ECOLE POLYTECHNIQUE

CENTRE DE MATHÉMATIQUES APPLIQUÉES
UMR CNRS 7641

On Total Variation Minimization and Surface Evolution using Parametric Maximum Flows

Antonin Chambolle and Jérôme Darbon

R.I. 632   *April 2008*

On Total Variation Minimization and Surface Evolution using Parametric Maximum Flows

Antonin Chambolle* and Jérôme Darbon†

Abstract

In a recent paper [13], Y. Boykov et al. propose an approach for computing curve and surface evolution using a variational approach and the *geo-cuts* method of Boykov and Kolmogorov [11]. We recall in this paper how this is related to well-known approaches for mean curvature motion, introduced by F. Almgren et al. [3] and S. Luckhaus and T. Sturzenhecker [44], 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

In [13], Y. Boykov, V. Kolmogorov, D. Cremers and A. Delong 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 [1]. 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 [12] 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

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

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

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

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, Taylor and Wang [3] and simultaneously by Luckhaus and Sturzenhecker [44], 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 $E \subset \mathbb{R}^N$

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

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

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

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 [3] 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 [3, 8] for a complete definition). Convergence results for generalized evolutions are found in [17, 20].

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

$$h\kappa_F^\phi(x) \ + \ d_E(x) \ = \ 0 \tag{3}$$

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

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

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 [17]) 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 [20].

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 equation (3) turns into

$$d_E(x) \ = \ -h\kappa_F^\phi(x) \ + \ hg(x, t))$$

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 [6, 15, 17, 19].

In this paper, we provide a framework for computing such evolutions by maxflow/mincut algorithms. The idea in [13] 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 [30, 36], to solve efficiently the discrete ROF problem; (ii) solve a discretized (ROF) problem to derive with a good (subpixel) precision an approximation of the set F which minimizes (2). In her seminal work [36], Hochbaum proposes an approach to solve the ROF model using parametric maximum-flow. This approach has been considered in [33] and we also refer the reader to [38, 39] 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 Section 3, we introduce our discrete setting, "discrete total variation" functionals, and basically state the same results as in Section 2 in this new setting. These properties lead to an efficient algorithm for the ROF problem, which we describe in Section 4. It is essentially a variant of the parametric max-flow algorithm [30] and has been first proposed by D. Hochbaum [36]. 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 Section 5 and show numerical examples in Section 6. Various appendices complete the paper. A modified version of the Boykov and Kolmogorov's maximum flow code [12], that implements the parametric approach to solves the ROF problem, is available through the authors' personal web page.

2 Minimal surface problems and total variation minimization

It is well known that level sets of function which minimize the total variation are themselves minimizing surfaces, and this fact is a main tool for the study of these surfaces and their regularity [28, 31]. 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 used, though it relies on the same celebrated "co-area formula" [28, 31]:

$$|Du|(\Omega) \;=\; \int_{-\infty}^{+\infty} Per\left(\{u > z\}, \Omega\right) dz \tag{5}$$

Let us just state the main equivalence:

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

$$\min_{u \in BV(\Omega)} \lambda \int_{\Omega} |Du| \;+\; \frac{1}{2} \int_{\Omega} |u(x) - g(x)|^2 \, dx \,. \tag{6}$$

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

$$\min_{E \subseteq \Omega} \lambda Per\left(E, \Omega\right) \;+\; \int_E z - g(x) \, dx. \tag{7}$$

Conversely, any minimizer E of (7) 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 (7) is unique up to a negligible set.

The proof of this proposition is relatively easy but is out of the scope of this paper. The first part (the super-level sets are minimizing) was shown for instance in [17], while the second (the converse) comes from a comparison principle for the minimizers of (7) which appears in [5]:

Lemma 2.2 ([5], Lemma 4, (i)). *Let $z > z'$ and E_z, $E_{z'}$ minimize (7) 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 (6) 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 (7). The cases where Ψ is simply convex, or lacks regularity, are also interesting, and partial results still hold in these cases: see [21, 25] where similar ideas are developed. We mention that theses principles have been used in a series of recent papers for studying the properties of minimizers of (6), see [2, 16].

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

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

where ϕ° is the polar of ϕ, defined by $\phi^\circ(\xi) = \sup_{\phi(\nu) \le 1} \nu \cdot \xi$ (and $\phi(\nu) = \sup_{\phi^\circ(\xi) \le 1} \nu \cdot \xi$). Then the perimeter in (7) 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 (6) and (7), since it extends the knowledge of some properties of solutions of one to the other, see [16].

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

$$\min_{u \in BV(\Omega)} \int_\Omega \phi(Du) + \frac{1}{2h}\int_\Omega (u(x) - d_E(x))^2\,dx \tag{8}$$

at least as soon as Ω is "large enough" (with respect to E). Problem (8) 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 (8) 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 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 [42, 47], and, also, to stress the similarities with the continuous setting (in the continuous setting, a general overview of these topics is found in [10]). 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:

$$J(u) = \int_{-\infty}^{+\infty} J(\chi^{\{u \ge z\}})\,dz \tag{9}$$

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 $E \subseteq \{1, \ldots, N\}$ 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 (9) 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 $\lambda \geq 0$.*

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. *$p \in \partial J(u) \Leftrightarrow (\forall z \in \mathbb{R}, p \in \partial J(\chi^{\{u \geq z\}}))$.*

5. *J is submodular: for any $u, u' \in \{0, 1\}^N$,*

$$J(u \vee u') + J(u \wedge u') \leq J(u) + J(u'). \tag{10}$$

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 (9) 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 $J(v') \geq J(v) + p \cdot (v' - v)$ for any v'. Equivalently, in this case, it is the set of $p \in \partial J(0)$ with $J(v) = p \cdot v$. See [27, 50] 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 $u \in \mathbb{R}_+^N$, taking 0 instead of $-\infty$ as the lower bound for the integral in (9). This is well-known in optimization theory as the Lovász' extension of J [43], or the Choquet integral (see for instance [42, Chap. 8] and [47]).

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)

$$J(u) = \sum_{\substack{1 \leq i < M \\ 1 \leq j \leq M}} |u_{i+1,j} - u_{i,j}| + \sum_{\substack{1 \leq i \leq M \\ 1 \leq j < M}} |u_{i,j+1} - u_{i,j}| \tag{11}$$

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

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

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:

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

The latter reduces to pairwise interactions of the form given by Eq. (11) by considering the following: without loss of generality we can assume that we have $u_{i,j} \leq u_{i+1,j} \leq u_{i,j+1}$ 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 Eq. (13) amounts to:

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

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*

$$\min_{u \in \mathbb{R}^N} \lambda J(u) + \frac{1}{2} \|u - g\|^2 \tag{14}$$

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

$$\min_{\theta \in \{0,1\}^N} \lambda J(\theta) + \sum_{i=1}^N \theta_i(z - g_i). \tag{15}$$

This is shown in [18, 26], but is also a consequence of the representation we will introduce in the next section for problems (14) and (15). 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 (15) 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 (7) for the respective values z and z' of the parameter. Then $\theta \leq \theta'$ (in other words, $\{\theta = 1\} \subseteq \{\theta' = 1\}$).*

This key property is proved, at least for a particular case of submodular functions, in [30, Lemma 2.4] (see also the references therein). We also refer the reader to [45] for further extensions of this approach. A proof based on stochastic arguments is found in [26], while we present in Appendix C the elementary proof given in [18].

Quantized total variation minimization problem. We will discuss in the next section how Problem (14) can be (efficiently) solved by successive minimizations of (15). (This was first proposed by Gallo, Grigoriadis and Tarjan [30], and improved by Hochbaum [36]. The authors of the present note rediscovered the latter algorithm [18, 26], following quite different paths.) Then, (14) will be seen as an approximation of (6), for some anisotropic variant of the perimeter. In practice, we cannot expect to solve (15) for all values of t. Hence, we introduce the quantized version of Problem (14):

$$\min \left\{ \lambda J(v) + \frac{1}{2} \|v - g\|^2 : v \in \mathbb{R}^N, \ v_i \in \{l_0 \ldots, l_n\} \, \forall i = 1, \ldots, N \right\} \tag{16}$$

where the real levels $(l_k)_{k=0}^n$ are given. That is, we minimize (14) 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 $k = 1, \ldots, n$, $l_k - l_{k-1} = \delta > 0$ (adaption to other cases is straightforward). Then the following result is true.

Proposition 3.5. *Let v be a solution of (16), and u be the solution of (14). Then for each $i = 1, \ldots, N$, if $l_0 \leq u_i \leq l_n$, $|u_i - v_i| \leq \delta/2$.*

In particular, if $l_0 \leq m$ and $l_n \geq M$, $\max_i |u_i - v_i| \leq \delta/2$. This means that the quantized problem (16) produces exactly a quantization of the solution of (14). 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 [18, Sec. 4]) which 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 (15) and how to use them to solve (14).

4 Parametric and dyadic-parametric maximum flow

4.1 Graph representation of binary energies

It was first observed by Picard and Ratliff [49] 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 [41] 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 [29]. In general, it does not seem to be known whether any submodular J can be represented on a graph in the way proposed in [49, 41]. For instance, it is easy for the particular example (12), although it may involve much more than three variables, see Appendix B. Note that other efficient algorithms exist for minimizing submodular functions [24, 35, 37, 52].

Let us apply this to (15), 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 [14, 34, 41, 49].

We consider problem (15), 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:

$$\begin{cases} c(s,i) \;=\; (z - g_i)^- & i = 1, \dots, N \,, \\ c(i,t) \;=\; (z - g_i)^+ & i = 1, \dots, N \,, \\ c(i,j) \;=\; \lambda \alpha_{i,j} & 1 \le i, j \le N \,. \end{cases} \tag{17}$$

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

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

Hence, up to a constant, it is nothing else than the energy in (15).

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 Ford and Fulkerson [1]. 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 [1]. Although the time complexity of the Boykov and Kolmogorov's maximum flow described in [12] 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 $z = z_1, \ldots, z_n$, with the global complexity of a single one. These approaches follow from the seminal work of Gallo, Grigoriadis and Tarjan [30], with an improvement due to Hochbaum [36].

4.2 Parametric max-flow algorithm

The main idea 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. The authors of [30] show that under these assumptions the monotony of the solutions given by Lemma 3.4 hold. In terms of graph it means the 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 [32] 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 [12]. 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 (17) (assuming we want to solve our problem for $z_1, z_2 = z_1 + \delta, \ldots, z_n = z_1 + (n-1)\delta$), and compute a first residual network. Then, in this new network, we increase by δ all residual capacities $\tilde{c}(i, t)$, $i = 1, \ldots, N$, 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 ([1] 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 [12], 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 [26]).

This approach provides a first, fast method for solving (16). See Subsection 4.4 for some experiments. Note that [39] describes an efficient approach to recompute a maximum-flow that handles arbitrary changes in the graph. The next section describes a slightly faster approach.

4.3 Dyadic-Parametric max-flow algorithm

It was first observed by Hochbaum [36] 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 [18, 26]. 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 [18], a new (disconnected) graph with N nodes was built to implement the corresponding energy, while in [26] 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 [36] 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 (cite [12]) that has been adapted for solving efficiently problem (16) 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 (16). 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 [18, 26] in which new graphs were rebuilt at each step (see the next subsection).

4.4 Comparisons

We now compare these different approaches for solving the discrete ROF model 6. Two kinds of discrete total variation are considered: the first one is given by Example 11, i.e., the image is

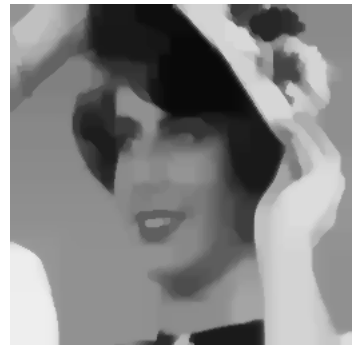(a)   (b)

Figure 1: Two original images: (a) *Cows*, (b) *Girl*.

(a)   (b)   (c)

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

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 have been implemented. The first one relies of a the push-relabel (PR) approach (see for instance [23] for more details), while the second is our adaptation of the maximum-flow implementation of Boykov-Kolmogorov (BK) [12]. Both the parametric and dyadic-parametric approaches are considered. We also compare with the previous approaches of [18, 26] by giving the time results of [26]. Time results for these five different algorithms are given for an Intel Core2 Q6600 processor running at 2.40GHz. They correspond to the average time of 10 runs. Figure 1 depicts two images, *cow* (400×600 and 800×1200) and *girl* (256^2) used for our experiments. Minimizers with the 8-connectivity for the *girl* and *cows* images are respectively depicted in Figure 2 and Figure 3. Time results with different regularization parameters for 4- and 8-connectivity are respectively given in Table 1 and Table 2.

Results clearly shows that the dyadic approach clearly outperforms the pure parametric one by an order of magnitude. We note that the parametric PR is much more dependent to the regularization parameter λ than BK's. We also observe that the order performance depends on the content of the image and on the value of the regularization parameter. Indeed, for small

10

<div align="center">(a)   (b)   (c)</div>

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

regularization ($\lambda = 10$), PR performs better for *cows* but worse and *girl*. This order is reversed for larger regularization.

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 Chambolle/Darbon-Sigelle approach while the PR parametric comes third. Finally note that one can adapt the PR parametric maximum-flow as proposed in [33] to also get an efficient TV minimization using the Push-Relabel approach.

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* [13] 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 may remain stuck). We propose to use (14) as a discretization of the continuous problem (6) (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 [53, 54];

- Solve the discrete version of (8):

$$\min_u J(u) + \frac{1}{2h}\|u - d^n\|^2 \tag{18}$$

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.

Table 1: Time results for several regularization parameter λ using 4-connectivity. Time results are in seconds.

| Images (size)	Approach	$\lambda = 10$	$\lambda = 20$	$\lambda = 60$
	Chambolle/Darbon-Sigelle	1.06	1.29	1.95
	Parametric PR	4.37	7.01	13.97
Cows (400×600)	Dyadic Parametric PR	1.71	2.73	6.51
	Parametric BK	6.13	6.45	7.37
	Dyadic Parametric BK	0.61	0.75	1.10
	Chambolle/Darbon-Sigelle	4.35	5.35	8.50
	Parametric PR	19.73	37.52	198.32
Cows (800×1200)	Dyadic Parametric PR	8.89	15.37	39.25
	Parametric BK	25.46	26.85	30.38
	Dyadic Parametric BK	2.44	3.18	4.83
	Chambolle/Darbon-Sigelle	0.30	0.38	0.52
	Parametric PR	1.83	2.01	2.50
Girl (256^2)	Dyadic Parametric PR	0.65	1.04	1.66
	Parametric BK	1.58	1.67	1.94
	Dyadic Parametric BK	0.17	0.21	0.29
	Chambolle/Darbon-Sigelle	1.17	1.50	2.36
	Parametric PR	14.24	42.96	33.80
Girl (512^2)	Dyadic Parametric PR	3.58	6.24	8.88
	Parametric BK	6.61	7.02	8.06
	Dyadic Parametric BK	0.72	0.92	1.49
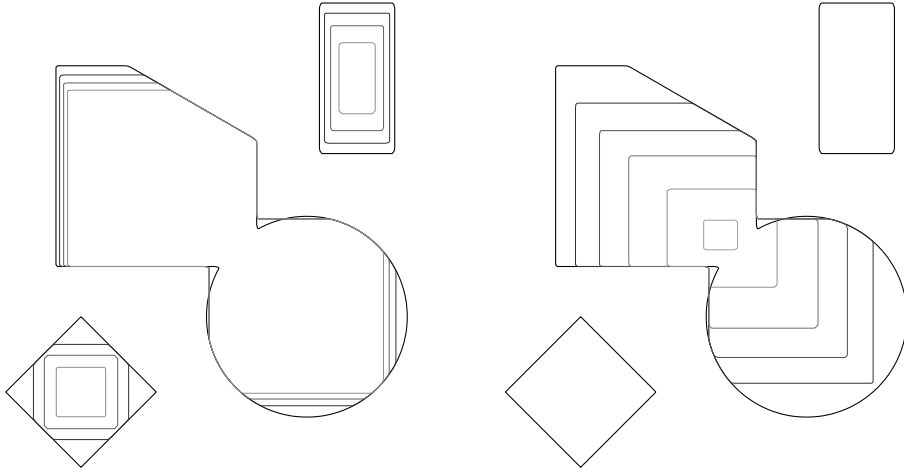
Figure 4: Evolutions with a square anisotropy (thick line: original curve, left: iterations 10, 20, 30, right: iterations 50, 100, 150, 200, 228).

For instance, the two-dimensional function

$$J(u) = \sum_{i,j} |u_{i+1,j} - u_{i,j}| + |u_{i,j+1} - u_{i,j}| \qquad (19)$$

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

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

Table 2: Time results for several regularization parameter λ using 8-connectivity. Time results are in seconds.

Images (size)	Approach	$\lambda = 10$	$\lambda = 20$	$\lambda = 60$
	Chambolle/Darbon-Sigelle	1.98	2.53	4.12
	Parametric PR	7.85	20.09	24.08
Cows (400×600)	Dyadic Parametric PR	4.50	2.72	16.26
	Parametric BK	12.96	13.61	16.03
	Dyadic Parametric BK	1.29	1.81	2.68
	Chambolle/Darbon-Sigelle	8.22	11.01	18.72s
	Parametric PR	145.08	279.44	306.48
Cows (800×1200)	Dyadic Parametric PR	24.81	50.40	111.03
	Parametric BK	48.31	52.31	62.45
	Dyadic Parametric BK	6.08	7.93	12.97
	Chambolle/Darbon-Sigelle	0.55	0.72	1.08
	Parametric PR	5.32	9.24	12.67
Girl (256^2)	Dyadic Parametric PR	1.49	2.05	3.42
	Parametric BK	2.97	3.25	4.04
	Dyadic Parametric BK	0.42	0.55	0.81
	Chambolle/Darbon-Sigelle	2.34	3.05	5.01
	Parametric PR	41.14	71.39	140.79
Girl (512^2)	Dyadic Parametric PR	10.15	12.92	22.17
	Parametric BK	11.90	13.19	17.08
	Dyadic Parametric BK	1.86	2.54	4.19

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 [8] 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

$$d_E^\phi(x) \;=\; \inf_{y \in E} \phi^\circ(x-y) - \inf_{y \notin E} \phi^\circ(y-x) \tag{20}$$

with $\phi^\circ(\xi) = \sup_{\phi(\nu) \leq 1} \nu \cdot \xi$ (see [8] 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 (18) with

$$\min_u J(u) \;+\; \frac{1}{h}\|u - d^n + hg^n\|^2 \,. \tag{21}$$

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 (19). The two examples illustrated by figures 4 and 5 follow the

definition of the crystalline motion in [8]; in each case, the distance function is computed using the polar $\phi°$: 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 [51].
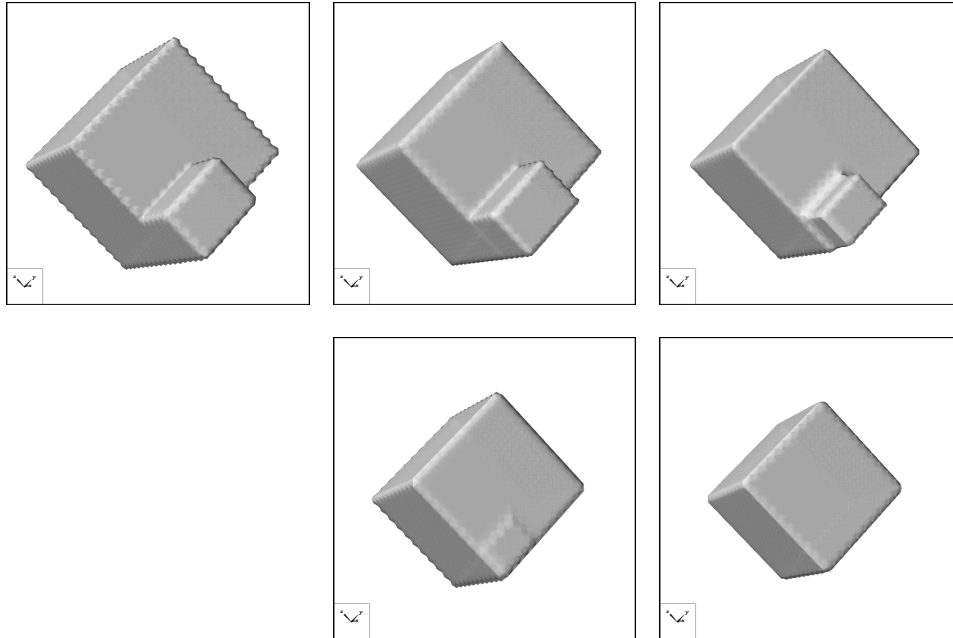

Figure 5: $3D$ evolutions with a cubic anisotropy: original shape and shape at times 1, 4, 7, 10.
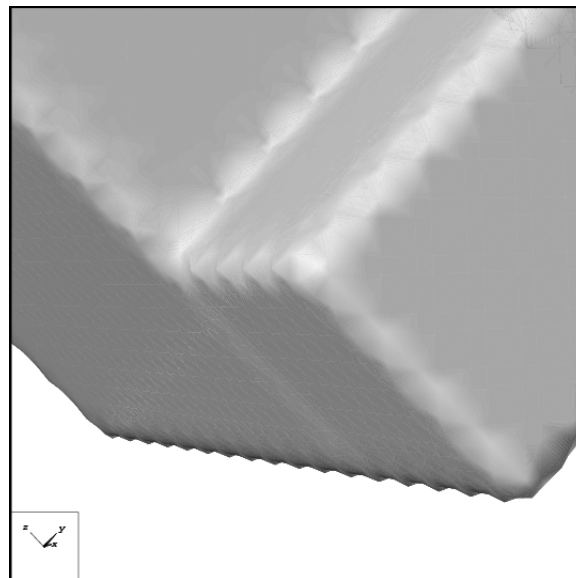

Figure 6: Detail of the "facet breaking" at time 1.

In the $3D$ example of Figure 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 [7, 48], see Figure 6.

It is possible, now, to compute "more isotropic" motions, or motion with more complex

anisotropies (see [11] for a general discussion on this topic). For instance, a hexagonal anisotropy can be implemented using nearest-neighbour interaction on a triangular lattice: see Figure 7, left. Nearly-isotropic evolutions are computed using more complex interactions, for instance, involving next-nearest neighbours and even further neighbours: see figure 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.
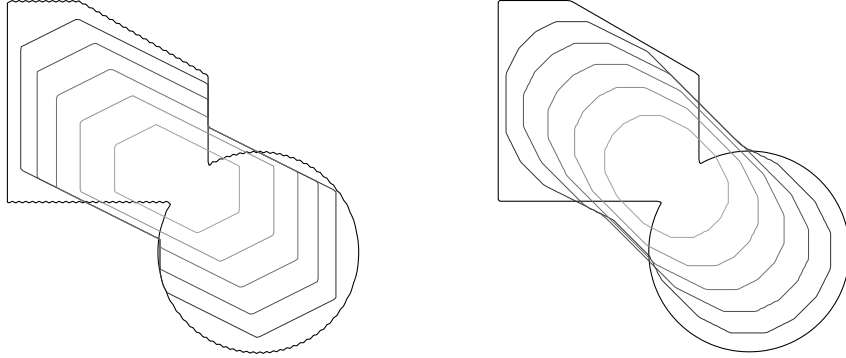

Figure 7: Evolution with a hexagonal anisotropy (left), and nearly-isotropic curvature motion (right), both at times 0, 20, 40, 60, 80, 100.

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 eq. (21)) 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 (18) 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 on Figure 8.
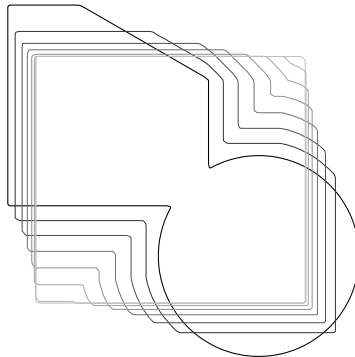

Figure 8: Evolution of a volume preserving crystalline curvature motion.
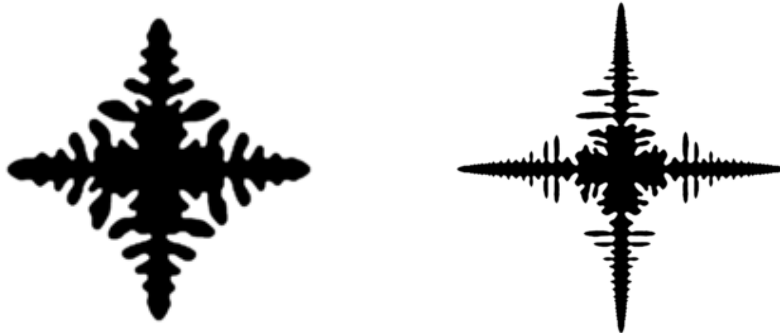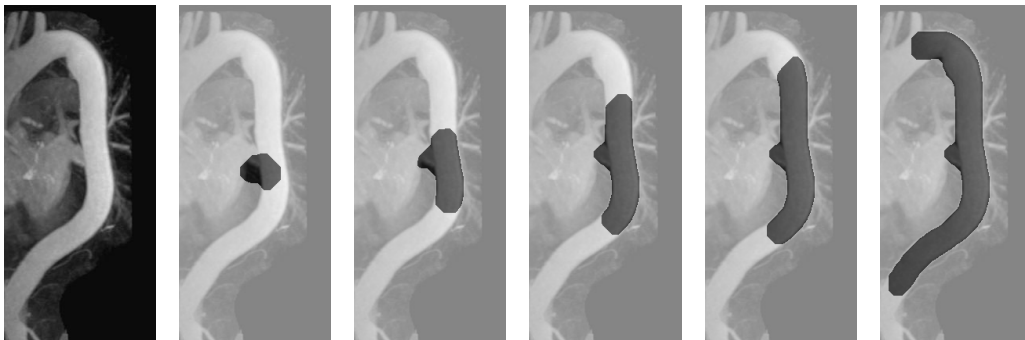
15

Figure 9: Two examples of 2D crystal growths.

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

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 [4], where it is implemented in a more standard way. At each step, we solve problem (21) where the external forcing field g^n depends on the temperature and is recomputed at each iteration. Results are shown on figure 9.

Our last example is an implementation of an active contour (snake) model, more precisely a "balloon", firstly introduced in [22]. 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.

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 $z \notin \{u_i, i = 1, \ldots, N\}$, $\chi^{\{u^n > z\}} = \chi^{\{u > z\}}$ 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)

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

(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 (9),

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

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

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

Hence (10) holds. If now u, u' are not binary, we still have

$$J(u \vee u') + J(u \wedge u') = \int_{-\infty}^{+\infty} J\left(\chi^{\{u \vee u' \geq z\}}\right) + J\left(\chi^{\{u \wedge u' \geq z\}}\right) dz$$

$$= \int_{-\infty}^{+\infty} J\left(\chi^{\{u \geq s\}} \vee \chi^{\{u' \geq s\}}\right) + J\left(\chi^{\{u \geq z\}} \wedge \chi^{\{u' \geq z\}}\right) dz \leq \int_{-\infty}^{+\infty} J\left(\chi^{\{u \geq z\}}\right) + J\left(\chi^{\{u' \geq z\}}\right) dz$$

$$= J(u) + J(v).$$

We now want to consider the converse assertion, that is, the convexity of the extension through (9) of a nonnegative submodular function. We consider $J : \{0, 1\}^N \to [0, +\infty]$ a submodular function, i.e., such that (10) 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 (9). (We observe that thanks to $J(\mathbf{1}) = 0$, if $u \in \{0, 1\}^N$ then (9) 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 [46, 42]. We propose here a very elementary proof.

First of all, points (1-3) of the thesis of Proposition 3.1 are deduced only from (9) 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

$$J(u + v) \leq J(u) + J(v) \tag{22}$$

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:

$$J(u) = \min\left\{\sum_{l=1}^n J(\theta^l) : n \geq 0, \theta^l \in \{0, 1\}^N, \sum_{l=1}^n \theta^l = u\right\}. \tag{23}$$

Indeed, denote by $H(u)$ the right-hand side of 23. Since

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

where $n = \max_i u_i$, we have $H(u) \leq J(u)$. The reverse inequality will hold if we show that the minimum in (23) is reached precisely for $n = \max_i u_i$ and $\theta^l = \chi^{\{u \geq l\}}$ (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 (10) is strict whenever the vectors are not ordered), then it is obvious: indeed, if the minimum in (23) is reached for $(\theta^l)_{l=1}^n$ and there are l, l' such that $\theta^l \not\leq \theta^{l'}$ nor $\theta^l \not\geq \theta^{l'}$, then, replacing θ^l

17

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 $\theta^l = \chi^{\{u \geq l\}}$, $l = 1, \ldots, n = \max_i u_i$.

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 (23) for J_ε will be attained for $\theta^l = \chi^{\{u \geq l\}}$, $l = 1, \ldots, n = \max_i u_i$. Passing to the limit, we still get that $H(u) = J(u)$ so that (23) is true.

Let us now show (22) 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 $m < \min\{u_i, v_i, i = 1, \ldots, N\}$ and $M > \max\{u_i, v_i, i = 1, \ldots, N\}$. 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

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

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 (22), and we find

$$J(u^\varepsilon + v^\varepsilon) = \varepsilon J(u_\#^\varepsilon + v_\#^\varepsilon) \leq \varepsilon(J(u_\#^\varepsilon) + J(v_\#^\varepsilon)) = J(u^\varepsilon) + J(v^\varepsilon).$$

Since the right-hand side converges to $J(u) + J(v)$ as $\varepsilon \to 0$, and J is l.s.c., we deduce (22). Hence J is convex.

Remark A.1. By standard convex analysis (see e.g. [27, 50]), we deduce that

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

where

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

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 (9) 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).

B   Representation of submodular functions on graphs

Following [40, 41], we say that the (necessarily submodular) $J(\theta_1, \ldots, \theta_N)$ can be represented on a graph if there exists M additional nodes $i \in \{N + 1, \ldots, N + M\}$ and weights $\alpha_{i,j} \geq 0$, $\beta_i \in \mathbb{R}$ $(i, j \in \{1, \ldots, N + M\})$ such that for any $\theta \in \{0, 1\}^N$,

$$J(\theta_1, \ldots, \theta_N) = \min_{(\theta_{N+1}, \ldots, \theta_{N+M}) \in \{0,1\}^M} \sum_{i,j=1}^{N+M} \alpha_{i,j}(\theta_i - \theta_j)^+ + \sum_{i=1}^{N+M} \beta_i \theta_i. \tag{24}$$

The energy in the right-hand side of (24) is clearly representable on a graph, following the standard construction in (4.1): hence (24) 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 [40, 41], it is shown that this is possible if $N \leq 3$ (and, of course, for total energies that are the sum of representable energies), at the cost of adding $M = 2$ additional nodes. See also [29, 9].

Notice, 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 (12) is a sum of terms of the following type:

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

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

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

where

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

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

$$J_1(\theta_1, \ldots, \theta_N) \;=\; g(\sum_{i=1}^N \theta_i)$$

is also representable.

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

$$\lambda J(\theta) + \sum_{i=1}^N \theta_i(z - g_i) \;\leq\; \lambda J(\theta \wedge \theta') + \sum_{i=1}^N (\theta_i \wedge \theta_i')(z - g_i), \qquad \text{and:}$$
$$\lambda J(\theta') + \sum_{i=1}^N \theta_i'(z' - g_i) \;\leq\; \lambda J(\theta \vee \theta') + \sum_{i=1}^N (\theta_i \vee \theta_i')(z' - g_i).$$

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

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

This is nothing else than

$$z \sum_{i=1}^N \theta_i - \theta_i \wedge \theta_i' \;\leq\; z' \sum_{i=1}^N \theta_i \vee \theta_i' - \theta_i',$$

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 $i = 1, \ldots, N$, that is, $\theta \leq \theta'$. $\qquad \square$

Proof of Prop. 3.3. We easily derive Proposition 3.3: indeed, if θ^z solve (15) for all values of z, and if we define $u \in \mathbb{R}^N$ by

$$u_i \;=\; \sup\{z : \theta_i^z = 1\}$$

then clearly $\chi^{\{u > z\}} \leq \theta^z \leq \chi^{\{u \geq z\}}$ for all z, as a consequence of Lemma 2.2. Also, $m = \min_j g_j \leq u_i \leq \max_j g_j = M$ for all i (since $\mathbf{1}$ is the unique solution of (15) if $z \leq m$, while 0 is the solution

if $z \geq M$). Hence, if $v \in \mathbb{R}^N$ and $m' \leq m \wedge (\min_i v_i)$, we have (using (9) and the minimality of each θ^z)

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

which shows our claim. $\qquad\square$

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 Prop. 3.5. In fact, For an admissible v in (16), 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 $k \geq 1$, θ^k is the binary vector defined by $\theta_i^k = 1$ iff $v_i \geq l_k$. Then, the fact $\theta^k \leq \theta^{k-1}$ for any $k \geq 2$, and the co-area formula (9), yield $J(z) = \sum_{k=1}^{n} \delta J(\theta^k)$. On the other hand,

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

hence, up to a constant, problem (16) is the same as

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

where the min is taken on all binary fields $(\theta^k)_{k=1}^n$, with the constraint that $\theta^k \leq \theta^{k-1}$ for any $k = 2, \ldots, n$. Each term in the sum is the energy that appears in problem (15), for $z = z_k = (l_k + l_{k-1})/2$. Now, by Lemma 3.4, if for each $k = 1, \ldots, n$, θ^k is a minimizer of the corresponding energy, then, $z_k > z_{k-1}$ yields $\theta^k \leq \theta^{k-1}$: 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 $\{u \geq z_k\}$. This shows that Proposition 3.5 is true. $\qquad\square$

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network flows.* Prentice Hall Inc., Englewood Cliffs, NJ, 1993. Theory, algorithms, and applications.

[2] W. K. Allard. Total variation regularization for image denoising, I. Geometric theory. *SIAM J. Math. Anal.*, 39(4):1150–1190, 2007.

[3] F. Almgren, J. E. Taylor, and L.-H. Wang. Curvature-driven flows: a variational approach. *SIAM J. Control Optim.*, 31(2):387–438, 1993.

[4] R. Almgren. Variational algorithms and pattern formation in dendritic solidification. *J. Comput. Phys.*, 106(2):337–354, 1993.

[5] F. Alter, V. Caselles, and A. Chambolle. A characterization of convex calibrable sets in $\mathbb{R}^N$. *Math. Ann.*, 332(2):329–366, June 2005.

[6] G. Bellettini, V. Caselles, A. Chambolle, and M. Novaga. Crystalline mean curvature flow of convex sets. *Arch. Ration. Mech. Anal.*, 179(1):109–152, 2006.

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

[8] G. Bellettini and M. Paolini. Anisotropic motion by mean curvature in the context of Finsler geometry. *Hokkaido Math. J.*, 25(3):537–566, 1996.

[9] E. Boros and P. L. Hammer. Pseudo-boolean optimization. *Discrete Appl. Math.*, 123(1-3):155–225, 2002.

[10] G. Bouchitté. Recent convexity arguments in the calculus of variations. (lecture notes from the 3rd Int. Summer School on the Calculus of Variations, Pisa), 1998.

[11] Y. Boykov and V. Kolmogorov. Computing geodesics and minimal surfaces via graph cuts. In *International Conference on Computer Vision*, pages 26–33, 2003.

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

[13] Y. Boykov, V. Kolmogorov, D. Cremers, and A. Delong. An integral solution to surface evolution PDEs via Geo-Cuts. In A. Leonardis, H. Bischof, and A. Pinz, editors, *European Conference on Computer Vision (ECCV)*, volume 3953 of *LNCS*, pages 409–422, Graz, Austria, May 2006. Springer.

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

[15] V. Caselles and A. Chambolle. Anisotropic curvature-driven flow of convex sets. *Nonlinear Anal.*, 65(8):1547–1577, 2006.

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

[17] A. Chambolle. An algorithm for mean curvature motion. *Interfaces Free Bound.*, 6(2):195–218, 2004.

[18] A. Chambolle. Total variation minimization and a class of binary mrf models. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, Lecture Notes in Computer Science, pages 136–152, 2005.

[19] A. Chambolle and M. Novaga. Implicit time discretization of the mean curvature flow with a discontinuous forcing term. Preprint CMAP #571, Ecole Polytechnique, France, March 2005.

[20] A. Chambolle and M. Novaga. Approximation of the anisotropic mean curvature flow. *Math. Models Methods Appl. Sci.*, 17(6):833 − 844, 2007.

[21] T. F. Chan and S. Esedoḡlu. Aspects of total variation regularized L^1 function approximation. *SIAM J. Appl. Math.*, 65(5):1817–1837 (electronic), 2005.

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

[23] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, 2001.

[24] W.H. Cunningham. On submodular function minimization. *Combinatoria*, 5:185–192, 1985.

[25] J. Darbon. 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.

[26] J. Darbon and M. Sigelle. Image restoration with discrete constrained Total Variation part I: Fast and exact optimization. *Journal of Mathematical Imaging and Vision*, 26(3):261–276, December 2006.

[27] I. Ekeland and R. Témam. *Convex analysis and variational problems*, volume 28 of *Classics in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, english edition, 1999. Translated from the French.

[28] H. Federer. *Geometric measure theory*. Springer-Verlag New York Inc., New York, 1969.

[29] D. Freedman and P. Drineas. Energy minimization via graph cuts: settling what is possible. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 939–946, 2005.

[30] G. Gallo, M. D. Grigoriadis, and R. E. Tarjan. A fast parametric maximum flow algorithm and applications. *SIAM J. Comput.*, 18(1):30–55, 1989.

[31] E. Giusti. *Minimal surfaces and functions of bounded variation*, volume 80 of *Monographs in Mathematics*. Birkhäuser Verlag, Basel, 1984.

[32] A. V. Goldberg and R. E. Tarjan. A new approach to the maximum flow problem. In *STOC '86: Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pages 136–146, New York, NY, USA, 1986. ACM Press.

[33] D. Goldfarb and Y. Yin. Parametric maximum flow algorithms for fast total variation minimization. Technical report, Rice University, 2007.

[34] D. M. Greig, B. T. Porteous, and A. H. Seheult. Exact maximum a posteriori estimation for binary images. *J. R. Statist. Soc. B*, 51:271–279, 1989.

[35] M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatoria*, 1:169–197, 1981.

[36] D. S. Hochbaum. An efficient algorithm for image segmentation, Markov random fields and related problems. *J. ACM*, 48(4):686–701 (electronic), 2001.

[37] S. Iwata, L. Fleischer, and S. Fujishige. A combinatorial, strongly polynomial-time algorithm for minimizing submodular functions. *Journal of the ACM*, pages 97–106, 2000.

[38] O. Juan and Y. Boykov. Active graph cuts. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1023–1029, 2006.

[39] P. Kholi and P. Torr. Efficient solving dynamic markov random fields using graph cuts. In *Proceedings of the 10th International International Conference on Computer Vision*, pages 922–929, 2005.

[40] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? In *European Conference on Computer Vision*, volume 3, pages 65–81, may 2002.

[41] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2(26):147–159, 2004.

[42] J. Lee. *A first course in combinatorial optimization*. Cambridge University Press, 2004.

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

[44] S. Luckhaus and T. Sturzenhecker. Implicit time discretization for the mean curvature flow equation. *Calc. Var. Partial Differential Equations*, 3(2):253–271, 1995.

[45] S. T. McCormick. 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*, pages 319 − 328, 1996.

[46] K. Murota. *Discrete convex analysis*. SIAM Monographs on Discrete Mathematics and Applications. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2003.

[47] K. Murota. *Discrete Convex Optimization*. SIAM Society for Industrial and Applied Mathematics, 2003.

[48] M. Paolini and F. Pasquarelli. Numerical simulation of crystalline curvature flow in 3D by interface diffusion. In *Free boundary problems: theory and applications, II (Chiba, 1999)*, volume 14 of *GAKUTO Internat. Ser. Math. Sci. Appl.*, pages 376–389. Gakkōtosho, Tokyo, 2000.

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

[50] R. T. Rockafellar. *Convex analysis*. Princeton Landmarks in Mathematics. Princeton University Press, Princeton, NJ, 1997. Reprint of the 1970 original, Princeton Paperbacks.

[51] E. Rouy and A. Tourin. A viscosity solutions approach to shape-from-shading. *SIAM J. Numer. Anal.*, 29(3):867–884, 1992.

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

[53] J. A. Sethian. Fast marching methods. *SIAM Rev.*, 41(2):199–235 (electronic), 1999.

[54] J. N. Tsitsiklis. Efficient algorithms for globally optimal trajectories. *IEEE Trans. Automat. Control*, 40(9):1528–1538, 1995.