.*, 24(1):57–78, 1997. 71

[62] R. R. Coifman, S. Lafon, A. B. Lee, M. Maggioni, B. Nadler, F. Warner, and S. W. Zucker. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proceedings of the National Academy of Sciences*, 102(21):7426–7431, 2005. 71

[63] P. L. Combettes, D. Dũng, and B. C. Vũ. Dualization of signal recovery problems. arXiv:0907.0436v4, 2009. 52

[64] P. L. Combettes and J.-C. Pesquet. Image restoration subject to a total variation constraint. *IEEE Trans. Image Process.*, 13(9):1213–1222, September 2004. 49

[65] P. L. Combettes and J.-C. Pesquet. Proximal thresholding algorithm for minimization over orthonormal bases. *SIAM J. Optim.*, 18:1351–1376, 2007. 57

[66] P. L. Combettes and J.-C. Pesquet. A proximal decomposition method for solving convex variational inverse problems. *Inverse problems*, 24(6), 2008. 15, 47, 48, 53, 62, 63, 64

[67] P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In H. H. Bauschke, R. Burachik, P. L. Combettes, V. Elser, D. R. Luke, and H. Wolkowicz, editors, *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pages 185–212. Springer-Verlag, New York, 2010. 16, 48, 52

[68] P. Coupé, P. Hellier, S. Prima, C. Kervrann, and C. Barillot. 3D wavelet subbands mixing for image denoising. *Journal of Biomedical Imaging*, 2008:1–11, January 2008. 61

[69] C. Couprie, X. Bresson, L. Najman, H. Talbot, and L. Grady. Surface reconstruction using power watersheds. In *Proc. of International Symposium on Mathematical Morphology, ISMM'11*, pages 381–392, 2011. 91

[70] C. Couprie, L. Grady, L. Najman, and H. Talbot. Power watersheds: a new image segmentation framework extending graph cuts, random walker and optimal spanning forest. In *Proceedings of International Conference of Computer Vision (ICCV)*, pages 731–738, 2009. 69, 104

[71] C. Couprie, L. Grady, L. Najman, and H. Talbot. Anisotropic diffusion using power watersheds. In *Proceedings of International Conference on Image Processing(ICIP)*, 2010. 91

[72] C. Couprie, L. Grady, L. Najman, and H. Talbot. Power Watersheds: A Unifying Graph Based Optimization Framework. *IEEE Trans. on PAMI*, 2011. 69, 104

[73] C. Couprie, L. Grady, H. Talbot, and L. Najman. Combinatorial Continuous Maximum Flow. *SIAM J. Imaging Sci.*, 2011. 19, 51

[74] C. Couprie, H. Talbot, J.-C. Pesquet, L. Najman, and L. J. Grady. Dual constrained TV-based regularization. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'11*, pages 945–948, Prague, Czech Republic, May 2011. 47

[75] M. Couprie, L. Najman, and G. Bertrand. Quasi-linear algorithms for the topological watershed. *J. Math. Imaging Vision*, 22(2-3):231–249, 2005. 81

[76] J. Cousty, G. Bertrand, L. Najman, and M. Couprie. Watershed cuts. In *7th international symposium on mathematical morphology (ISMM '07)*, volume 1, pages 301–312. INPE, 2007. 70

[77] J. Cousty, G. Bertrand, L. Najman, and M. Couprie. Watershed Cuts: Minimum Spanning Forests and the Drop of Water Principle. *IEEE Trans. on PAMI*, 31(8):1362–1374, Aug. 2009. 12, 70, 76, 104, 131

[78] J. Cousty, G. Bertrand, L. Najman, and M. Couprie. Watershed cuts: thinnings, shortest-path forests and topological watersheds. *IEEE Trans. on PAMI*, 32(5):925–939, 2010. 70, 131

[79] A. Criminisi, T. Sharp, and A. Blake. GeoS: Geodesic image segmentation. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 99–112, 2008. 11, 72

[80] J. H. D. Shulman. Regularization of discontinuous flow fields. *Proceedings Workshop on Visual Motion*, pages 81–86, 1989. 6

[81] J. Darbon and M. Sigelle. Image restoration with discrete constrained total variation part I: Fast and exact optimization. *J. Math. Imaging Vision*, 26:261–276, 2006. 10.1007/s10851-006-8803-0. 48

[82] J. Darbon and M. Sigelle. Image restoration with discrete constrained total variation part I: Fast and exact optimization. *J. Math. Imaging Vision*, 26(3):261–276, Dec. 2006. 92

[83] J. Darbon and M. Sigelle. Image restoration with discrete constrained total variation part II: Levelable functions, convex priors and non-convex cases. *J. Math. Imaging Vision*, 26(3):277–291, 12 2006. 20

[84] T. De Mazancourt and D. Gerlic. The inverse of a block-circulant matrix. *IEEE Trans. on Antennas and Propagation*, 31(5):808–810, 1983. 54

[85] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269271, 1959. 10

[86] O. Duchenne, J.-Y. Audibert, R. Keriven, J. Ponce, and F. Ségonne. Segmentation by transduction. In *Proceedings of IEEE Conference in Computer Vision and Pattern Recognition (CVPR)*, 2008. 71

[87] J. Edmonds and R. M. Karp. Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems. *J. ACM*, 19(2):248–264, 1972. 10

[88] N. El-Zehiry and L. Grady. Fast global optimization of curvature. In *Proceedings of the 23th IEEE Conference on Computer Vision and Pattern Recognition*, CVPR'10, 2010. 116

[89] A. Elmoataz, O. Lézoray, and S. Bougleux. Nonlocal discrete regularization on weighted graphs: A framework for image and manifold processing. *IEEE Trans. Image Process.*, 17(7):1047–1060, 2008. 4, 23, 25, 48, 57, 58

[90] E. Esser, X. Zhang, and T. F. Chan. A general framework for a class of first order primal-dual algorithms for convex optimization in imaging science. *SIAM J. Imaging Sci.*, 3(4):1015–1046, 2010. 55

[91] A. X. Falcão, R. A. Lotufo, and G. Araujo. The image foresting transformation. *IEEE Trans. on PAMI*, 26(1):19–29, 2004. 11, 72

[92] A. X. Falcão, J. K. Udupa, S. Samarasekera, S. Sharma, B. H. Elliot, and R. de A. Lotufo. User-steered image segmentation paradigms: Live wire and live lane. *GMIP*, 60(4):233–260, 1998. 71

[93] A. X. Falcao, J. Stolfi, and de Alencar. The image foresting transform: theory, algorithms, and applications. *IEEE Trans. on PAMI*, 26(1):19–29, 2004. 81

[94] C. L. Fefferman. Existence and smoothness of the Navier-Stokes equation, 2000. http://www.claymath.org/millennium/Navier-Stokes_Equations/navierstokes.pdf. 21

[95] P. Felzenszwalb and D. Huttenlocher. Efficient Graph-Based Image Segmentation. *Int. J. Comput. Vis.*, 59(2):167–181, Sept. 2004. 12

[96] B. Fischer and J. Modersitzki. Fast inversion of matrices arising in image processing. *Numerical Algorithms*. 54

[97] L. R. Ford and D. R. Fulkerson. Maximal flow through a network. *Canadian Journal of Mathematics*, 8:399–404, 1956. 7, 10, 31, 103

[98] D. Geman and G. Reynolds. Constrained restoration and the discovery of discontinuities. *IEEE Trans. on PAMI*, 14(3):367–383, March 1992. 130

[99] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Trans. on PAMI*, 6(6):721–741, 1984. 71

[100] S. Geman and D. McClure. Statistical methods for tomographic image reconstruction. In *Proc. 46th Sess. Int. Stat. Inst. Bulletin ISI*, volume 52, pages 4–21, Sept. 1987. 130

[101] T. Géraud, H. Talbot, and M. Van Droogenbroeck. Algorithms for mathematical morphology. In L. Najman and H. Talbot, editors, *Mathematical morphology: from theory to applications*, pages 345–373. Wiley-ISTE, 2010. 131

[102] G. Gilboa, J. Darbon, S. Osher, and T. Chan. Nonlocal convex functionals for image regularization, 2006. UCLA CAM Report 06-57. 48

[103] G. Gilboa and S. Osher. Nonlocal linear image regularization and supervised segmentation. *Multiscale Model. Simul.*, 6(2):595–630, 2007. 7, 32, 48, 49, 50

[104] R. Glowinski and P. Tallec. Augmented Lagrangian and operator-splitting methods in nonlinear mechanics. *Studies in Applied Mathematics, SIAM*, 1989. 109

[105] A. V. Goldberg and R. E. Tarjan. A new approach to the maximum flow problem. *STOC '86, Theory of computing*, pages 136–146, 1986. 10, 24

[106] D. Goldfarb and W. Yin. Second-order cone programming based methods for total variation image restoration. *SIAM J. Sci. Comput.*, 27:622–645, 2005. 22

[107] T. Goldstein, X. Bresson, and S. Osher. Geometric applications of the split Bregman method: Segmentation and surface reconstruction. CAM Report 09-06, UCLA, 2009. 103

[108] T. Goldstein and S. Osher. The split Bregman method for ℓ_1-regularized problems. *SIAM J. Imaging Sci.*, 2(2):323–343, 2009. 7, 38, 48, 109

[109] L. Grady. *Space-Variant Computer Vision: A Graph-Theoretic Approach*. PhD thesis, Boston University, Boston, MA, 2004. 67

[110] L. Grady. Random walks for image segmentation. *IEEE Trans. on PAMI*, 28(11):1768–1783, 2006. 8, 9, 71, 104

[111] L. Grady. A lattice-preserving multigrid method for solving the inhomogeneous Poisson equations used in image analysis. In D. Forsyth, P. Torr, and A. Zisserman, editors, *Proceedings of European Conference on Computer Vision (ECCV)*, volume 5303 of *LNCS*, pages 252–264. Springer, 2008. 87

[112] L. Grady. Minimal surfaces extend shortest path segmentation methods to 3D. *IEEE Trans. on PAMI*, 32(2):321–334, Feb. 2010. 71

[113] L. Grady and C. Alvino. The piecewise smooth Mumford-Shah functional on an arbitrary graph. *IEEE Trans. Image Process.*, 18(11):2547–2561, Nov. 2009. 4, 25, 92

[114] L. Grady, T. Schiwetz, S. Aharon, and R. Westermann. Random walks for interactive organ segmentation in two and three dimensions: Implementation and validation. In *Proc. In'l Conf. Medical Image Computing and Computer-Assisted Intervention*, pages 773–780, 2005. 30, 38

[115] L. Grady and E. L. Schwartz. Anisotropic interpolation on graphs: The combinatorial dirichlet problem. Technical report, Boston University, 2003. 8

[116] L. Grady and E. L. Schwartz. The Graph Analysis Toolbox: Image processing on arbitrary graphs. Technical Report CAS/CNS-TR-03-021, Department of Cognitive and Neural Systems, Boston University, Boston, MA, Aug. 2003. 67, 134

[117] L. Grady and E. L. Schwartz. Faster graph-theoretic image processing via small-world and quadtree topologies. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR'04)*, volume 2, pages 360–365. IEEE, June 2004. 23

[118] L. Grady and A. K. Sinop. Fast approximate random walker segmentation using eigenvector precomputation. In *Proceedings of IEEE Conference in Computer Vision and Pattern Recognition (CVPR)*, 2008. 71

[119] L. J. Grady and J. R. Polimeni. *Discrete Calculus: Applied Analysis on Graphs for Computational Science*. Springer-Verlag, New York, 2010. 3, 4, 5, 25, 74

[120] D. M. Greig, B. T. Porteous, and A. H. Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society*, 51(2):271–279, 1989. 74, 82

[121] L. Guigues, J. P. Cocquerez, and H. L. Men. Scale-sets image analysis. *Int. J. Comput. Vis.*, 68(3):289–317, 2006. 115

[122] P. Guillataud. *Contribution l'analyse dendroniques des images*. PhD thesis, Universit de Bordeaux I, 1992. 130

[123] P. Hanusse and P. Guillataud. Sémantique des images par analyse dendronique. In *8ème Reconnaissance des Formes et Intelligence Artificielle (RFIA)*, volume 2, pages 577–588, 1992. 130

[124] A. N. Hirani. *Discrete exterior calculus*. PhD thesis, Cal. Tech, 2003. 4

[125] T. Hirata. A unified linear-time algorithm for computing distance maps. *Information Processing Letters*, 58(3):129–133, 1996. 104

[126] D. S. Hochbaum and V. Singh. An efficient algorithm for co-segmentation. In *Proceedings of International Conference of Computer Vision (ICCV)*, 2009. 116

[127] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. *SIGGRAPH Comput. Graph.*, 26:71–78, July 1992. 111

[128] M. Iri. Theory of flows in continua as approximation to flows in networks. *Survey of Mathematical Programming*, 1979. 20, 24

[129] H. Ishikawa. Exact optimization for Markov random fields with convex priors. *IEEE Trans. on PAMI*, 25(10):1333–1336, Oct. 2003. 8, 20

[130] K. Jalalzai and A. Chambolle. Enhancement of blurred and noisy images based on an original variant of the total variation. In *Proceedings of International Conference on Scale Space and Variational Methods in Computer Vision*, SSVM '09, pages 368–376, Voss, Norway, 2009. 48

[131] A. C. Jalba and J. B. T. M. Roerdink. Efficient surface reconstruction using generalized coulomb potentials. *IEEE Transactions on Visualization and Computer Graphics*, 13:1512–1519, November 2007. 102, 103, 111

[132] R. Jenatton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for hierarchical sparse coding. *Journal of Machine Learning Research*, 12:2297–2334, Jul 2011. 48

[133] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *Int. J. Comput. Vis.*, 1(4):321–331, January 1988. 3

[134] J. P. Kaufhold. *Energy Formulations of Medical Image Segmentations*. PhD thesis, Boston University, 2000. 130

[135] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, SGP '06, pages 61–70, Aire-la-Ville, Switzerland, 2006. Eurographics Association. 102, 103, 111

[136] M. M. Kazhdan. Reconstruction of solid models from oriented point sets. In *Symposium on Geometry Processing*, pages 73–82, 2005. 102, 103, 111

[137] S.-J. Kim, K. Koh, M. Lustig, and S. Boyd. An efficient method for compressed sensing. In *Proceedings of International Conference on Image Processing(ICIP)*, pages 117–120. IEEE, 2007. 12

[138] P. Kohli, M. P. Kumar, and P. Torr. P3 & beyond: Solving energies with higher order cliques. In *Proceedings of IEEE Conference in Computer Vision and Pattern Recognition (CVPR)*, 2007. 72

[139] P. Kohli, L. Ladicky, and P. Torr. Robust higher order potentials for enforcing label consistency. In *Proceedings of IEEE Conference in Computer Vision and Pattern Recognition (CVPR)*, 2008. 72

[140] V. Kolmogorov and R. Zabih. Computing visual correspondence with occlusions via graph cuts. In *Proceedings of International Conference of Computer Vision (ICCV)*, pages 508–515, 2001. 8

[141] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE Transactions on PAMI*, 26(2):147–159, 2004. 7, 8

[142] K. Krajsek and H. Scharr. Diffusion filtering without parameter tuning: Models and inference tools. In *Proceedings of the 23th IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10)*, 2010. 130

[143] J. Kruger and R. Westermann. Linear algebra operators for GPU implementation of numerical algorithms. In *SIGGRAPH*, pages 908–916, 2003. 30, 38

[144] J. Kruskal. On the shortest spanning tree of a graph and the traveling salesman problem. *procs. Amer. Math. Soc.*, 7:48–50, 1956. 11, 71

[145] V. Lempitsky. Surface extraction from binary volumes with higher-order smoothness. In *Proceedings of the 23th IEEE Conference on Computer Vision and Pattern Recognition*, 2010. 107

[146] V. Lempitsky and Y. Boykov. Global Optimization for Shape Fitting. In *Proceedings of the 20th IEEE Conference on Computer Vision and Pattern Recognition (CVPR'2007)*, Minneapolis, USA, 2007. 102, 103

[147] V. S. Lempitsky, S. Roth, and C. Rother. Fusionflow: Discrete-continuous optimization for optical flow estimation. In *Proceedings of IEEE Conference in Computer Vision and Pattern Recognition (CVPR)*, 2008. 89

[148] A. Levin, D. Lischinski, and Y. Weiss. A closed form solution to natural image matting. *IEEE Trans. on PAMI*, 30(2):228–242, 2008. 71

[149] E. Levitan and G. Herman. A maximum a posteriori probability expectation maximization algorithm for image reconstruction in emission tomography. *IEEE Transactions on Medical Imaging*, MI-6(3):185–192, Sept. 1987. 130

[150] Y. Li and D. P. Huttenlocher. Learning for optical flow using stochastic optimization. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 379–391, 2008. 130

[151] Y. Li, J. Sun, C. Tang, and H. Shum. Lazy snapping. In *SIGGRAPH*, pages 303–308, 2004. 71

[152] P. L. Lions and B. Mercier. Splitting algorithms for the sum of two nonlinear operators. *SIAM J. Numer. Anal.*, 16:964–979, 1979. 15

[153] R. A. Lippert. Discrete approximations to continuum optimal flow problems. *Studies in Applied Mathematics*, 117(4):321–333, 2006. 22, 34

[154] F. Malgouyres. Minimizing the total variation under a general convex constraint for image restoration. *IEEE Trans. Image Process.*, 11(12):1450–1456, 2002. 49

[155] P. Matas, E. Dokládalova, M. Akil, T. Grandpierre, L. Najman, M. Poupa, and V. Georgiev. Parallel Algorithm for Concurrent Computation of Connected Component Tree. In *Advanced Concepts for Intelligent Vision Systems (ACIVS'08)*, volume 5259/2008 of *Lecture Notes in Computer Science*, pages 230–241. Springer-Verlag, Oct. 2008. 131

[156] F. Meyer. Minimum spanning forests for morphological segmentation. In *Proceedings of international symposium on mathematical morphology (ISMM'94)*, pages 77–84, 1994. 12

[157] F. Meyer and S. Beucher. Morphological segmentation. *Journal of Visual Communication and Image Representation*, 1(1):21–46, Sept. 1990. 131

[158] F. Meyer and L. Najman. Segmentation, minimum spanning tree and hierarchies. In L. Najman and H. Talbot, editors, *Mathematical morphology: from theory to applications*, chapter 9, pages 255–287. Wiley-ISTE, 2010. 131

[159] H. S. Michael, M. J. Black, and H. W. Haussecker. Image statistics and anisotropic diffusion. In *Proceedings of International Conference of Computer Vision (ICCV)*, pages 840–847, 2003. 130

[160] E. N. Mortensen and W. A. Barrett. Intelligent scissors for image composition. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '95, pages 191–198, New York, NY, USA, 1995. ACM. 10

[161] E. N. Mortensen and W. A. Barrett. Interactive segmentation with intelligent scissors. *Graphical Models and Image Processing*, 60(5):349–384, 1998. 71

[162] D. Mumford and J. Shah. Optimal approximations by piecewise smooth functions and associated variational problems. 71, 92

[163] L. Najman. Ultrametric watersheds. In M. Wilkinson and J. Roerdink, editors, *9th international symposium on mathematical morphology (ISMM'09)*, volume 5720 of *Lecture Notes in Computer Science*, pages 181–192. Springer-Verlag, Aug. 2009. 115

[164] L. Najman. Ultrametric watersheds: a bijection theorem for hierarchical edge-segmentation. *CoRR*, abs/1002.1887, 2010. 115

[165] L. Najman and M. Couprie. Building the component tree in quasi-linear time. *IEEE Trans. Image Process.*, 15(11):3531–3539, 2006. 104, 131

[166] L. Najman and M. Schmitt. Watershed of a Continuous Function. *Signal Processing*, 38:99–112, 1994. Special issue on Mathematical Morphology. 70, 81

[167] L. Najman and M. Schmitt. Geodesic Saliency of Watershed Contours and Hierarchical Segmentation. *IEEE Trans. on PAMI*, 18(12):1163–1173, 1996. 115, 131

[168] L. Najman and H. Talbot, editors. *Mathematical Morphology: from theory to applications*. Wiley-ISTE, July 2010. 71

[169] Y. Nesterov. Gradient methods for minimizing composite objective function. CORE Discussion Papers 2007076, Universit catholique de Louvain, Center for Operations Research and Econometrics (CORE), Sept. 2007. 7

[170] M. Nikolova, S. Esedoglu, and T. F. Chan. Algorithms for finding global minimizers of image segmentation and denoising models. *SIAM J. Math. Anal.*, 66(5):1632–1648, 2006. 20

[171] R. Nozawa. Max-flow min-cut theorem in an anisotropic network. *Osaka J. Math*, 27:805–842, 1990. 21

[172] R. Nozawa. Examples of max-flow and min-cut problems with duality gaps in continuous networks. *Mathematical Programming*, 63:213–234, 1994. 10.1007/BF01582067. 21, 34

[173] M. d. J. L. O.J. Morris and A. G. Constantinides. Graph theory for image analysis: an approach based on the shortest spanning tree. In *IEEE Proc. on Communication, Radar, and Signal*, number 2, pages 146–152, 1986. 12

[174] C. H. Papadimitriou and K. Streiglitz. *Combinatorial optimization, algorithms and complexity*. Dover publications, 1998. 9

[175] N. Paragios. *Geodesic Active Regions and Level Set Methods : Contributions and Applications in Artificial Vision*. PhD thesis, INRIA Sophia Antipolis, Jan. 2000. 3

[176] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. on PAMI*, 12(7):629–639, July 1990. 92, 93

[177] J.-C. Pesquet and N. Pustelnik. A parallel inertial proximal optimization method, 2011. Preprint. 16, 17, 53

[178] G. Peyre and L. D. Cohen. *Geodesic Methods for Shape and Surface Processing*, volume 13, chapter Advances in Computational Vision and Medical Image Processing: Methods and Applications, pages 29–56. Springer, 2008. 11

[179] T. Pock, D. Cremers, H. Bischof, and A. Chambolle. An algorithm for minimizing the Mumford-Shah functional. In *Int. Conf. Comp. Vis.*, 2009. 34

[180] T. Pock, T. Schoenemann, G. Graber, H. Bischof, and D. Cremers. A convex formulation of continuous multi-label problems. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 792–805, 2008. 21, 34

[181] T. Pock, T. Schoenemann, G. Graber, H. Bischof, and D. Cremers. A convex formulation of continuous multi-label problems. In *Proceedings of the 10th European Conference on Computer Vision: Part III*, ECCV '08, pages 792–805, Berlin, Heidelberg, 2008. Springer-Verlag. 47

[182] R. Prim. Shortest connection networks and some generalizations. *Bell Syst. Tech. J.*, 36:1389–1401, 1957. 11, 71

[183] N. Pustelnik. *Proximal methods for the resolution of inverse problems : application to positron emission tomography*. PhD thesis, Université Paris Est, 2010. 16

[184] N. Pustelnik, C. Chaux, and J.-C. Pesquet. Parallel proximal algorithm for image restoration using hybrid regularization. *IEEE Trans. Image Process.*, 2011. 48

[185] E. R. R. Goldenberg, R. Kimmel and M. Rudzsky. Fast geodesic active contours. *IEEE Trans. Image Process.*, 10(10):1467–1475, 2001. 10

[186] I. Ragnemalm. The euclidean distance transform in arbitrary dimensions. *Pattern Recognition Letters*, 14(11):883 – 888, 1993. 104, 106

[187] J. Roerdink and A. Meijster. The watershed transform: Definitions, algorithms, and parallellization strategies. *Fund. Informaticae*, 41:187–228, 2000. 70, 81

[188] C. Rother, V. Kolmogorov, and A. Blake. "GrabCut" — Interactive foreground extraction using iterated graph cuts. In *SIGGRAPH*, pages 309–314, 2004. 8, 39, 71, 84

[189] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60(1-4):259–268, 1992. 6, 47, 92

[190] P. Salembier, A. Oliveras, and L. Garrido. Anti-extensive connected operators for image and sequence processing. *IEEE Trans. Image Process.*, 7(4):555–570, Apr. 1998. 130

[191] K. G. G. Samuel and M. F. Tappen. Learning optimized MAP estimates in continuously-valued MRF models. In *Proceedings of the 22th IEEE Conference on Computer Vision and Pattern Recognition (CVPR'09)*, 2009. 130

[192] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int. J. Comput. Vis.*, 47:7–42, 2001. 100

[193] U. Schmidt, Q. Gao, and S. Roth. A generative perspective on MRFs in low-level vision. In *Proceedings of the 23th IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10)*, 2010. 130

[194] J. A. Sethian. *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*. Cambridge University Press on Applied and Computational Mathematics, 1999. 3, 10

[195] S. Setzer. Split Bregman algorithm, douglas-rachford splitting and frame shrinkage. In *International Conference on Scale Space and Variational Methods in Computer Vision*, SSVM '09, pages 464–476, Berlin, Heidelberg, 2009. Springer-Verlag. 109

[196] R. Shen, I. Cheng, X. Li, and A. Basu. Stereo matching using random walks. In *International Conference on Pattern Recognition*, 2008. 89

[197] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. on PAMI*, 22(8):888–905, 2000. 71

[198] D. Singaraju, L. Grady, A. K. Sinop, and R. Vidal. Continuous valued MRFs for image segmentation. In A. Blake, P. Kohli, and C. Rother, editors, *Advances in Markov Random Fields for Vision and Image Processing*. MIT Press, 2010. in press. 128

[199] D. Singaraju, L. Grady, and R. Vidal. P-brush: Continuous valued MRFs with normed pairwise distributions for image segmentation. In *Proceedings of IEEE Conference in Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, IEEE, June 2009. 74

[200] A. K. Sinop and L. Grady. A seeded image segmentation framework unifying graph cuts and random walker which yields a new algorithm. In *Proceedings of International Conference of Computer Vision (ICCV)*, 2007. 69, 71, 72, 73, 74, 85, 89, 114, 125

[201] M. Stoer and F. Wagner. A simple min-cut algorithm. *J. ACM*, 44:585–591, July 1997. 10

[202] G. Strang. l^1 and l^∞ approximation of vector fields in the plane. In *Nonlinear Partial Differential Equations in Applied Science; Proceedings of the U.S. - Japan Seminar*, volume 5 of *Lecture Notes in Num. Appl. Anal.*, pages 273–288, 1982. 74

[203] G. Strang. Maximum flows through a domain. *J. Math. Prog.*, 26:123–143, 1983. 20, 24, 37, 103

[204] G. Strang. Maximum flows and minimum cuts in the plane. *Journal of Global Optimization*, 2008. 21

[205] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for Markov random fields with smoothness-based priors. *IEEE Trans. on PAMI*, 30(6):1068–1080, 2008. 89

[206] M. F. Tappen, C. Liu, E. H. Adelson, and W. T. Freeman. Learning Gaussian conditional random fields for low-level vision. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR'07)*, 2007. 130

[207] R. E. Tarjan. Efficiency of a good but not linear set union algorithm. *J. ACM*, 22:215–225, April 1975. 11, 131

[208] G. Taubin, T. Zhang, and G. H. Golub. Optimal surface smoothing as filter design. In *Proceedings of the 4th European Conference on Computer Vision - Volume I*, ECCV '96, pages 283–292, London, UK, 1996. Springer-Verlag. 67

[209] M. Unger, T. Pock, and H. Bischof. Interactive globally optimal image segmentation. Technical Report ICG-TR-08/02, Graz University of Technology, Gratz, Austria, jul 2008. 21, 31, 34, 41

[210] M. Unger, T. Pock, W. Trobin, D. Cremers, and H. Bischof. TVSeg - Interactive total variation based image segmentation. In *British Machine Vision Conference*, 2008. 21, 26, 34, 71

[211] E. Van den Berg and M. P. Friedlander. Probing the Pareto frontier for basis pursuit solutions. *SIAM J. Sci. Comput.*, 31(2):890–912, 2008. 54

[212] O. Veksler. Star shape prior for graph-cut image segmentation. In *Proceedings of the 10th European Conference on Computer Vision: Part III*, pages 454–467, Berlin, Heidelberg, 2008. Springer-Verlag. 116

[213] L. Vese and T. Chan. A multiphase level set framework for image segmentation using the Mumford and Shah model. *Int. J. of Comp. Vis.*, 50(3):271–293, 2002. 92

[214] S. Vicente, V. Kolmogorov, and C. Rother. Graph cut based image segmentation with connectivity priors. In *Proceedings of IEEE Conference in Computer Vision and Pattern Recognition (CVPR)*, 2008. 71

[215] L. Vincent. Morphological grayscale reconstruction in image analysis: Applications and efficient algorithms. *IEEE Trans. Image Process.*, 2:176–201, 1993. 131

[216] L. Vincent and P. Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Trans. on PAMI*, 13(6):583–598, June 1991. 70

[217] M. H. F. Wilkinson, H. Gao, W. H. Hesselink, J.-E. Jonker, and A. Meijster. Concurrent computation of attribute filters on shared memory parallel machines. *IEEE Trans. on PAMI*, 30(10):1800–1813, 2008. 131

[218] O. Woodford, P. Torr, I. Reid, and A. Fitzgibbon. Global stereo reconstruction under second-order smoothness priors. *IEEE Trans. on PAMI*, 31:2115–2128, 2009. 100

[219] A. Yang, J. Wright, Y. Ma, and S. Sastry. Unsupervised segmentation of natural images via lossy data compression. *Computer Vision and Image Understanding*, 110(2):212–225, May 2008. 85

[220] K. K. Yann LeCun and C. Farabet. Convolutional networks and applications in vision. In *Proceedings of International Symposium on Circuits and Systems (ISCAS'10)*, 2010. 116

[221] J. Ye, X. Bresson, T. Goldstein, and S. Osher. A fast variational method for surface reconstruction from sets of scattered points, 2010. UCLA CAM Report 10-01. 103

[222] V. K. Yuri Boykov. Computing geodesics and minimal surfaces via graph cuts. *Proceedings of International Conference of Computer Vision (ICCV)*, pages 26–33, 2003. 20

[223] C. Zach, D. Gallup, J.-M. Frahm, and M. Niethammer. Fast global labeling for real-time stereo using multiple plane sweeps. In *Vis. Mod. Vis*, 2008. 21, 34, 100

[224] W. Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33:452–473, 1977. 42, 43

[225] X. Zhang, M. Burger, X. Bresson, and S. Osher. Bregmanized nonlocal regularization for deconvolution and sparse reconstruction. *SIAM J. Imaging Sci.*, 3(3):253–276, 2010. 48

[226] H.-K. Zhao, S. Osher, B. Merriman, and M. Kang. Implicit, nonparametric shape reconstruction from unorganized points using a variational level set method. *Computer Vision and Image Understanding*, 80:295–319, 1998. 102