Int J Comput Vis (2009) 84: 288–307
DOI 10.1007/s11263-009-0238-9

On Total Variation Minimization and Surface Evolution Using Parametric Maximum Flows

Antonin Chambolle · Jérôme Darbon

Received: 23 June 2005 / Accepted: 8 April 2009 / Published online: 23 April 2009

Abstract In a recent paper Boykov et al. (LNCS, Vol. 3953, pp. 409–422, 2006) propose an approach for computing curve and surface evolution using a variational approach and the *geo-cuts* method of Boykov and Kolmogorov (International conference on computer vision, pp. 26–33, 2003). We recall in this paper how this is related to well-known approaches for mean curvature motion, introduced by Almgren et al. (SIAM Journal on Control and Optimization 31(2):387–438, 1993) and Luckhaus and Sturzenhecker (Calculus of Variations and Partial Differential Equations 3(2):253–271, 1995), and show how the corresponding problems can be solved with *sub-pixel accuracy* using Parametric Maximum Flow techniques. This provides interesting algorithms for computing crystalline curvature motion, possibly with a forcing term.

Keywords Crystalline and anisotropic mean curvature flow · Variational approaches · Total variation · Submodular functions · Max-flow/min-cut · Parametric max-flow algorithms

1 Introduction

Boykov et al. (2006) discuss the possibility of evolving curves and surfaces by their mean curvature by solving a discrete minimal surface problem, using a maximum flow/graph-cut algorithm (Ahuja et al. 1993). This kind of technique has become very popular in the past year in image processing, for segmentation problems but also stereo correspondence, etc., in particular since the apparition of quite efficient algorithms (Boykov and Kolmogorov 2004) for graphs with low-connectivity, typically in use in this kind of applications.

The idea of Boykov et al. consists of evolving a contour C_t by finding C_{t+dt} through the minimization of the following variational problem

$$\min_C F(C) + \frac{1}{2dt}\operatorname{dist}(C, C_t), \tag{1}$$

where $F(C)$ is an energy (in general, the length or surface of C) and $\operatorname{dist}(C, C_t)$ is (approximately) the L^2-distance, given by

$$\operatorname{dist}(C, C_t) = 2\int_{\Delta C} \operatorname{dist}(p, C)\, dp,$$

where ΔC is the region between the two curves or surfaces C and C_t. They conjecture that if for instance F is the Euclidean length or surface of C, then this process will approximate the Mean Curvature Flow, which is in this case the gradient flow of F.

It turns out that this approach to the mean curvature flow has been proposed in the early 90's by Almgren et al. (1993) and simultaneously by Luckhaus and Sturzenhecker (1995), in the following way: we consider ϕ a convex, one homogeneous function in $\mathbb{R}^N$ (with $(1/c)|x| \le \phi(x) \le c|x|$ for some $c > 0$) and the corresponding anisotropic perimeter of $E \subset \mathbb{R}^N$

$$Per_\phi(E) = \int_{\partial E} \phi(\nu_E(x))\, ds,$$

A. Chambolle's research supported by ANR project "MICA", grant ANR-08-BLAN-0082.
J. Darbon's research supported by ONR grant N000140710810.

A. Chambolle (✉)
CMAP, Ecole Polytechnique, CNRS, 91128 Palaiseau, France
e-mail: antonin.chambolle@polytechnique.fr

J. Darbon
Mathematics Department, UCLA, Los Angeles, USA
e-mail: jerome@math.ucla.edu

where $\nu(x)$ is the inner normal to E at x and ds the surface integral on ∂E (more generally, $ds = d\mathcal{H}^{N-1}$, the $(N-1)$-dimensional Hausdorff measure). Given a set E, let

$$d_E(x) = \operatorname{dist}(x, E) - \operatorname{dist}(x, \mathbb{R}^N \setminus E)$$

be the signed distance to the boundary of E (negative inside, and positive outside). Then, for $E \subset \mathbb{R}^N$ (bounded, or of bounded complement, so that ∂E is bounded), and given a time-step $h > 0$, we define $T_h E$ as a solution of

$$\min_{F \subset \mathbb{R}^N} Per_\phi(F) + \frac{1}{h}\int_{F \triangle E} |d_E(x)|\,dx, \tag{2}$$

where $F \triangle E$ is the symmetric difference between the sets F and E: this is exactly another way of writing (1). Then the above-mentioned authors define a discrete-in-time evolution $E_h(t)$ starting from E by letting $E_h(t) = T_h^{[t/h]}E$ where $[\cdot]$ denotes the integer part. It is shown in Almgren et al. (1993) that if E and ϕ are smooth enough, then as $h \to 0$, $\partial E_h(t)$ converges (in the Hausdorff sense) to $\partial E(t)$ where $E(t)$ is the Mean ϕ-Curvature Flow starting from E, which is in some sense, as expected, the gradient flow of the perimeter $Per_\phi(E)$ (this is defined as the motion where ∂E evolves along its normal by the opposite of its "anisotropic mean curvature" κ^ϕ, see for instance Almgren et al. (1993), Bellettini and Paolini (1996) for a complete definition). Convergence results for generalized evolutions are found in Chambolle (2004), Chambolle and Novaga (2007).

The idea, here, is that the Euler-Lagrange equation for Problem (2) is

$$h\kappa^\phi_F(x) + d_E(x) = 0 \tag{3}$$

and since $d_E(x)$ measures exactly how far the point x has moved away from ∂E along its (outer) normal, this may be seen as a implicit time-discrete scheme for the mean curvature motion.

Remark 1.1 Since

$$\begin{aligned}\int_{F \triangle E} |d_E(x)|\,dx &= \int_{F\setminus E} d_E(x)\,dx - \int_{E\setminus F} d_E(x)\,dx\\ &= \int_F d_E(x)\,dx - \int_E d_E(x)\,dx,\end{aligned}$$

we observe that, whenever E is bounded, it is equivalent to minimize (2) and to solve

$$\min_{F \subset \mathbb{R}^N} Per_\phi(F) + \frac{1}{h}\int_F d_E(x)\,dx. \tag{4}$$

It can be shown (see Chambolle 2004) that this algorithm enjoys a monotonicity property, in the sense that if $E \subset E'$ then the minimal (respectively maximal) solution $T_h E$ is contained in the minimal (resp., maximal) solution $T_h E'$. This yields the convergence of $E_h(t)$ to the generalized flow, in the sense of viscosity solutions, at least when this is unique. See also Chambolle and Novaga (2007).

Adding an external force (forcing term along the normal) in this formulation is quite easy: if d_E is replaced in (4) with a term of the form $d_E(x) - hg(t,x)$, then (3) turns into

$$d_E(x) = -h\kappa^\phi_F(x) + hg(x,t)$$

which means that now, x moves along the normal of h times the opposite of the curvature plus the forcing term g.

This approach has been widely studied in the past years, mostly as a tool for the theoretical study of the anisotropic and "crystalline" mean curvature motion (the crystalline case is the case where ϕ is non smooth, and is of particular importance here since the discrete approaches we consider will *only* work in such cases). See in particular Bellettini et al. (2006), Caselles and Chambolle (2006), Chambolle (2004), Chambolle and Novaga (2008).

In this paper, we provide a framework for computing such evolutions by maxflow/mincut algorithms. The idea in Boykov et al. (2006) is to solve a discrete version of (1) using such combinatorial optimization techniques. However, such an approach produces a discrete set C defined on a discrete grid, and this has then to be refined a lot to capture the motion with a good precision. We show that problems (1) or (2) are related to a convex minimization problem known in image processing as the "Rudin-Osher-Fatemi" (ROF) problem (Total Variation minimization with a quadratic penalization). This connection, exploited both in the continuous and discrete setting, allows simultaneously to (i) use maxflow/mincut approaches, in a "parametric" way (Gallo et al. 1989; Hochbaum 2001), to solve efficiently the discrete ROF problem; (ii) solve a discretized (ROF) problem to derive with a good (sub-pixel) precision an approximation of the set F which minimizes (2). In her seminal work, Hochbaum (2001) proposes an approach to solve the ROF model using parametric maximum-flow. This approach has been considered in Goldfarb and Yin (2007) and we also refer the reader to Juan and Boykov (2006), Kholi and Torr (2005) for similar ideas used in computer vision.

In the next section, we recall some results that link Problem (4) to the celebrated "Rudin-Osher-Fatemi" problem in computer vision, and provide an approach for its resolution. Then, in Sect. 3, we introduce our discrete setting, "discrete total variation" functionals, and basically state the same results as in Sect. 2 in this new setting. These properties lead to an efficient algorithm for the *ROF* problem, which we describe in Sect. 4. It is essentially a variant of the parametric max-flow algorithm (Gallo et al. 1989) and has been first proposed by Hochbaum (2001). Its most salient features are that it solves the problem in polynomial time and up to an arbitrary precision. Eventually, we propose our technique for solving surface evolution problems in Sect. 5

and show numerical examples in Sect. 6. Various appendices complete the paper. A modified version of the Boykov and Kolmogorov's (2004) maximum flow code, that implements the parametric approach to solves the *ROF* problem, is available through the authors' personal web page.

Let us point out that while a first version of this paper was under complete rewritting, very similar ideas were published by Kolmogorov et al. (2007), with applications to the precise computation of energy-minimizing contours.

2 Minimal Surface Problems and Total Variation Minimization

The total variation $|Du|(\Omega)$ of a function $u \in L^1(\Omega)$ (in this section, Ω is an open subset of $\mathbb{R}^N$, $N \geq 1$—and typically $N = 2, 3$), is classically defined by duality as follows (Giusti 1984)

$$|Du|(\Omega) = \int_\Omega |Du| := \sup\left\{ \int_\Omega u \operatorname{div} \xi : \xi \in C_c^1(\Omega; \mathbb{R}^N), \|\xi(x)\| \leq 1\ \forall x \in \Omega \right\}, \tag{5}$$

while the perimeter of a set in Ω is the total variation of its characteristic function. It is well known that level sets of function which minimize the total variation are themselves minimal surfaces (that is, sets with minimal perimeter, at least up to compactly supported perturbations), and this fact is a main tool for the study of these surfaces and their regularity (Federer 1969; Giusti 1984). However, the relationship between surfaces with prescribed curvature (minimizing their perimeter plus an external field) and total variation minimization with an additional penalization of the function seems to have been less often noticed, though it relies on the same celebrated "co-area formula" (Federer 1969; Giusti 1984):

$$|Du|(\Omega) = \int_{-\infty}^{+\infty} Per(\{u > z\}, \Omega)\, dz \tag{6}$$

Let us just state the main equivalence:

Proposition 2.1 *Let u be the (unique) solution of*

$$\min_{u \in BV(\Omega)} \lambda \int_\Omega |Du| + \frac{1}{2} \int_\Omega |u(x) - g(x)|^2\, dx. \tag{7}$$

Then, for all $z > 0$, the super-level sets $E_z = \{u \geq z\}$ and $E'_z = \{u > z\}$ are both minimizers of

$$\min_{E \subseteq \Omega} \lambda Per(E, \Omega) + \int_E z - g(x)\, dx. \tag{8}$$

Conversely, any minimizer E of (8) is between E'_z and E_z: $E'_z \subseteq E \subseteq E_z$. In particular, for all z but a countable set in $\mathbb{R}$, $\{u = z\}$ has zero measure and the solution of (8) is unique up to a negligible set.

The proof of this proposition is relatively easy, and quite classical, but is out of the scope of this paper. The first part (the super-level sets are minimizing) is shown for instance in Chambolle (2004), while the second (the converse) comes from a comparison principle for the minimizers of (8) which appears in Alter et al. (2005):

Lemma 2.2 (Alter et al. 2005, Lemma 4, (i)) *Let $z > z'$ and E_z, $E_{z'}$ minimize (8) for the respective values z and z': then $E_z \subseteq E'_z$.*

An observation which is clear from the proofs is that these properties remain true if the term

$$\frac{1}{2} \int_\Omega |u(x) - g(x)|^2\, dx$$

in (7) is replaced with a term of the kind $\int_\Omega \Psi(x, u(x))\, dx$ where Ψ is uniformly convex and C^1 with respect to $u(x)$, and $\int_E z - g(x)\, dx$ is replaced with $\int_E \partial\Psi/\partial z(x, z)\, dx$ in (8). The cases where Ψ is simply convex, or lacks regularity, are also interesting, and partial results still hold in these cases: see Chan and Esedoḡlu (2005), Darbon (2005) where similar ideas are developed. We mention that theses principles have also been used in a series of recent papers for studying the properties of minimizers of (7), see Allard (2007), Caselles et al. (2007).

Also, in a more general setting, one may replace the total variation (5) with an anisotropic total variation

$$\int_\Omega \phi(Du) := \sup\left\{ \int_\Omega u \operatorname{div} \xi : \xi \in C_c^1(\Omega; \mathbb{R}^N), \phi^\circ(\xi(x)) \leq 1 \forall x \in \Omega \right\},$$

where ϕ° is the polar of ϕ, defined by $\phi^\circ(\xi) = \sup_{\phi(\nu) \leq 1} \nu \cdot \xi$ (and $\phi(\nu) = \sup_{\phi^\circ(\xi) \leq 1} \nu \cdot \xi$). Then the perimeter in (8) is replaced by the corresponding anisotropic perimeter

$$Per_\phi(E) = \int_\Omega \phi(D\chi^E) = \int_{\partial E} \phi(\nu_E)\, d\sigma,$$

where the last expression holds if E is smooth enough and ν_E is then the inner normal to ∂E. This will be useful in the sequel, since the total variations and perimeters that are approximated by discrete methods in this paper are strongly anisotropic.

The equivalence in Proposition 2.1 is interesting for both studies of problems (7) and (8), since it extends the knowledge of some properties of solutions of one to the other, see for instance Caselles et al. (2007).

It also gives a practical way to solve (4). Indeed, we deduce that a solution is given by $F = \{u \le 0\}$ where u is the minimizer of

$$\min_{u\in BV(\Omega)} \int_\Omega \phi(Du) + \frac{1}{2h}\int_\Omega (u(x) - d_E(x))^2\, dx \tag{9}$$

at least as soon as Ω is "large enough" (with respect to E). Problem (9) is the classical convex problem in image processing known as the "Rudin-Osher-Fatemi" denoising problem, and can be solved in many ways. Although it does not seem that standard ways for solving (9) yield very efficient algorithms for the mean curvature flow, we will introduce now a discrete setting in which the resolution of such problems is very fast using combinatorial optimization approaches, and leads to efficient algorithms for the crystalline mean curvature flow (and probably many other applications in shape computation/optimization).

We now introduce the discrete setting and what can be seen as the discrete analogs of Proposition 2.1 and Lemma 2.2.

3 Discrete Perimeters and Discrete Total Variation

Most of the results in these section are well known in combinatorial optimization, we present them for completeness (Lee 2004; Murota 2003), and, also, to stress the similarities with the continuous setting (in the continuous setting, a general overview of these topics is found in Bouchitté 1998). By analogy with this setting, we define a discrete total variation as a convex, nonnegative function $J : \mathbb{R}^N \to [0, +\infty]$ satisfying a discrete *co-area* formula:

$$J(u) = \int_{-\infty}^{+\infty} J(\chi^{\{u\ge z\}})\, dz \tag{10}$$

where $\chi^{\{u\ge z\}} \in \{0,1\}^N$ denotes the vector such that $\chi_i^{\{u\ge z\}} = 0$ if $u_i \le z$ and $\chi_i^{\{u\ge z\}} = 1$ if $u_i \le z$.

By analogy, given $E \subseteq \{1,\dots,N\}$ we also define a discrete perimeter as $P_J(E) := J(\chi^E)$ where the characteristic vector χ^E is defined by $\chi_i^E = 1$ if $i \in E$ and $\chi_i^E = 0$ else.

We assume that J is not identically $+\infty$. Under these assumptions, it is easy to derive from (10) the following properties:

Proposition 3.1 *Let J be a discrete total variation. Then*:

1. *J is positively homogeneous*: *$J(\lambda u) = \lambda J(u)$ for any $u \in \mathbb{R}^N$ and $\lambda \ge 0$.*
2. *J is invariant by addition of a constant*: *$J(c\mathbf{1} + u) = J(u)$ for any $u \in \mathbb{R}^N$ and $c \in \mathbb{R}$, where $\mathbf{1} = (1,\dots,1) \in \mathbb{R}^N$ is a constant vector. In particular, $J(\mathbf{1}) = 0$.*
3. *J is lower-semicontinuous.*
4. *$p \in \partial J(u) \Leftrightarrow (\forall z \in \mathbb{R}, p \in \partial J(\chi^{\{u\ge z\}}))$.*
5. *J is submodular*: *for any $u, u' \in \{0,1\}^N$,*

$$J(u \vee u') + J(u \wedge u') \le J(u) + J(u'). \tag{11}$$

More generally, this will hold for any $u, u' \in \mathbb{R}^N$.

Conversely, if $J : \{0,1\}^N \to [0,+\infty]$ is a submodular function with $J(0) = J(\mathbf{1}) = 0$, then the co-area formula (10) extends it to $\mathbb{R}^N$ into a convex function, hence a discrete total variation.

In the 4th point, the *subgradient* $\partial J(v)$ of J at v is defined as the set of vectors p such that $J(v') \ge J(v) + p\cdot(v' - v)$ for any v'. Equivalently, in this case, it is the set of $p \in \partial J(0)$ with $J(v) = p \cdot v$. See Ekeland and Témam (1999), Rockafellar (1997) for details.

Remark 3.2 If J is a general real-valued submodular function with $J(0) = 0$, then it can be extended in a similar way to a convex functions of *non-negative* vectors $u \in \mathbb{R}_+^N$, by the same formula as (10) but where the integral on $\mathbb{R}$ is replaced with an integral on $\mathbb{R}_+ = [0, +\infty)$. This is well-known in optimization theory as the Lovász' extension of J (Lovász 1983), or the Choquet integral (see for instance Lee 2004, Chap. 8 and Murota 2003).

We prove the proposition in Appendix A. A typical example of discrete total variation (and associated perimeters) is (for $u = u_{i,j}$ a $2D$ image in $\mathbb{R}^{M\times M}$, hence $N = M^2$ here)

$$J(u) = \sum_{\substack{1\le i<M\\1\le j\le M}} |u_{i+1,j} - u_{i,j}| + \sum_{\substack{1\le i\le M\\1\le j<M}} |u_{i,j+1} - u_{i,j}| \tag{12}$$

but infinitely many other examples can be built, involving interaction between neighboring pixels further and further apart. Also, less standard convex functions enter this framework, such as the "oscillation"

$$J(u) = \sum_{\substack{1\le i<M\\1\le j<M}} \max\{u_{i,j}, u_{i+1,j}, u_{i,j+1}, u_{i+1,j+1}\} - \sum_{\substack{1\le i<M\\1\le j<M}} \min\{u_{i,j}, u_{i+1,j}, u_{i,j+1}, u_{i+1,j+1}\} \tag{13}$$

which is also, in some sense, an approximation of an anisotropic total variation. See Appendix B for how the minimization of this example can be implemented.

Another particular example is a pairwise circulant oscillation involving three pixels (which might be useful for images defined on $2D$ hexagons lattice endowed with the

6-connectivity) defined as follows:

$$J(u) = \sum_{\substack{1\le i<M\\1\le j\le M}} \max\{|u_{i,j}-u_{i+1,j}|,\\ |u_{i,j}-u_{i,j+1}|, |u_{i+1,j}-u_{i,j+1}|\}. \tag{14}$$

The latter reduces to pairwise interactions of the form given by (12) by considering the following: without loss of generality we can assume that we have $u_{i,j} \le u_{i+1,j} \le u_{i,j+1}$ and by noticing that $|u_{i,j}-u_{i,j+1}| = |u_{i,j}-u_{i+1,j}| + |u_{i+1,j}-u_{i,j+1}|$ we get that (14) amounts to

$$J(u) = \frac{1}{2}\sum_{\substack{1\le i<M\\1\le j\le M}} (|u_{i,j}-u_{i+1,j}| + |u_{i,j}-u_{i,j+1}|\\ + |u_{i+1,j}-u_{i,j+1}|).$$

Using similar arguments, one can show that any circulant oscillations involving any odd number (greater than 1) of pixels can be casted into a pairwise interactions form.

If J is a discrete total variation, then the discrete counterpart of Proposition 2.1 holds:

Proposition 3.3 *Let $g \in \mathbb{R}^N$ and let $u \in \mathbb{R}^N$ be the (unique) solution of*

$$\min_{u\in\mathbb{R}^N} \lambda J(u) + \frac{1}{2}\|u-g\|^2. \tag{15}$$

Then, for all $z>0$, the characteristic functions of the super-level sets $E_z = \{u \ge z\}$ and $E'_z = \{u > z\}$ (which are different only if $z \in \{u_i, i=1,\dots,N\}$) are respectively the largest and smallest minimizer of

$$\min_{\theta\in\{0,1\}^N} \lambda J(\theta) + \sum_{i=1}^N \theta_i(z-g_i). \tag{16}$$

Observe that a consequence of this proposition is that problems (16) have at most N different solutions, as z runs from $-\infty$ to $+\infty$, however in practice this number can be much smaller, since the level sets $\{u=z\}$, when nonempty, often contain more than just one vertex (up to containing *all* vertices when λ is large enough). This proposition is shown in Chambolle (2005), Darbon and Sigelle (2006), but is also a consequence of the representation we will introduce in the next section for problems (15) and (16). Again, here, the quadratic term $\|u-g\|^2$ can be replaced with any term of the form $\sum_i \Psi_i(u_i)$, with Ψ_i strictly convex and C^1, replacing then $\theta_i(z-g_i)$ in (16) with $\Psi'_i(z)$. We postpone the proof of this result to the Appendix C. Let us just mention here that it relies on the following discrete counterpart of Lemma 2.2 which is a consequence of the submodularity of J:

Lemma 3.4 *Let $z>z'$ and θ, θ' solve (8) for the respective values z and z' of the parameter. Then $\theta \le \theta'$ (in other words, $\{\theta=1\}\subseteq\{\theta'=1\}$).*

This key property is proved, at least for a particular case of submodular functions, in Gallo et al. 1989, Lemma 2.4) (see also the references therein and in particular Eisner and Severance 1976). We also refer the reader to McCormick (1996) for further extensions of this approach. A proof based on stochastic arguments is found in Darbon and Sigelle (2006), while we present in Appendix C the elementary proof given in Chambolle (2005).

Quantized Total Variation Minimization Problem We will discuss in the next section how Problem (15) can be (efficiently) solved by successive minimizations of (16). It seems that efficiently solving the successive minimizations has been first proposed in the seminal work of Eisner and Severance (1976) in the context of augmenting-path maximum-flow algorithms. It was then developed, analyzed and improved by Gallo et al. (1989) for preflow-based algorithms. Successive improvements were also proposed by Hochbaum (2001), specifically for the application in view in this paper, that is, the minimization of (15). We also refer to the work of Kolmogorov et al. (2007) for detailed discussions about this approach. (The authors of the present note rediscovered the latter algorithm (Chambolle 2005; Darbon and Sigelle 2006), following quite different paths.)

Following Eisner and Severance (1976), and assuming that one can perform exact floating point operations, one could solve (16) for all values of z. We will follow a different approach and introduce the following quantized version of Problem (15):

$$\min\Big\{\lambda J(v) + \frac{1}{2}\|v-g\|^2 : v\in\mathbb{R}^N,\\ v_i \in \{l_0\dots,l_n\}\forall i=1,\dots,N\Big\} \tag{17}$$

where the real levels $(l_k)_{k=0}^n$ are given. That is, we minimize (15) only among functions that take values in a prescribed, finite set. Without loss of generality, we assume that $l_0 < l_1 < \cdots < l_n$, and for simplicity that for all $k = 1,\dots,n$, $l_k - l_{k-1} = \delta > 0$ (adaption to other cases is straightforward).

Our approach is therefore suboptimal. In fact, the problem which is solved (exactly) by (17) can also be interpreted as an approximate problem where the quadratic potential $|v_i-g_i|^2/2$ is replaced for each node i with a piecewise affine potential, taking the same values for $v_i\in\{l_0,\dots,l_n\}$. It is a very general approach, in the sense that if $\|v-g\|^2/2$ is replaced with an arbitrary convex (C^1, otherwise some (simple) adaption is required) potential $\sum_i \Psi_i(v_i)$, then (17) can be tackled in the same way with obvious modification (and this is, in general, optimal). However, for some "simple" potentials and in particular the one in (17), the method described in Eisner and Severance (1976), Gallo et al. (1989), Hochbaum (2001) computes the exact solution (of course, in practice, up to machine precision) and is therefore optimal. We will return to this later on, when discussing the practical implementation, see Sect. 4.3.2.

Concerning (17), the following result is true:

Proposition 3.5 *Let v be a solution of* (17), *and u be the solution of* (15). *Then for each $i = 1, \dots, N$, if $l_0 \le u_i \le l_n$, $|u_i - v_i| \le \delta/2$.*

In particular, if $l_0 \le m$ and $l_n \ge M$, $\max_i |u_i - v_i| \le \delta/2$. This means that the quantized problem (17) produces exactly a quantization of the solution of (15). We note that this approach leads to algorithms which solve our problem with an L^∞ a priori error bound. This is quite different from more standard (PDE-based) techniques (see for instance Chambolle 2005, Sect. 4) which typically will produce a solution up to some L^2 error. Again, the proof of this proposition is given in Appendix C.

In the next section, we describe well-known algorithms for solving (16) and how to use them to solve (15).

4 Parametric and Dyadic-Parametric Maximum Flow

4.1 Graph Representation of Binary Energies

It was first observed by Picard and Ratliff (1975) that binary Ising-like energies, that is, of the form

$$\sum_{i,j} \alpha_{i,j} |\theta_i - \theta_j| - \sum_i \beta_i \theta_i,$$

could be represented on a graph and minimized by standard optimization techniques, and more precisely using maximum flow algorithms. Kolmogorov and Zabih (2004) showed that submodularity is a necessary condition, while, up to sums of ternary submodular interactions, it is also a sufficient condition in order to be representable on a graph. Sufficient conditions for higher order interactions are given in Freedman and Drineas (2005). In general, it does not seem to be known whether any submodular J can be represented on a graph in the way proposed in Picard and Ratliff (1975), Kolmogorov and Zabih (2004). For instance, it is easy for the particular example (13), although it may involve much more than three variables, see Appendix B. Note that other efficient algorithms exist for minimizing submodular functions (Cunningham 1985; Grötschel et al. 1981; Iwata et al. 2000; Schrijver 2000).

Let us apply this to (16), in the simpler case where J has only pairwise interactions, hence:

$$J(u) = \sum_{i,j} \alpha_{i,j} (u_i - u_j)^+.$$

The construction we will describe has been presented in Boykov et al. (2001), Greig et al. (1989), Kolmogorov and Zabih (2004), Picard and Ratliff (1975).

We consider problem (16), for a given value of t. We build a graph as follows: we consider $\mathcal{E} = \{1, \dots, N\} \cup \{s\} \cup \{t\}$ where the two special nodes s and t are respectively called the "source" and the "sink". We consider then oriented edges (s,i) and (i,t), $i = 1, \dots, N$, and (i,j), $1 \le i,j \le N$, and to each edge we associate a capacity defined as follows:

$$\begin{cases} c(s,i) = (z - g_i)^-, & i = 1, \dots, N, \\ c(i,t) = (z - g_i)^+, & i = 1, \dots, N, \\ c(i,j) = \lambda \alpha_{i,j}, & 1 \le i,j \le N. \end{cases} \tag{18}$$

By convention, we consider there is no edge between two nodes if the capacity is zero. Let us denote by $\mathcal{E}$ the set of edges with nonzero capacity and by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ the resulting oriented graph.

We then define a "cut" in the graph as a partition of $\mathcal{E}$ into two sets $\mathcal{S}$ and $\mathcal{T}$, with $s \in \mathcal{S}$ and $t \in \mathcal{T}$. The cost of a cut is then defined as the total sum of the capacities of the edges that start on the source-side of the cut and land on the sink-side:

$$C(\mathcal{S}, \mathcal{T}) = \sum_{\substack{(\mu,\nu) \in \mathcal{E} \\ \mu \in \mathcal{S}, \nu \in \mathcal{T}}} c(\mu, \nu).$$

Then, if we let $\theta \in \{0,1\}^N$ be the characteristic function of $\mathcal{S} \cap \{1, \dots, N\}$, we clearly have

$$\begin{aligned} C(\mathcal{S}, \mathcal{T}) &= \sum_{i=1}^N (1 - \theta_i)(z - g_i)^- + \theta_i (z - g_i)^+ \\ &\quad + \sum_{i,j=1}^N \lambda \alpha_{i,j} (\theta_i - \theta_j)^+ \\ &= \lambda J(\theta) + \sum_{i=1}^N \theta_i (z - g_i) + \sum_{i=1}^N (z - g_i)^-. \end{aligned}$$

Hence, up to a constant, it is nothing else than the energy in (16).

So far, the problem has just been reformulated. The interesting part is that very efficient algorithms are available for finding a minimum cut, based on a duality result of Ahuja et al. (1993). The idea is to find the maximum flow in the graph, in the following sense: starting from s, we "push" a quantity along the oriented edges of graph, with the constraint that the flow along each edge (μ,ν) should remain between 0 and $c(\mu,\nu)$, and that each "interior" node i must get as much as it sends (while the source s only sends flow to the network, and the sink t only receives). It is clear that the total flow which can be such sent is bounded from above, less clear, but not hard to show, that this bound is given my a minimal-cost cut $(\mathcal{S},\mathcal{T})$. The duality theorem of Ford and Fulkerson expresses the fact that this bound is actually reached, and the partition $(\mathcal{S},\mathcal{T})$ is obtained by cutting along the saturated edges, where the flow is equal to the capacity while the possible reverse flow is zero. More precisely, we can find starting from s the first saturated edge along the graph, and cut there, or do the same starting from t and scanning the reverse graph, this will usually give the same solution except for a finite number of levels z. Several efficient algorithms are available to compute a maximum flow in polynomial time (Ahuja et al. 1993). Although the time complexity of the Boykov and Kolmogorov's maximum flow described in Boykov and Kolmogorov (2004) is not polynomial, this algorithm outperforms others in terms in time computations. We now describe how these techniques can be adapted to solve efficiently a series of problems, corresponding to varying levels $z = z_1,\dots,z_n$, with the global complexity of a single one. These approaches follow from the seminal work of Eisner and Severance (1976) and Gallo et al. (1989), with an improvement due to Hochbaum (2001).

4.2 Parametric Max-Flow Algorithm

The main idea of a parametric max-flow is to reuse the flow found for a given problem for the next one. It works for a series of problem where the capacities from the sink to the source are nondecreasing while those from the source to the sink are non-increasing and all other capacities remain unchanged. Gallo et al. (1989) show that under these assumptions the monotony of the solutions given by Lemma 3.4 hold. In terms of graph it means the set of nodes connected to the source is growing as the level z is decreasing. They take benefit from this property by modifying the preflow-push algorithm of Goldberg and Tarjan (1986) using the "residual" preflow obtained at the previous stage as a starting point for the next one. Using this strategy, they show that the total time complexity of solving these series of max flows is exactly the one for solving a single one.

Of course, the same idea can also be embedded in augmented path-based algorithms, such as the one of Boykov and Kolmogorov (2004). Let us describe quickly how it works. A convenient way to describe a flow f in a graph $\mathcal{G} = (\mathcal{V},\mathcal{E})$ is the notion of the residual network $\mathcal{G}_f = (\mathcal{V},\tilde{\mathcal{E}})$. It has the same set of nodes as $\mathcal{G}$, but the set of edges with positive capacity may be different. For each arc the flow sent along an edge is deduced from its capacity while it is added to the capacity of the opposite arc. More precisely, for all arcs (μ,ν) we have $\tilde{c}(\mu,\nu) = c(\mu,\nu) - f(\mu,\nu) + f(\nu,\mu)$ and $\tilde{c}(\nu,\mu) = c(\nu,\mu) - f(\nu,\mu) + f(\nu,\mu)$. After one run of an augmented path-based max-flow algorithm, the initial graph is usually replaced with a residual network whose saturated arcs (μ,ν) have been removed (i.e., their capacity has been set to zero, while $\tilde{c}(\nu,\mu) = c(\nu,\mu) + c(\mu,\nu)$ is maximal).

The implementation of the parametric algorithm is based on this representation. We start with a minimal level $z = z_1$ in (18) (assuming we want to solve our problem for $z_1, z_2 = z_1 + \delta,\dots,z_n = z_1 + (n-1)\delta)$, and compute a first residual network. Then, in this new network, we increase by δ all residual capacities $\tilde{c}(i,t)$, $i = 1,\dots,N$, and start again the augmented path algorithm. If i was in $\mathcal{T}$, it can not get any new flow from any node (since all path from s are still saturated at some point), hence nothing will happen there (and actually, the real implementation of the algorithm does not even increase the corresponding capacity $c(i,t)$). In particular, edges from i to some node of $\mathcal{S}$ (which need not be saturated) do not get any new flow, and the output would remain the same if these edges had been deleted before starting again the algorithm. This remark is crucial for the variant of this algorithm we will discuss in the next section. On the other hand, if i was in $\mathcal{S}$, then it gets connected to the sink t again and flow may pass through. This flow will saturate some edge closer to the source, so that i may either stay connected to the source or become connected to the sink after the next run. This shows again why as z increases, the corresponding set $\mathcal{S}$ decreases.

This procedure is iterated until the last level is reached.

For simple cases (Ahuja et al. 1993 for instance, when the complexity is deduced from the properties of a nondecreasing distance function) one can verify that the global complexity of the parametric max-flow algorithm is the one of a maximum flow plus $O(Nn)$. The latter corresponds to the number of operations required to update the capacities and retrieve the solution.

To our knowledge, the general case remains unknown. Considering this scheme applied with the algorithm of Boykov and Kolmogorov (2004), we do not know the complexity. However, we observe a much faster convergence, compared to a naive approach that consists in re-creating a new graph for each z_k (we take into account the monotonicity property by deleting from this new graph the nodes where the solution is already found to be below the level $z_{k-1} < z_k$, see Darbon and Sigelle 2006).

This approach provides a first, fast method for solving (17). See Sect. 4.4 for some experiments. Note that Kholi and Torr (2005) describes an efficient approach to recompute a maximum-flow that handles arbitrary changes in the graph. The next section describes a slightly faster approach which can even be modified to produce solutions with high precision (up to machine precision), see Sect. 4.3.2.

4.3 Dyadic-Parametric Max-Flow Algorithm

4.3.1 A Fast Algorithm for (17)

It was first observed by Hochbaum (2001) that this parametric approach could be improved according to the following observation: a pixel only needs to be involved in $O(\log_2(n))$ computations (by a dichotomy approach) instead of $O(n)$. this fact has also been noted and used in Chambolle (2005), Darbon and Sigelle (2006). For the sake of clarity we adopt a dyadic scheme to implement this dichotomic approach and we assume $n = 2^Q - 1$ for some $Q > 1$.

The algorithm works as follows. Instead of starting with z_1 we begin with $z_{(n+1)/2}$, and we compute the max-flow. We find a set $\mathcal{S}_1 \setminus \{s\}$ of pixels i with value $u_i \geq z_{(n+1)/2}$ and a complement $\mathcal{T}_1 \setminus \{t\}$ of pixels i with $u_i \leq z_{(n+1)/2}$. On the first one, we solve now for the level $z_{3(n+1)/2}$, while on the second one we solve for the level $z_{(n+1)/4}$. This can be done in many ways: in Chambolle (2005), a new (disconnected) graph with N nodes was built to implement the corresponding energy, while in Darbon and Sigelle (2006) a more clever (and faster) approach, separating the various connected components of these two sets, was implemented. It is more efficient, however, to try to "continue" the previous graph-cut, as described in Hochbaum (2001) in the framework of a preflow-push implementation. In an augmented path algorithm, we are left with a residual graph, such that no arc from $\mathcal{S}_1$ to $\mathcal{T}_1$ has positive capacity.

We then continue as follow: we first set to zero the capacities of the residual edges from $\mathcal{T}_1$ to $\mathcal{S}_1$ which means, we eliminate the corresponding edge, ending up with a totally disconnected graph. Then, for $i \in \mathcal{S}_1$ we increase by $\Delta = z_{3(n+1)/2} - z_{(n+1)/2} = z_{(n+1)/2} - z_{(n+1)/4}$ the capacity $c(i,t)$ while if $i \in \mathcal{T}_1$ we increase by Δ the capacity $c(s,i)$. We continue the augmented path algorithm, to find a new cut $(\mathcal{S}_2, \mathcal{T}_2)$. The discussion in the previous section shows that if $i \in \mathcal{S}_1 \cap \mathcal{S}_2$, then $u_i \geq z_{3(n+1)/2}$, if $i \in \mathcal{S}_1 \cap \mathcal{T}_2$, $z_{(n+1)/2} \leq u_i \leq z_{3(n+1)/2}$, if $i \in \mathcal{T}_1 \cap \mathcal{S}_2$, $z_{(n+1)/2} \geq u_i \geq z_{(n+1)/4}$ and if $i \in \mathcal{T}_1 \cap \mathcal{T}_2$, $u_i \leq z_{(n+1)/4}$.

After the qth step, we are left with a new cut $(\mathcal{S}_q, \mathcal{T}_q)$. Again we disconnect this partition, setting to 0 the capacities $c(i,j)$ for $i \in \mathcal{T}_q$ and $j \in \mathcal{S}_q$), replace Δ with $\Delta/2$ and update the capacities $c(i,t)$ and $c(s,i)$ as before: if $i \in \mathcal{S}_q$, $c(i,t)$ is increased by Δ, while if $i \in \mathcal{T}_q$, $c(s,i)$ is increased by Δ. We repeat this until $q = Q$: in the end, we have partitioned the nodes into sets where u_i is between two consecutive values of z_k.

Our modified version of the maximum flow code of Boykov and Kolmogorov (2004) that has been adapted for solving efficiently problem (17) is available through the authors' web site.

Again, Hochbaum shows that this procedure, implemented upon the preflow-push algorithm, has a complexity which is roughly the same as one of a max-flow computation, plus $O(NQ)$ (that is, $O(N\log_2 n)$), leading to a globally polynomial algorithm for (17). We do not know if this is still true for the variant we have implemented upon Boykov and Kolmogorov's algorithm, but it clearly outperforms the previous implementations presented in Chambolle (2005), Darbon and Sigelle (2006) in which new graphs were rebuilt at each step (see the next subsection).

4.3.2 Towards the Exact Solution of (15)

In fact, it is observed by Eisner and Severance (1976) and Hochbaum (2001) that a variant of the parametric algorithm can produce the exact solution of the problem. We now explain how the code can be modified (as well as the standard parametric approach, see Eisner and Severance 1976) in order to produce (still in polynomial time in its preflow-push version, Hochbaum 2001) the exact solution of (15), up to machine precision. The idea is to update the capacities, not with a constant factor Δ, but in a way to detect the "breakpoints", that is, the values of z for which the solution to (16) is multiple (or, equivalently, changes), which are nothing else as the values $\{u_i : i \in 1, \dots, N\}$ of the exact solution to (15). To be more specific, assume $\bar{z} > \bar{z}'$, and $\bar{\theta}$, $\bar{\theta}'$ are solutions to (16) with the values $\bar{z}$ and $\bar{z}'$ respectively. From Lemma 3.4, we know that $\bar{\theta} \leq \bar{\theta}'$. If $\bar{\theta}$ and $\bar{\theta}'$ are different, there must a breakpoint in $[\bar{z}', \bar{z}]$. Then, there are two closed sets $[z_1, z_2] \ni \bar{z}$, $[z_1', z_2'] \ni \bar{z}'$, with $z_1' < z_2' \leq z_1 < z_2$ such that for $z \in [z_1, z_2]$, $\bar{\theta}$ solves (16) while if $z \in [z_1', z_2']$, $\bar{\theta}'$ solves (16). In particular, the minimal energy in $[z_1', z_2']$ is given by

$$e'(z) = \left(\lambda J(\bar{\theta}') - \sum_{i=1}^{N} \bar{\theta}_i' g_i\right) + z\sharp\{i \in \{1, \dots, N\} : \bar{\theta}_i' = 1\}$$

while if $z \in [z_1, z_2]$, it is

$$e(z) = \left(\lambda J(\bar{\theta}) - \sum_{i=1}^{N} \bar{\theta}_i g_i\right) + z\sharp\{i \in \{1, \dots, N\} : \bar{\theta}_i = 1\}$$

which has a strictly lower slope, since $\bar{\theta} \leq \bar{\theta}'$ and they are different. Let $\hat{z} \in [\bar{z}', \bar{z}]$ be the value for which $e' = e$, and let us solve (16) for this new value: then, either the energy is strictly below $e(\hat{z}) = e'(\hat{z})$, meaning that the solution $\hat{\theta}$ is neither $\bar{\theta}$ nor $\bar{\theta}'$, and there are two breakpoints, one in $[\bar{z}', \hat{z})$ and the other in $(\hat{z}, \bar{z}]$: then we will divide again these intervals. Or, the new energy is equal to the common value $e(\hat{z}) = e'(\hat{z})$, which means that $\bar{\theta}$ and $\bar{\theta}'$ are respectively the minimal and maximal solution of (16) for $z = \hat{z}$ and $\hat{z}$ is a breakpoint.

In practice, it seems that implementing this dichotomic search will be expensive (although still polynomial), since one needs to compute the values of the energy (and more precisely, of $(\lambda J(\theta) - \sum_{i=1}^{N} \theta_i g_i)$ and $\sharp\{i \in \{1, \ldots, N\} : \theta_i = 1\}$), for each new minimizer θ of (16) which is computed. However, the residual graph makes it easier. Assume we start at a first stage with $\bar{z}' < \min_i g_i$ and $\bar{z} > \max_i g_i$. Then, $\bar{\theta}_i = 0$ while $\bar{\theta}'_i = 1$, for all i, and one easily checks that the new value $\hat{z}$ for which $e(\hat{z}) = e'(\hat{z})$ is simply the average $(\sum_{i=1}^{N} g_i)/N$. In the graph representation, it means that one updates the capacities $c(s,i)$ and $c(i,t)$, by adding a value Δ to either all $c(s,i)$ or all $c(i,t)$ in such a way that the new values satisfy $\sum_i c(s,i) = \sum_i c(i,t)$. One can easily show that this is the correct update to perform, also at the subsequent steps. We now describe two ways to implement this version.

A first variant of the dyadic algorithm described in Sect. 4.3.1 is as follows. Recall that each time a maximum-flow is computed, the solution is refined. This means that the level sets that are not yet within the given precision are divided into two subsets (one connected to the source, one connected to the sink), in which a new value will be computed. If one of these subsets is empty, this means that a breakpoint has been discovered (i.e., the value on the level set was the exact value of the solution). The corresponding level set should not be considered for further optimization and is thus removed from the graph.

For each level set that is actually divided into two non-empty new sets, the value in each new set is updated as follows: we average the values of the residual capacities from the source minus the residual capacities to the sink, and we add this average to the old value in the level set, while updating the residual capacities accordingly (as explained above). After this update, we can compute the largest residual capacity from the source or to the sink: if this is less that the given precision, then we know that the value of this level is correct up to the precision. Then, again, this set should not be considered for further optimization. It is thus removed from the graph.

The latter procedure can be further improved using ideas similar to the ones described in Darbon and Sigelle (2006). Note that the way the above process adjusts the levels of the next cuts does not take into account the geometry. Indeed, only the current gray-level values, and the associated level sets, are considered for the update. However, following Darbon and Sigelle (2006), one sees that once a maximum-flow is computed it separates the problem into smaller problems that involve the *connected components* of the level sets of the current solution. More precisely, after a cut, the solution restricted to a connected component can be computed independently from any other connected components. This means that the update of the residual capacities can be performed *independently* on each connected component of the level sets rather than on each level set. One can expect that less iterations are needed to reach a solution with the desired precision since the refinement is performed in a finer way.

4.4 Comparisons

We now compare these different approaches for solving the discrete ROF model (7). Two kinds of discrete total variations are considered: the first one is given by example (12), i.e., the image is defined on a regular lattice endowed with the 4-connectivity while the second one assumes the 8-connectivity. For the latter, the interactions terms involving the 4-nearest neighbors are weighted by 1 while diagonal interaction terms are weighted by $\frac{1}{\sqrt{2}}$.

Two parametric maximum-flows algorithm have been implemented. The first one relies on the push-relabel (PR) approach with the highest label strategy, a gap strategy and a global relabeling heuristics (that we implemented, see for instance Cormen et al. 2001 and Ahuja et al. 1993, p. 233 for more details), while the second is our adaptation of the maximum-flow implementation of Boykov-Kolmogorov (BK) (2004). We have considered the following versions of the parametric approach: the standard parametric one, the dyadic-parametric version (Sect. 4.3.1), the Eisner-Severance/Hochbaum one (referred to as ESH) and its connected component variation (referred to as ESH CC). Note that the ESH versions have only been implemented upon the BK maximum-flow algorithm, which experimentally appears to be much faster that the standard push-relabel algorithm, as already observed in Boykov and Kolmogorov (2004). We also compare with the previous approach of Darbon and Sigelle (2006). Time results for these seven different algorithms are given for an Intel Core2 Q9650 processor running at 3 GHz. Figure 1 depicts two images, *cows* (400×600 and 800×1200) and *girl* (256^2 and 512^2) used for our experiments. The original grey-level values are integer values ranging from 0 to 255. Minimizers with the 8-connectivity for the *girl* and *cows* images are respectively depicted in Fig. 2 and Fig. 3 for several regularization parameters used for our experiments.

Time results with different regularization parameters for 4- and 8-connectivity, and with a precision of 1 on the result, are respectively given in Table 1 and Table 2. Results show that the dyadic and ESH-based approaches clearly outperforms the pure parametric one by an order of magnitude. We note that the Push-Relabel-based version of the parametric approach is much more dependent on the regularization parameter λ than BK's. We also observe that the performance order depends on the content of the image and on the value of the regularization parameter. Indeed, for small regularization ($\lambda = 10$), PR performs better for *cows* but

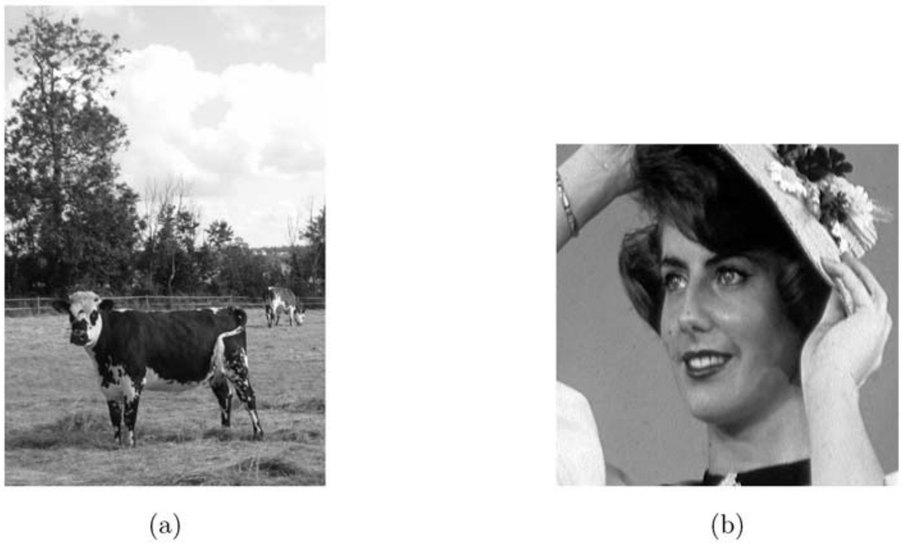
Fig. 1 Two original images: (**a**) cows, (**b**) girl

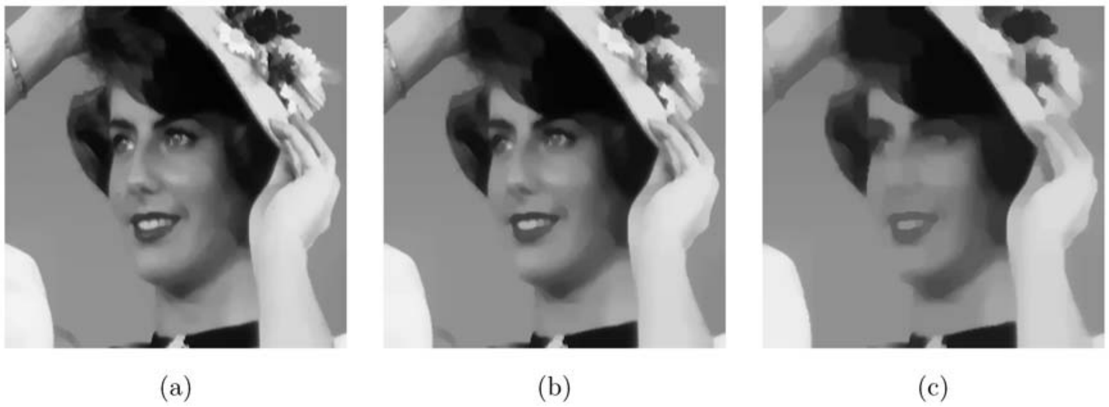
Fig. 2 Regularized *girl* (256^2) images: (**a**) $\lambda = 10$, (**b**) $\lambda = 20$, (**c**) $\lambda = 60$

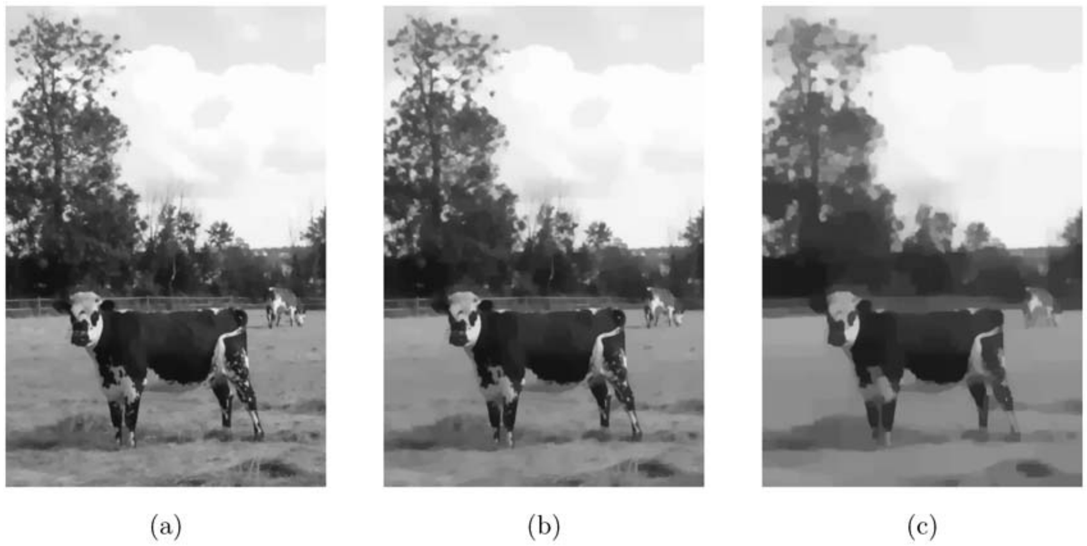
Fig. 3 Regularized *cows* (400×600) images: (**a**) $\lambda = 10$, (**b**) $\lambda = 20$, (**c**) $\lambda = 60$

Table 1 Time results for several regularization parameter λ using 4-connectivity and a precision of 1. Time results are in seconds

Images (size)	Approach	$\lambda = 10$	$\lambda = 20$	$\lambda = 60$
Cows (400×600)	Darbon-Sigelle	0.57	0.69	1.03
	Parametric PR	3.08	4.63	8.34
	Dyadic parametric PR	1.05	1.65	3.81
	Parametric BK	3.74	3.94	4.41
	Dyadic parametric BK	0.34	0.43	0.59
	ESH parametric BK	0.38	0.45	0.73
	ESH CC parametric BK	0.41	0.50	0.83
Cows (800×1200)	Darbon-Sigelle	2.4	2.91	4.55
	Parametric PR	14.81	24.65	111.35
	Dyadic parametric PR	5.58	9.36	23.21
	Parametric BK	16.25	17.03	19.02
	Dyadic parametric BK	1.42	1.81	2.75
	ESH parametric BK	1.76	2.04	3.00
	ESH CC parametric BK	1.85	2.31	3.61
Girl (256^2)	Darbon-Sigelle	0.16	0.19	0.27
	Parametric PR	0.95	1.06	1.3
	Dyadic parametric PR	0.34	0.54	0.86
	Parametric BK	0.83	0.88	1.03
	Dyadic parametric BK	0.08	0.10	0.14
	ESH parametric BK	0.08	0.10	0.16
	ESH CC parametric BK	0.09	0.13	0.18
Girl (512^2)	Darbon-Sigelle	0.63	0.79	1.25
	Parametric PR	8.65	20.01	21.26
	Dyadic parametric PR	2.07	3.58	5.03
	Parametric BK	4.09	4.27	4.87
	Dyadic parametric BK	0.41	0.51	0.81
	ESH parametric BK	0.44	0.56	0.94
	ESH CC parametric BK	0.50	0.65	1.11

worse on *girl*. This order is reversed for larger regularization. The overall performance of these two versions are comparable.

Considering the dyadic approaches, the order is stable over regularization parameters and image contents: the best one is the dyadic parametric BK algorithm followed by the Darbon-Sigelle approach while the dyadic parametric PR comes third. Finally note that our implementation of PR parametric maximum-flow with the highest label approach has not been fully tuned. We refer the reader to Goldfarb and Yin (2007) where an efficient implementation of a Push-Relabel approach is described for TV minimization.

At precision 1, we observe that the dyadic parametric approach relying on BK is also faster the ESH versions. This is probably explained by the fact the speed up that we get by computing more cleverly the capacities does not compensate the computational cost of these updates. We shall check that this is not true anymore for higher precisions.

Tables 3 and 4 present the time results with 4- and 8-connectivity for minimizers that have a precision of 2^{-8} using the three fastest approaches: namely the dyadic parametric version using BK, the ESH and the connected component-based ESH ones. These experiments show that, at this level of precision, the ESH parametric version running BK performs similarly as the simpler dyadic approach. We also note that the connected component-based ESH version turns out to be faster the standard ESH approach. This behavior is amplified when using a 2^{-16} precision (which, for these experiments, roughly corresponds to the machine precision) as can be seen in Tables 5 and 6. At this precision level, spending time to perform better updates clearly improves the performances, compared to the direct dyadic approach. This is further improved when using the connected component version.

Concerning the actual results, results with a precision of 1 for the dyadic or parametric approaches are excellent and cannot be distinguished visually from the solutions with very high precision (this is also true for the shape evolution examples shown later in the paper, although it is less obvious in this case). However, note that the possibility of computing

Table 2 Time results for several regularization parameter λ using 8-connectivity and a precision of 1. Time results are in seconds

Images (size)	Approach	$\lambda = 10$	$\lambda = 20$	$\lambda = 60$
Cows (400×600)	Darbon-Sigelle	1.06	1.34	2.18
	Parametric PR	5.24	11.89	14.10
	Dyadic parametric PR	2.68	4.45	9.16
	Parametric BK	7.99	8.45	9.67
	Dyadic parametric BK	0.71	0.98	1.43
	ESH parametric BK	0.72	1.08	1.69
	ESH CC parametric BK	0.77	1.19	1.93
Cows (800×1200)	Darbon-Sigelle	4.41	5.84	10.06
	Parametric PR	83.67	157.72	174.13
	Dyadic parametric PR	14.80	29.40	63.46
	Parametric BK	30.24	32.17	37.49
	Dyadic parametric BK	3.33	4.32	7.15
	ESH parametric BK	3.47	4.47	8.74
	ESH CC parametric BK	4.01	5.01	10.93
Girl (256^2)	Darbon-Sigelle	0.29	0.37	0.57
	Parametric PR	2.90	5.03	6.83
	Dyadic parametric PR	0.82	1.10	1.84
	Parametric BK	1.65	1.80	2.24
	Dyadic parametric BK	0.21	0.27	0.40
	ESH parametric BK	0.21	0.23	0.36
	ESH CC parametric BK	0.23	0.30	0.47
Girl (512^2)	Darbon-Sigelle	1.24	1.63	2.67
	Parametric PR	23.50	40.12	78.48
	Dyadic parametric PR	5.74	7.36	12.51
	Parametric BK	7.05	7.77	9.87
	Dyadic parametric BK	0.97	1.34	2.25
	ESH parametric BK	0.94	1.34	2.24
	ESH CC parametric BK	1.13	1.54	2.92

Table 3 Time results for several regularization parameter λ using 4-connectivity and a precision of 2^{-8}. Time results are in seconds

Images (size)	Approach	$\lambda = 10$	$\lambda = 20$	$\lambda = 60$
Cows (400×600)	Dyadic parametric BK	0.79	0.97	1.37
	ESH parametric BK	0.85	1.22	1.66
	ESH CC parametric BK	0.79	1.01	1.25
Cows (800×1200)	Dyadic parametric BK	3.89	4.86	7.74
	ESH parametric BK	4.77	9.64	15.60
	ESH CC parametric BK	3.79	9.21	15.26
Girl (256^2)	Dyadic parametric BK	0.18	0.23	0.34
	ESH parametric BK	0.16	0.20	0.31
	ESH CC parametric BK	0.14	0.17	0.28
Girl (512^2)	Dyadic parametric BK	0.96	1.26	1.95
	ESH parametric BK	0.91	1.24	2.09
	ESH CC parametric BK	0.71	1.06	1.83

solutions of the $TV - L^2$ problem with a very high precision may be of particular interest for solving some image restoration (deconvolution, reconstruction) problems with TV regularization through proximal algorithms (forward-backwards splitting, see Combettes and Wajs 2005 or Beck and Teboulle 2009 and Nesterov 2004, 2007).

Table 4 Time results for several regularization parameter λ using 8-connectivity and a precision of 2^{-8}. Time results are in seconds

Images (size)	Approach	$\lambda = 10$	$\lambda = 20$	$\lambda = 60$
Cows (400 × 600)	Dyadic parametric BK	1.62	1.98	2.98
	ESH parametric BK	1.64	2.37	3.34
	ESH CC parametric BK	1.33	1.90	3.21
Cows (800 × 1200)	Dyadic parametric BK	7.85	9.61	15.85
	ESH parametric BK	8.14	9.67	16.61
	ESH CC parametric BK	8.13	8.50	15.81
Girl (256^2)	Dyadic parametric BK	0.42	0.51	0.75
	ESH parametric BK	0.39	0.47	0.62
	ESH CC parametric BK	0.34	0.42	0.60
Girl (512^2)	Dyadic parametric BK	2.00	2.64	4.64
	ESH parametric BK	2.02	2.56	4.77
	ESH CC parametric BK	1.83	2.54	5.29

Table 5 Time results for several regularization parameter λ using 4-connectivity and a precision of 2^{-16}. Time results are in seconds

Images (size)	Approach	$\lambda = 10$	$\lambda = 20$	$\lambda = 60$
Cows (400 × 600)	Dyadic parametric BK	1.61	1.88	2.62
	ESH parametric BK	0.85	1.22	1.66
	ESH CC parametric BK	0.79	1.02	1.26
Cows (800 × 1200)	Dyadic parametric BK	10.10	12.43	17.84
	ESH parametric BK	5.17	10.49	16.80
	ESH CC parametric BK	3.79	9.38	16.09
Girl (256^2)	Dyadic parametric BK	0.48	10.54	0.84
	ESH parametric BK	0.17	0.22	0.33
	ESH CC parametric BK	0.15	0.18	0.28
Girl (512^2)	Dyadic parametric BK	1.69	2.24	3.65
	ESH parametric BK	0.93	1.29	2.18
	ESH CC parametric BK	0.74	1.07	1.90

Table 6 Time results for several regularization parameter λ using 8-connectivity and a precision of 2^{-16}. Time results are in seconds

Images (size)	Approach	$\lambda = 10$	$\lambda = 20$	$\lambda = 60$
Cows (400 × 600)	Dyadic parametric BK	2.92	3.91	5.88
	ESH parametric BK	1.79	2.51	3.47
	ESH CC parametric BK	1.40	2.44	3.26
Cows (800 × 1200)	Dyadic parametric BK	16.19	21.76	31.31
	ESH parametric BK	9.19	11.27	19.67
	ESH CC parametric BK	12.86	11.00	18.31
Girl (256^2)	Dyadic parametric BK	0.82	0.98	1.60
	ESH parametric BK	0.40	0.50	0.64
	ESH CC parametric BK	0.34	0.42	0.63
Girl (512^2)	Dyadic parametric BK	3.36	4.72	8.58
	ESH parametric BK	2.12	3.04	5.10
	ESH CC parametric BK	1.90	2.82	4.93

5 Surface Evolution Using Parametric Maximum Flows

We now show how all this can be used to approximate the mean curvature flow of an hypersurface (in some anisotropic geometry). Boykov et al. (2006) simply solve (1) by a simple graph cut (one run of the maxflow algorithm), so that the output is a discrete set. In this way, subpixel motion cannot be grasped (and in particular surfaces of very low curvature

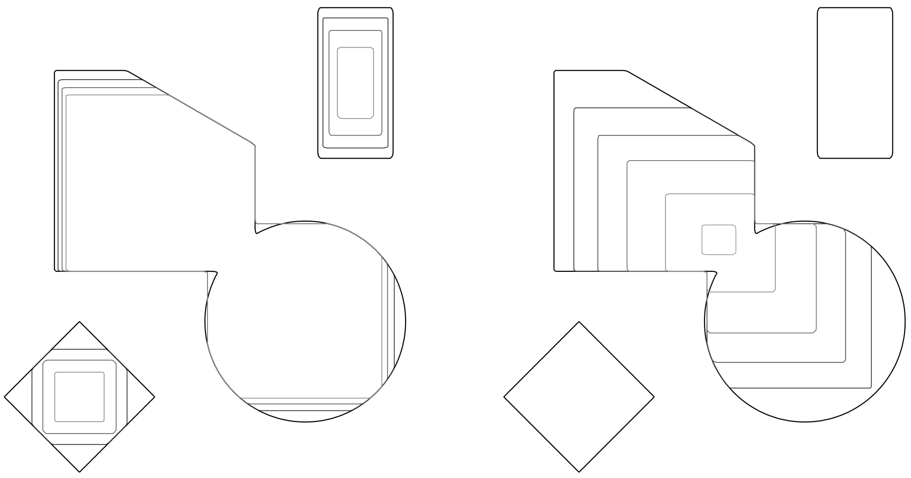
Fig. 4 Evolutions with a square anisotropy (*thick line*: original curve, *left*: iterations 10, 20, 30, *right*: iterations 50, 100, 150, 200, 228)

may remain stuck). We propose to use (15) as a discretization of the continuous problem (7) (for an anisotropy ϕ related to J) and then to estimate (by a linear interpolation) the position of the new hypersurface with a subpixel precision. In particular, it means that we estimate, for the next step, the new distance function to the zero level set of the function obtained at the previous step. Hence our algorithm is as follows: the initial surface is given as the zero level set of a function u^0, defined on our discrete grid. We fix a time-step $h > 0$, and assume for the sake of clarity that our discrete grid has a spatial resolution of 1. Given a discrete perimeter J, we alternatively, for $n \geq 0$;

- Compute the signed distance function d^n to the boundary of $\{u^n \leq 0\}$, by (for instance) a fast-marching algorithm (Sethian 1999; Tsitsiklis 1995).
- Solve the discrete version of (9):

$$\min_u J(u) + \frac{1}{2h}\|u - d^n\|^2 \tag{19}$$

by the dyadic-parametric max flow algorithm, and call u^{n+1} the solution.

Then, the surfaces $\Gamma^n = \{u^n = 0\}$ will be approximation of the anisotropic curvature flow with normal velocity κ^ϕ if J is an approximation, in some variational sense, of the perimeter Per_ϕ.

For instance, the two-dimensional function

$$J(u) = \sum_{i,j} |u_{i+1,j} - u_{i,j}| + |u_{i,j+1} - u_{i,j}| \tag{20}$$

is an approximation (as the grid step goes to zero), of the anisotropic perimeter

$$Per_\phi(E) = \int_{\partial E} |\nu_1|(x) + |\nu_2|(x)\, dx$$

corresponding to the anisotropy $\phi(\nu) = |\nu_1| + |\nu_2|$. Less anisotropic examples are easily built by considering more interactions (in other directions) in the definition of J (but this is not the only way).

The crystalline curvature motion in the sense of Bellettini and Paolini (1996) is obtained by computing in the first step a non-euclidean distance function, and more precisely, the signed distance function given by the polar of ϕ

$$d_E^\phi(x) = \inf_{y\in E} \phi^\circ(x-y) - \inf_{y\notin E} \phi^\circ(y-x) \tag{21}$$

with $\phi^\circ(\xi) = \sup_{\phi(\nu)\leq 1} \nu \cdot \xi$ (see Bellettini and Paolini 1996 for details). In the case of $\phi(\nu) = |\nu_1| + |\nu_2|$, one has $\phi^\circ(\nu) = \max\{|\nu_1|, |\nu_2|\}$ and the fast marching algorithm has to be modified accordingly to compute the appropriate distance (see Fig. 4 for an example of this crystalline flow).

Numerous improvements to the algorithms can be done: for instance, one may compute the distance function up to some given threshold (that might be adapted to the current shape), and solve the Total Variation problem only in a neighborhood of the surface (where $|d^n|$ is small).

Also, an additional normal force g (depending on the space and the time) is implemented by replacing (19) with

$$\min_u J(u) + \frac{1}{h}\|u - d^n + hg^n\|^2. \tag{22}$$

6 Numerical Examples

6.1 (Anisotropic) Curvature Flow

We have computed several $2D$ and $3D$ evolutions with this technique. The simplest cases correspond to the square (in $2D$) or cubic (in $3D$) anisotropy, that is, with $\phi(\nu) = \sum_i |\nu_i|$. Indeed, these cases are discretized on graphs with only nearest-neighbour interaction, for instance, in $2D$, the discrete energy is given by (20). The two examples illustrated by Figs. 4 and 5 follow the definition of the crystalline

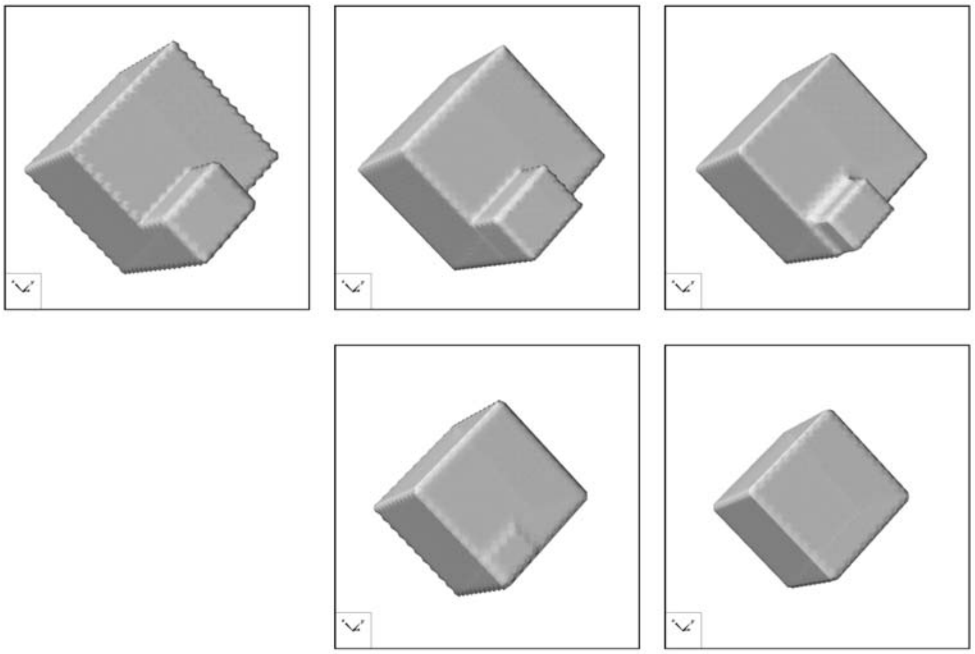
Fig. 5 3D evolutions with a cubic anisotropy: original shape and shape at times 1, 4, 7, 10

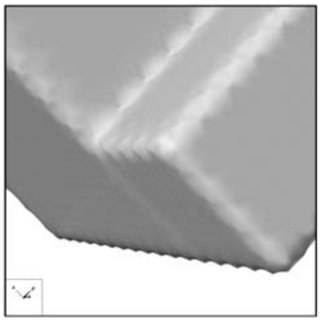
Fig. 6 Detail of the "facet breaking" at time 1

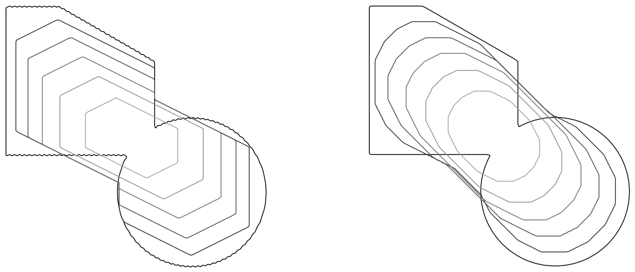
Fig. 7 Evolution with a hexagonal anisotropy (*left*), and nearly-isotropic curvature motion (*right*), both at times 0, 20, 40, 60, 80, 100

motion in Bellettini and Paolini (1996); in each case, the distance function is computed using the polar ϕ°: in practice, a fast-marching algorithm is implemented with a local solver implementing the discretization of $\phi(\nabla d^\phi) = 1$, taking into account the direction of the incoming characteristics just as in the isotropic case (Rouy and Tourin 1992).

In the 3D example of Fig. 5, one observes the celebrated "facet-breaking" phenomenon: a L-shaped facet of the original shape breaks into two rectangular facets which evolve at different speed, as predicted, and observed, in Bellettini et al. (1999), Paolini and Pasquarelli (2000), see Fig. 6.

It is possible, now, to compute "more isotropic" motions, or motion with more complex anisotropies (see Boykov and Kolmogorov 2003 for a general discussion on this topic). For instance, a hexagonal anisotropy can be implemented using nearest-neighbour interaction on a triangular lattice: see Fig. 7, left. Nearly-isotropic evolutions are computed using more complex interactions, for instance, involving next-nearest neighbours and even further neighbours: see Fig. 7, right. However, in this last case, one still sees that the evolution looks crystalline, with a shape presenting a small number of facets after some time.

6.2 Flows with Forcing Term

The mean curvature flow with constant volume is the simplest flow with a forcing term that can be implemented using this approach with little extra cost. In this case, a normal force is added (following (22)) which keeps the volume of the shape equal to the volume of the initial shape. In this particular case, this is simply done by thresholding the solution u of (19) not at the level $u = 0$, but at the level s such that $|\{u < s\}| = |\{d^n < 0\}| = V$, V being the initial volume. Such an evolution (with a square anisotropy) is depicted in Fig. 8.

We briefly show, without entering into the details, two other flows with forcing term computed with this technique. The first one is a basic 2D implementation of a crystal growth problem (Stefan's system of equations). We have followed the variational numerical method described in Almgren (1993), where it is implemented in a more standard way. At each step, we solve problem (22) where the external forcing field g^n depends on the temperature and is recomputed at each iteration. Results are shown in Fig. 9.

Our last example is an implementation of an active contour (snake) model, more precisely a "balloon", firstly introduced in Cohen (1991). Here, the curve follows the gradient flow of a modified perimeter which takes into account the intensities of the original image (and is cheaper when the curve goes through higher gradients). An internal (here, constant) inflating force is added in order to try to invade a whole region of interest. Figure 10 depicts an image of a heart in which we wish to segment a vein. We initialize the process with a little circle in the middle of the image. As is, this implementation is probably not very efficient with respect to more standard snake models, but this very simple approach gives good results.

Acknowledgements The authors would like to thank the referees for their careful reading of the paper, their comments and encouraging remarks, as well as for mentioning the references of Kolmogorov et al. (2007) and Eisner and Severance (1976). We were unaware of the first one, and had missed the particular relevance of the second.

Appendix A: Proof of Proposition 3.1

In this appendix, we prove shortly Proposition 3.1. Let us prove the lower-semicontinuity and the last assertion: first of all, J is lower semicontinuous because if $u^n \to u$, then for all $z \notin \{u_i, i = 1, \dots, N\}$, $\chi^{\{u^n>z\}} = \chi^{\{u>z\}}$ as soon as n is large enough. Hence, $J(\chi^{\{u^n>z\}}) \to J(\chi^{\{u>z\}})$ for a.e. z, so that (Fatou's lemma)

$$J(u) = \int_{-\infty}^{+\infty} J(\chi^{\{u>z\}})\,dz$$
$$\le \liminf_{n\to\infty} \int_{-\infty}^{+\infty} J(\chi^{\{u^n>z\}})\,dz = \liminf_{n\to\infty} J(u^n).$$

(If J is everywhere finite, then it follows from the convexity that it is locally Lipschitz-continuous.)

Let us now show the submodularity: assume first u and u' are binary. Then, $u_i + u'_i = u_i \vee u'_i + u_i \wedge u'_i = 0$ if $u_i \vee u'_i = 0$, 1 if $u_i \vee u'_i = 0$ but $u_i \wedge u'_i = 1$, and 2 if $u_i \wedge u'_i = 1$. Hence, by (10),

$$J(u+u') = \int_0^2 J\left(\chi^{\{u\vee u' + u\wedge u' \ge z\}}\right)dz$$
$$= J(u\vee u') + J(u\wedge u').$$

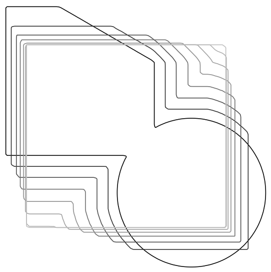
Fig. 8 Evolution of a volume preserving crystalline curvature motion

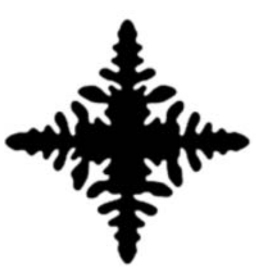
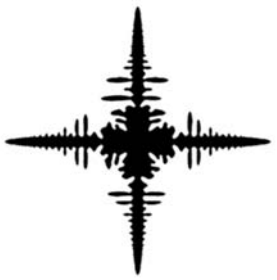
Fig. 9 Two examples of 2D crystal growths

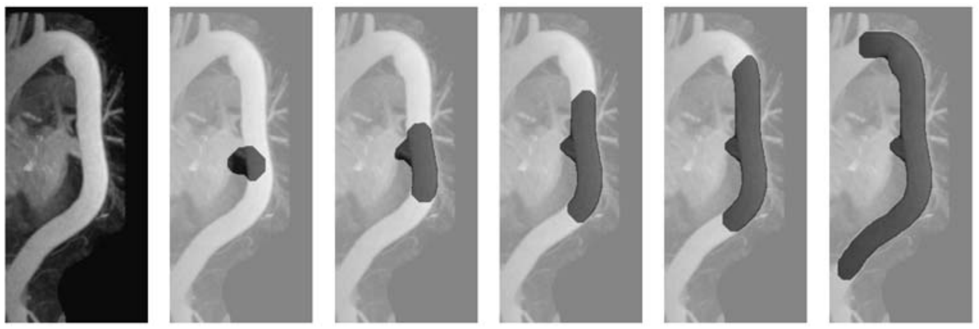
Fig. 10 An image of the heart (*left*) and the segmentation of a vein using an active "balloon" (*right*, final state)

On the other hand, since J is 1-homogeneous and convex,

$$J(u+u') = 2J\left(\frac{u+u'}{2}\right) \le 2\left(\frac{1}{2}J(u) + \frac{1}{2}J(u')\right)$$
$$= J(u) + J(u').$$

Hence (11) holds. If now u, u' are not binary, we still have

$$J(u \vee u') + J(u \wedge u')$$
$$= \int_{-\infty}^{+\infty} J\big(\chi^{\{u\vee u' \ge z\}}\big) + J\big(\chi^{\{u\wedge u' \ge z\}}\big)\, dz$$
$$= \int_{-\infty}^{+\infty} J\big(\chi^{\{u\ge s\}} \vee \chi^{\{u'\ge s\}}\big) + J\big(\chi^{\{u\ge z\}} \wedge \chi^{\{u'\ge z\}}\big)\, dz$$
$$\le \int_{-\infty}^{+\infty} J\big(\chi^{\{u\ge z\}}\big) + J\big(\chi^{\{u'\ge z\}}\big)\, dz = J(u) + J(v).$$

We now want to consider the converse assertion, that is, the convexity of the extension through (10) of a nonnegative submodular function. We consider $J : \{0,1\}^N \to [0,+\infty]$ a submodular function, i.e., such that (11) holds for any pair of binary vectors u, u'. We assume moreover that $J(0) = J(\mathbf{1}) = 0$, and we still denote by J its extension to $\mathbb{R}^N$ by the co-area formula (10). (We observe that thanks to $J(\mathbf{1}) = 0$, if $u \in \{0,1\}^N$ then (10) is already true.)

Then J is convex: hence it is a "discrete total variation". This extends quite easily to the continuous case. In the discrete case, however, this result is well known and usually proved in the framework of the linear optimization theory, using duality (Murota 2003; Lee 2004). We propose here a very elementary proof.

First of all, points 1–3 of the thesis of Proposition 3.1 are deduced only from (10) and therefore hold for J even if it were not convex.

Let us now show the convexity of J. Since it is 1-homogeneous it is equivalent to show that

$$J(u+v) \le J(u) + J(v) \tag{23}$$

for any $u, v \in \mathbb{R}^N$. We first consider nonnegative, integer-valued vectors u, v. We observe that if u is integer-valued, then J can be defined by the following inf-convolution formula:

$$J(u) = \min\left\{\sum_{l=1}^{n} J(\theta^l) : n \ge 0, \theta^l \in \{0,1\}^N, \sum_{l=1}^{n} \theta^l = u\right\}. \tag{24}$$

Indeed, denote by $H(u)$ the right-hand side of (24). Since

$$J(u) = \int_0^\infty J(\chi^{\{u>z\}})\, dz = \sum_{l=1}^{n} J(\chi^{\{u\ge l\}}),$$

where $n = \max_i u_i$, we have $H(u) \le J(u)$. The reverse inequality will hold if we show that the minimum in (24) is reached precisely for $n = \max_i u_i$ and $\theta^l = \chi^{\{u\ge l\}}$ (and, of course, any permutation of these), or, equivalently, if we show that it is reached for a monotone sequence of binary vectors θ^l.

This follows from the submodularity of J. If J is strictly submodular (that is, if the inequality in (11) is strict whenever the vectors are not ordered), then it is obvious: indeed, if the minimum in (24) is reached for $(\theta^l)_{l=1}^n$ and there are l, l' such that $\theta^l \not\le \theta^{l'}$ nor $\theta^l \not\ge \theta^{l'}$, then, replacing θ^l with $\theta^l \wedge \theta^{l'}$ and $\theta^{l'}$ with $\theta^l \vee \theta^{l'}$ we see that we strictly decrease the value of the minimum, a contradiction. Hence the minimum is reached for $\theta^l = \chi^{\{u\ge l\}}$, $l = 1,\dots,n = \max_i u_i$.

If J is not strictly submodular, we choose a strictly submodular function $\hat{J}$ (for instance, $\hat{J}(\theta) = g(\sum_{i=1}^N \theta_i)$, where g is a strictly concave function with $g(0) = g(N) = 0$), and for $\varepsilon > 0$ small we let $J_\varepsilon = J + \varepsilon \hat{J}$. Then, the minimal value in (24) for J_ε will be attained for $\theta^l = \chi^{\{u\ge l\}}$, $l = 1,\dots,n = \max_i u_i$. Passing to the limit, we still get that $H(u) = J(u)$ so that (24) is true.

Let us now show (23) for a general pair of vectors u, v. We may obviously assume that $J(u) < +\infty$ and $J(v) < +\infty$. In particular (since u and v take at most N value) we have $J(\chi^{\{u>z\}}) < +\infty$ and $J(\chi^{\{v>z\}}) < +\infty$ for any $z \in \mathbb{R}$. Let $m < \min\{u_i, v_i, i = 1,\dots,N\}$ and $M > \max\{u_i, v_i, i = 1,\dots,N\}$. We have $u = m\mathbf{1} + \int_m^M \chi^{\{u>z\}}\, dz$ and the same holds for v. Now, for $\varepsilon > 0$ small, we let

$$u^\varepsilon = m\mathbf{1} + \varepsilon \sum_{\substack{k\in\mathbb{Z}\\ m\le k\varepsilon\le M}} \chi^{\{u>k\varepsilon\}} \quad \text{and}$$
$$v^\varepsilon = m\mathbf{1} + \varepsilon \sum_{\substack{k\in\mathbb{Z}\\ m\le k\varepsilon\le M}} \chi^{\{v>k\varepsilon\}},$$

clearly, $u^\varepsilon \to u$ and $v^\varepsilon \to v$ as $\varepsilon \to 0$, and $J(u^\varepsilon) \to J(u)$, $J(v^\varepsilon) \to J(v)$. Now, letting $u^\varepsilon_\# = (u^\varepsilon - m\mathbf{1})/\varepsilon$ and $v^\varepsilon_\# = (v^\varepsilon - m\mathbf{1})/\varepsilon$, we have two non-negative integer-valued vectors to which we can apply (23), and we find

$$J(u^\varepsilon + v^\varepsilon) = \varepsilon J(u^\varepsilon_\# + v^\varepsilon_\#) \le \varepsilon(J(u^\varepsilon_\#) + J(v^\varepsilon_\#))$$
$$= J(u^\varepsilon) + J(v^\varepsilon).$$

Since the right-hand side converges to $J(u) + J(v)$ as $\varepsilon \to 0$, and J is l.s.c., we deduce (23). Hence J is convex.

Remark A.1 By standard convex analysis (see e.g. Ekeland and Témam 1999; Rockafellar 1997), we deduce that

$$J(u) = \sup_{q\in K} q\cdot u$$

where

$$K = \left\{ q \in \mathbb{R}^N : \sum_{i=1}^{N} q_i \theta_i \le J(\theta),\ \forall\, \theta \in \{0,1\}^N \right\} = \partial J(0),$$

the subgradient of J at 0. Then, it is standard that for any u, $\partial J(u) = \{q \in K : q \cdot u = J(u)\}$ and using (10) one shows easily that $q \in \partial J(u) \Rightarrow q \in \partial J(\chi^{\{u>s\}})$ for any $s \in \mathbb{R}$ (point 4 of Proposition 3.1).

Appendix B: Representation of Submodular Functions on Graphs

Following Kolmogorov and Zabih (2002, 2004), we say that the (necessarily submodular) $J(\theta_1, \dots, \theta_N)$ can be represented on a graph if there exists M additional nodes $i \in \{N+1, \dots, N+M\}$ and weights $\alpha_{i,j} \ge 0$, $\beta_i \in \mathbb{R}$ $(i, j \in \{1, \dots, N+M\})$ such that for any $\theta \in \{0,1\}^N$,

$$J(\theta_1, \dots, \theta_N) = \min_{(\theta_{N+1},\dots,\theta_{N+M}) \in \{0,1\}^M} \sum_{i,j=1}^{N+M} \alpha_{i,j}(\theta_i - \theta_j)^+ + \sum_{i=1}^{N+M} \beta_i \theta_i. \tag{25}$$

The energy in the right-hand side of (25) is clearly representable on a graph, following the standard construction in Sect. 4.1: hence (25) shows that J can be represented on a graph involving M additional nodes. Of course, this is really interesting only if M remains small, at most of the order of N. In Kolmogorov and Zabih (2002, 2004), it is shown that this is possible if $N \le 3$ (and, of course, for total energies that are the sum of representable energies), at the cost of adding $M = 2$ additional nodes. See also Freedman and Drineas (2005), Boros and Hammer (2002).

Note, however, that it is not difficult to build many other examples, involving more than 3 variables, which still enter this category. For instance, the energy in (13) is a sum of terms of the following type:

$$J_0(\theta_1, \dots, \theta_N) = \max\{\theta_i, i = 1, \dots, N\} - \min\{\theta_i, i = 1, \dots, N\}.$$

Such energies are representable: it is enough to add two additional nodes, corresponding to two additional variables $\overline{w}$ and $\underline{w}$, and observe that

$$J_0(\theta_1, \dots, \theta_N) = \min_{\overline{w}, \underline{w} \in \{0,1\}} \hat{J}_0(\theta_1, \dots, \theta_N, \overline{w}, \underline{w})$$

where

$$\hat{J}_0(\theta_1, \dots, \theta_N, \overline{w}, \underline{w}) = (\overline{w} - \underline{w})^+ + \sum_{i=1}^{N} \left((\theta_i - \overline{w})^+ + (\underline{w} - \theta_i)^+ \right).$$

It is clear that if θ is a constant vector, then taking $\overline{w} = \underline{w} = \theta_i$ give the value 0, while if θ is not constant, then the only way to make both terms in the sum less than 1 is by letting $\overline{w} = 1$ and $\underline{w} = 0$, but then the first term is 1.

Other examples are easily built, for instance if g is a concave function with $g(0) = g(N) = 0$, then

$$J_1(\theta_1, \dots, \theta_N) = g\left(\sum_{i=1}^{N} \theta_i \right)$$

is also representable.

Appendix C: Proofs of Propositions 3.3 and 3.5

We give in this appendix short proofs of Propositions 3.3 and 3.5. As mentioned before, the first relies on the comparison Lemma 3.4.

Proof of Lemma 3.4 We have

$$\lambda J(\theta) + \sum_{i=1}^{N} \theta_i (z - g_i) \le \lambda J(\theta \wedge \theta') + \sum_{i=1}^{N} (\theta_i \wedge \theta'_i)(z - g_i)$$

and

$$\lambda J(\theta') + \sum_{i=1}^{N} \theta'_i (z' - g_i) \le \lambda J(\theta \vee \theta') + \sum_{i=1}^{N} (\theta_i \vee \theta'_i)(z' - g_i).$$

Summing both inequality and using the submodularity of J, we end up with

$$\sum_{i=1}^{N} \theta_i (z - g_i) + \theta'_i (z' - g_i) \le \sum_{i=1}^{N} (\theta_i \wedge \theta'_i)(z - g_i) + (\theta_i \vee \theta'_i)(z' - g_i).$$

This is nothing else than

$$z \sum_{i=1}^{N} \theta_i - \theta_i \wedge \theta'_i \le z' \sum_{i=1}^{N} \theta_i \vee \theta'_i - \theta'_i,$$

but since $\theta_i - \theta_i \wedge \theta'_i = \theta_i \vee \theta'_i - \theta'_i = (\theta_i - \theta'_i)^+$, we find that if $z > z'$, $(\theta_i - \theta'_i)^+ = 0$ for all $i = 1, \dots, N$, that is, $\theta \le \theta'$. □

Proof of Proposition 3.3 We easily derive Proposition 3.3: indeed, if θ^z solve (16) for all values of z, and if we define $u \in \mathbb{R}^N$ by

$$u_i = \sup\{z : \theta^z_i = 1\}$$

then clearly $\chi^{\{u>z\}} \le \theta^z \le \chi^{\{u\ge z\}}$ for all z, as a consequence of Lemma 2.2. Also, $m = \min_j g_j \le u_i \le \max_j g_j = M$ for all i (since $\mathbf{1}$ is the unique solution of (16) if $z \le m$, while 0 is the solution if $z \ge M$). Hence, if $v \in \mathbb{R}^N$ and $m' \le m \wedge (\min_i v_i)$, we have (using (10) and the minimality of each θ^z)

$$\begin{aligned}\lambda J(u) &+ \frac{1}{2}\|u-g\|^2 \\ &= \int_{m'}^{+\infty} \lambda J(\theta^z) + \sum_{i=1}^N \theta^z_i (z-g_i)\,dz + \sum_{i=1}^N \frac{(m'-g_i)^2}{2} \\ &\le \int_{m'}^{+\infty} \lambda J(\chi^{\{v\ge z\}}) + \sum_{i=1}^N \chi_i^{\{v\ge z\}}(z-g_i)\,dz \\ &\quad + \sum_{i=1}^N \frac{(m'-g_i)^2}{2} = \lambda J(v) + \frac{1}{2}\|v-g\|^2\end{aligned}$$

which shows our claim. □

We give now the proof that the quantized *ROF* problem actually produces a solution which is exact (in the sup-norm), up to the quantification.

Proof of Proposition 3.5 In fact, For an admissible v in (17), we can write

$$v = l_0 + \sum_{k=1}^n (l_k - l_{k-1})\theta^k = l_0 + \delta \sum_{k=1}^n \theta^k$$

where for each $k \ge 1$, θ^k is the binary vector defined by $\theta^k_i = 1$ iff $v_i \ge l_k$. Then, the fact $\theta^k \le \theta^{k-1}$ for any $k \ge 2$, and the co-area formula (10), yield $J(z) = \sum_{k=1}^n \delta J(\theta^k)$. On the other hand,

$$\|g-v\|^2 = \sum_{i=1}^N (g_i - l_0)^2 + 2\delta \sum_{k=1}^n \sum_{i=1}^N \left(\frac{l_k + l_{k-1}}{2} - g_i\right)\theta^k_i,$$

hence, up to a constant, problem (17) is the same as

$$\min_{\theta^k} \sum_{k=1}^n \left(\lambda J(\theta^k) + \sum_{i=1}^N \left(\frac{l_k + l_{k-1}}{2} - g_i\right)\theta^k_i\right),$$

where the min is taken on all binary fields $(\theta^k)_{k=1}^n$, with the constraint that $\theta^k \le \theta^{k-1}$ for any $k = 2, \dots, n$. Each term in the sum is the energy that appears in problem (16), for $z = z_k = (l_k + l_{k-1})/2$. Now, by Lemma 3.4, if for each $k = 1, \dots, n$, θ^k is a minimizer of the corresponding energy, then, $z_k > z_{k-1}$ yields $\theta^k \le \theta^{k-1}$: hence the minimum problem above is in fact *unconstrained*. In particular, by Proposition 3.3 each θ^k is the between the characteristic functions of $\{u > z_k\}$ and $\{u \ge z_k\}$. This shows that Proposition 3.5 is true. □

References

Ahuja, R. K., Magnanti, T. L., & Orlin, J. B. (1993). *Network flows. Theory, algorithms, and applications*. Englewood Cliffs: Prentice-Hall.

Allard, W. K. (2007). Total variation regularization for image denoising, I. Geometric theory. *SIAM Journal on Mathematical Analysis*, *39*(4), 1150–1190.

Almgren, R. (1993). Variational algorithms and pattern formation in dendritic solidification. *Journal of Computational Physics*, *106*(2), 337–354.

Almgren, F., Taylor, J. E., & Wang, L.-H. (1993). Curvature-driven flows: a variational approach. *SIAM Journal on Control and Optimization*, *31*(2), 387–438.

Alter, F., Caselles, V., & Chambolle, A. (2005). A characterization of convex calibrable sets in $\mathbb{R}^N$. *Mathematische Annalen*, *332*(2), 329–366.

Babenko, M. A., Derryberry, J., Goldberg, A. V., Tarjan, R. E., & Zhou, Y. (2007). Experimental evaluation of parametric max-flow algorithms. In *Lecture notes in computer science* (Vol. 4252, pp. 256–269). Berlin: Springer.

Beck, A., & Teboulle, M. (2009). A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, *2*(1), 183–202.

Bellettini, G., & Paolini, M. (1996). Anisotropic motion by mean curvature in the context of Finsler geometry. *Hokkaido Mathematical Journal*, *25*(3), 537–566.

Bellettini, G., Novaga, M., & Paolini, M. (1999). Facet-breaking for three-dimensional crystals evolving by mean curvature. *Interfaces and Free Boundaries*, *1*(1), 39–55.

Bellettini, G., Caselles, V., Chambolle, A., & Novaga, M. (2006). Crystalline mean curvature flow of convex sets. *Archive for Rational Mechanics and Analysis*, *179*(1), 109–152.

Boros, E., & Hammer, P. L. (2002). Pseudo-boolean optimization. *Discrete Applied Mathematics*, *123*(1–3), 155–225.

Bouchitté, G. (1998). Recent convexity arguments in the calculus of variations. In *Lecture notes from the 3rd int. summer school on the calculus of variations*, Pisa.

Boykov, Y., & Kolmogorov, V. (2003). Computing geodesics and minimal surfaces via graph cuts. In *International conference on computer vision*, pp. 26–33.

Boykov, Y., & Kolmogorov, V. (2004). An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*(9), 1124–1137.

Boykov, Y., Veksler, O., & Zabih, R. (2001). Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *23*(11), 1222–1239.

Boykov, Y., Kolmogorov, V., Cremers, D., & Delong, A. (2006). An integral solution to surface evolution PDEs via Geo-Cuts. In A. Leonardis, H. Bischof, & A. Pinz (Eds.), *Lecture notes in computer science: Vol. 3953. European conference on computer vision (ECCV)* (pp. 409–422). Graz, Austria, May 2006. Berlin: Springer.

Caselles, V., & Chambolle, A. (2006). Anisotropic curvature-driven flow of convex sets. *Nonlinear Analysis*, *65*(8), 1547–1577.

Caselles, V., Chambolle, A., & Novaga, M. (2007). The discontinuity set of solutions of the TV denoising problem and some extensions. *Multiscale Modeling and Simulation*, *6*(3), 879–894.

Chambolle, A. (2004). An algorithm for mean curvature motion. *Interfaces and Free Boundaries*, *6*(2), 195–218.

Chambolle, A. (2005). Total variation minimization and a class of binary MRF models. In *Lecture notes in computer science*. *Energy minimization methods in computer vision and pattern recognition* (pp. 136–152). Berlin: Springer.

Chambolle, A., & Novaga, M. (2007). Approximation of the anisotropic mean curvature flow. *Mathematical Models and Methods in the Applied Sciences*, *17*(6), 833–844.

Chambolle, A., & Novaga, M. (2008). Implicit time discretization of the mean curvature flow with a discontinuous forcing term. *Interfaces and Free Boundaries*, *10*(3), 283–300.

Chan, T. F., & Esedoḡlu, S. (2005). Aspects of total variation regularized L^1 function approximation. *SIAM Journal on Applied Mathematics*, *65*(5), 1817–1837 (electronic).

Cohen, L. D. (1991). On active contour models and balloons. *CVGIP: Image Understanding*, *53*(2), 211–218.

Combettes, P. L., & Wajs, V. R. (2005). Signal recovery by proximal forward-backward splitting. *Multiscale Modeling and Simulation*, *4*(4), 1168–1200 (electronic).

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2001). *Introduction to algorithms*. Cambridge: MIT Press.

Cunningham, W. H. (1985). On submodular function minimization. *Combinatoria*, *5*, 185–192.

Darbon, J. (2005) Total variation minimization with L^1 data fidelity as a contrast invariant filter. In *Proceedings of the 4th international symposium on image and signal processing and analysis (ISPA 2005)*, Zagreb, Croatia, September 2005.

Darbon, J., & Sigelle, M. (2006). Image restoration with discrete constrained total variation part i: fast and exact optimization. *Journal of Mathematical Imaging and Vision*, *26*(3), 261–276.

Eisner, M. J., & Severance, D. G. (1976). Mathematical techniques for efficient record segmentation in large shared databases. *Journal of the ACM*, *23*(4), 619–635.

Ekeland, I., & Témam, R. (1999). *Classics in applied mathematics: Vol. 28. Convex analysis and variational problems*. Philadelphia: SIAM. (English ed., translated from the French).

Federer, H. (1969). *Geometric measure theory*. New York: Springer.

Freedman, D., & Drineas, P. (2005). Energy minimization via graph cuts: settling what is possible. In *IEEE computer society conference on computer vision and pattern recognition (CVPR)* (pp. 939–946).

Gallo, G., Grigoriadis, M. D., & Tarjan, R. E. (1989). A fast parametric maximum flow algorithm and applications. *SIAM Journal on Computing*, *18*(1), 30–55.

Giusti, E. (1984). *Monographs in mathematics: Vol. 80. Minimal surfaces and functions of bounded variation*. Basel: Birkhäuser.

Goldberg, A. V., & Tarjan, R. E. (1986). A new approach to the maximum flow problem. In *STOC'86: proceedings of the eighteenth annual ACM symposium on theory of computing* (pp. 136–146). New York: ACM Press.

Goldfarb, D., & Yin, Y. (2007). *Parametric maximum flow algorithms for fast total variation minimization*. Technical report, Rice University (2007).

Greig, D. M., Porteous, B. T., & Seheult, A. H. (1989). Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society Series B*, *51*, 271–279.

Grötschel, M., Lovász, L., & Schrijver, A. (1981). The ellipsoid method and its consequences in combinatorial optimization. *Combinatoria*, *1*, 169–197.

Hochbaum, D. S. (2001). An efficient algorithm for image segmentation, Markov random fields and related problems. *Journal of the ACM*, *48*(4), 686–701 (electronic).

Hochbaum, D. S. (2005). Complexity and algorithms for convex network optimization and other nonlinear problems. *A Quarterly Journal of Operations Research*, *3*(3), 171–216.

Iwata, S., Fleischer, L., & Fujishige, S. (2000). A combinatorial, strongly polynomial-time algorithm for minimizing submodular functions. In *STOC'00: Proceedings of the thirty-second annual ACM symposium on theory of computing* (pp. 97–106). New York: ACM.

Juan, O., & Boykov, Y. (2006). Active graph cuts. In *Proceedings of the IEEE Computer Society conference on computer vision and pattern recognition (CVPR)*, (pp. 1023–1029).

Kholi, P., & Torr, P. (2005). Efficient solving dynamic Markov random fields using graph cuts. In *Proceedings of the 10th international conference on computer vision* (pp. 922–929).

Kolmogorov, V., & Zabih, R. (2002). What energy functions can be minimized via graph cuts? In *European conference on computer vision* (Vol. 3, pp. 65–81), May 2002.

Kolmogorov, V., & Zabih, R. (2004). What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *2*(26), 147–159.

Kolmogorov, V., Boykov, Y., & Rother, C. (2007). Applications of parametric maxflow in computer vision. In *Proceedings of the IEEE 11th international conference on computer vision (ICCV 2007)* (pp. 1–8).

Lee, J. (2004). *A first course in combinatorial optimization*. Cambridge: Cambridge University Press.

Lovász, L. (1983). Submodular functions and convexity. In *Mathematical programming: the state of the art* (pp. 235–257). Bonn, 1982. Berlin: Springer.

Luckhaus, S., & Sturzenhecker, T. (1995). Implicit time discretization for the mean curvature flow equation. *Calculus of Variations and Partial Differential Equations*, *3*(2), 253–271.

McCormick, S. T. (1996) Fast algorithms for parametric scheduling come from extensions to parametric maximum flow. In *Proceedings of the twenty-eighth annual ACM symposium on theory of computing* (pp. 319–328).

Murota, K. (2003). *SIAM monographs on discrete mathematics and applications*. *Discrete convex analysis*. Philadelphia: SIAM.

Nesterov, Y. (2004). *Introductory lectures on convex optimization: a basic course*. Dordrecht: Kluwer.

Nesterov, Y. (2007) Gradient methods for minimizing composite objective function. Technical report, CORE discussion paper 2007/76, Catholic University of Louvain (2007).

Paolini, M., & Pasquarelli, F. (2000). Numerical simulation of crystalline curvature flow in 3D by interface diffusion. In *GAKUTO international series. Mathematical sciences and applications: Vol. 14. Free boundary problems: theory and applications, II* (pp. 376–389). Chiba, 1999. Tokyo: Gakkōtosho.

Picard, J. C., & Ratliff, H. D. (1975). Minimum cuts and related problems. *Networks*, *5*(4), 357–370.

Rockafellar, R. T. (1997). *Princeton landmarks in mathematics*. *Convex analysis*. Princeton: Princeton University Press (Reprint of the 1970 original Princeton paperbacks).

Rouy, E., & Tourin, A. (1992). A viscosity solutions approach to shape-from-shading. *SIAM Journal on Numerical Analysis*, *29*(3), 867–884.

Schrijver, A. (2000). A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *Journal of Combinatorial Theory (B)*, *80*, 436–355.

Sethian, J. A. (1999). Fast marching methods. *SIAM Review*, *41*(2), 199–235 (electronic).

Tsitsiklis, J. N. (1995). Efficient algorithms for globally optimal trajectories. *IEEE Transactions on Automatic Control*, *40*(9), 1528–1538.