Available online at www.sciencedirect.com

ScienceDirect

Comput. Methods Appl. Mech. Engrg. 196 (2007) 3898–3914

Computer methods in applied mechanics and engineering

www.elsevier.com/locate/cma

Discrete gradient flows for shape optimization and applications

G. Doğan [a,1], P. Morin [b,2], R.H. Nochetto [c,*,3], M. Verani [d,4]

[a] *Department of Mathematics, University of Maryland, College Park, MD 20742, USA*
[b] *Departamento de Matemática, Facultad de Ingeniería Química, Instituto de Matemática Aplicada del Litoral, Universidad Nacional del Litoral, CONICET, Güemes 3450, 3000 Santa Fe, Argentina*
[c] *Department of Mathematics, Institute for Physical Science and Technology, University of Maryland, College Park, MD 20742, USA*
[d] *MOX, Dipartimento di Matematica, Politecnico di Milano, 20133 Milano, Italy*

Received 1 November 2005; accepted 30 October 2006
Available online 27 March 2007

Dedicated to Ivo Babuška on the occasion of his 80th birthday.

Abstract

We present a variational framework for shape optimization problems that establishes clear and explicit connections among the continuous formulation, its full discretization and the resulting linear algebraic systems. Our approach hinges on the following essential features: shape differential calculus, a semi-implicit time discretization and a finite element method for space discretization. We use shape differential calculus to express variations of bulk and surface energies with respect to domain changes. The semi-implicit time discretization allows us to track the domain boundary without an explicit parametrization, and has the flexibility to choose different descent directions by varying the scalar product used for the computation of normal velocity. We propose a Schur complement approach to solve the resulting linear systems efficiently. We discuss applications of this framework to image segmentation, optimal shape design for PDE, and surface diffusion, along with the choice of suitable scalar products in each case. We illustrate the method with several numerical experiments, some developing pinch-off and topological changes in finite time.
© 2007 Elsevier B.V. All rights reserved.

MSC: 49Q10; 65M60; 65K10

Keywords: Shape optimization; Scalar product; Gradient flow; Semi-implicit discretization; Finite elements; Surface diffusion; Image segmentation

1. Shape optimization and gradient flows

Shape optimization problems are ubiquitous in science, engineering and industrial applications. They can be formulated as minimization problems with respect to the shape of a domain Ω in $\mathbb{R}^d$. If $y(\Omega)$ is the solution of a boundary value problem in Ω

$$Ly(\Omega) = 0, \tag{1}$$

and $J(\Omega, y(\Omega))$ is a cost functional, then we consider the minimization problem

$$\Omega^* \in \mathscr{U}_{\mathrm{ad}}: \quad J(\Omega^*, y(\Omega^*)) = \inf_{\Omega\in\mathscr{U}_{\mathrm{ad}}} J(\Omega, y(\Omega)), \tag{2}$$

* Corresponding author. Tel.: +1 301 405 5108; fax: +1 301 314 0827.
E-mail addresses: gunay@math.umd.edu (G. Doğan), pmorin@math.unl.edu.ar (P. Morin), rhn@math.umd.edu (R.H. Nochetto), marco.verani@mate.polimi.it (M. Verani).

[1] Partially supported by NSF Grants DMS-0204670 and DMS-0505454.
[2] Partially supported by CONICET through Grant PEI 6208, by SeCyT of Argentina through Grant PICT 03-13023 and by NSF Grant DMS-0204670.
[3] Partially supported by NSF Grants DMS-0204670 and DMS-0505454.
[4] Partially supported by the European IHP Network "Breaking Complexity", by NSF Grant DMS-0204670 and by M.U.R.S.T. Cofin 2003 Grant "Numerical Modelling for Scientific Computing and Advanced Applications".

0045-7825/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.cma.2006.10.046

where $\mathscr{U}_{\text{ad}}$ is a set of admissible domains in $\mathbb{R}^d$. If the problem is purely geometric, namely there is no state constraint (1), then we simply denote the functional $J(\Omega)$. We refer to the books [11,15,19,21,24] for a description of this problem and numerous applied examples (see e.g. [16,20]). In any case, we review some basic material in Section 2 and discuss three relevant examples throughout the paper.

Our main goal is to present a variational method which explicitly and clearly leads first to design a flow $\Omega(t)$, starting from an initial configuration $\Omega(0)$ to a relative minimum $\Omega(\infty)$, that decreases the function $t \mapsto J(\Omega(t), y(\Omega(t)))$, and next to discretize in time and space, thereby obtaining a computable descent direction, and a sequence of approximate domains $\{\Omega_n\}$. Our approach hinges on three essential features:

- *Shape sensitivity analysis*: This allows us to express variations of bulk and surface energies with respect to domain changes and formalize the notion of shape derivative and shape gradient.
- *Semi-implicit time discretization*: This is crucial in order to maintain an implicit computation of geometric quantities such as mean curvature and normal velocity but not the entire geometry. This can be realized without explicit parametrization of the domain boundary, and is sufficiently flexible to accommodate several scalar products for the computation of normal velocity depending on the application.
- *Adaptive finite element method for space discretization*: This is important for the intrinsic computation of mean curvature as well as the control of local meshsize to increase resolution.

We discuss shape sensitivity in Section 2, with special emphasis on our three sample problems, and present the time and space discretization of the resulting gradient flows in Section 3. We finally conclude in Section 4 with several numerical experiments that illustrate performance of the method, choice of scalar products, and large domain deformations sometimes leading to pinch-off and topological change.

In the rest of the introduction we briefly describe our three basic model problems and the notion of gradient flow. We now introduce our examples: image segmentation, optimal shape design for PDE, and surface diffusion. They are simple models of shape optimization with quite distinct behavior and requirements, which can nonetheless be studied within a unified framework. We make also explicit the concept of shape derivative of $J(\Omega)$ in the direction of a normal velocity V, namely

$$\mathrm{d}J(\Omega; V) = \int_\Gamma GV\,\mathrm{d}S, \tag{3}$$

but derive the expressions of G in Section 2 for each case. We then indicate how to exploit this information to design a gradient flow. Throughout the paper we will denote with Γ that part of the boundary of Ω which is free to deform, with κ the sum of the principal curvatures of Γ and with $\vec{v}$ the outer unit normal of Γ; thus $V := \vec{V} \cdot \vec{v}$. We use the sign convention that a circle with outward normal has positive mean curvature. The symbol $\langle \cdot, \cdot \rangle$ stands for either the L^2-scalar product or a duality pairing on Γ.

1.1. Image segmentation

Image segmentation has been one of the central problems of image processing ever since the inception of this discipline. Given an image the goal is to identify the "objects" or homogeneous regions with respect to some image features, such as image intensity, texture etc. The *geodesic active contour model* proposed in [9] addresses this problem in an energy minimization context and identifies object boundaries by a set of curves in 2D or surfaces in 3D. In the following we cast this model within our framework.

Let $I(x) : \mathscr{D} \subset \mathbb{R}^d \to \mathbb{R}$ be a given smoothed-out image intensity function on an open and bounded image domain $\mathscr{D}$. Since values of $I(x)$ vary significantly at object boundaries, the image gradient $\nabla I(x)$ gets large at these locations. We can use this to define the edge detector function $H(x)$ as follows:

$$H(x) = h(|\nabla I(x)|), \quad h(s) = \frac{1}{1+s^2},$$

so that $H(x)$ is small on object boundaries and $H(x) \approx 1$ on smooth regions of the image. We now associate an energy $J(\Omega)$ to a given curve Γ and enclosed domain Ω so that object boundaries correspond to local minima of $J(\Omega)$. Such an energy is given by the geometric functional

$$J(\Omega) := \int_\Gamma H(x)\,\mathrm{d}S + \lambda \int_\Omega H(x)\,\mathrm{d}x, \quad \lambda \geqslant 0. \tag{4}$$

Note that the first integral is minimized when Γ coincides with the object boundaries in the image. It is also common to include the domain integral in the optimization process because it speeds up the convergence of the curve to the object boundaries and helps detection of object concavities; see [5,9] for more details. We will see in Section 2.2.1 that G in (3) has the explicit form

$$G = (\kappa + \lambda)H(x) + \partial_\nu H(x). \tag{5}$$

1.2. Optimal shape design for PDE

Motivated by the optimal shape design of a drug eluting stent, we consider an extremely simplified problem, still presenting some of the main mathematical difficulties one has to face in trying to solve a more realistic situation (for more details on the mathematical modelling see e.g. [27]). A drug eluting stent is a normal metal stent that has been coated with a drug that is known to interfere with the process of restenosis of the artery. Roughly speaking in optimal shape design for drug eluting stents, one is interested in optimizing some of the geometric properties of the stent in order to control the distribution of the drug released in the arterial wall.

In this context Ω is the cross-section of part of the arterial wall, Γ is the boundary of the cross-section of a stent wire, D is the control region for the drug distribution and z_g is some clinical data to match.

Let Ω_i, $i = 1, 2$ be sufficiently regular open bounded sets of $\mathbb{R}^d$, such that $\Omega_1 \subset \Omega_2$. We denote by $\Omega = \Omega_2 \setminus \overline{\Omega}_1$ and $\partial\Omega = \Gamma \cup \Sigma$, where $\Gamma = \partial\Omega_1$ and $\Sigma = \partial\Omega_2 \setminus \Gamma$. Finally, let D be an open bounded set of $\mathbb{R}^d$ such that $D \subset\subset \Omega$. Let us now define the set $\mathscr{U}_{\text{ad}}$ of admissible domains in $\mathbb{R}^d$: it contains all domains obtained through a deformation of Ω by keeping Σ fixed and by moving only Γ in such a way that $\Gamma \cap D = \emptyset$.

We are interested in the solution of a simple shape optimization problem of the form (2), associated to the energy functional

$$J(\Omega, y(\Omega)) := \frac{1}{2}\int_D \left(y(\Omega) - z_g\right)^2 \mathrm{d}x + \gamma \int_\Gamma \mathrm{d}S, \tag{6}$$

where $\gamma > 0$ is a penalization parameter for the length of the moving boundary Γ, $z_g : D \to \mathbb{R}$ is a given function and $y(\Omega)$ is the solution of the following elliptic problem on Ω

$$-\Delta y = 0 \quad \text{in } \Omega, \qquad y = 0 \quad \text{on } \Sigma, \qquad \partial_\nu y = 1 \quad \text{on } \Gamma. \tag{7}$$

We will see in Section 2.2.2 that G in (3) has the explicit form

$$G := -\nabla_\Gamma y \nabla_\Gamma p + \kappa p + \kappa\gamma, \tag{8}$$

where p is the solution of a suitable elliptic problem (the adjoint problem) on the domain Ω (see (42)).

1.3. Surface diffusion and epitaxially stressed solids

A very simple model of epitaxially stressed thin films can be described as follows [4,8,25]. Consider an elastic solid with lattice spacing different from that of a substrate. This mismatch induces stresses in the solid. On the other hand, material particles on the free surface Γ in contact with gas are free to move and rearrange their position so as to minimize surface tension, thereby yielding a plastic deformation of the solid. Phenomenological arguments lead to the fourth order (highly nonlinear) PDE

$$V = \Delta_\Gamma(\gamma\kappa + \epsilon), \tag{9}$$

where γ is the surface tension constant and ϵ is the elastic energy density on Γ. In this paper we consider a simplified situation in that elasticity is replaced by the Laplace operator in Ω and thus $\epsilon := |\nabla y(\Omega)|^2$, where $y(\Omega)$ solves the boundary value problem

$$\begin{aligned} &-\Delta y(\Omega) = 0 \quad \text{in } \Omega, \\ &\partial_\nu y(\Omega) = 0 \quad \text{on } \Gamma, \qquad y(\Omega) = g \quad \text{on } \Sigma, \end{aligned} \tag{10}$$

where Σ is that part of the boundary of Ω in contact with the substrate (to mimic a misfit). We will see in Section 2.2.3 that the physical law (9) is the H^{-1}-gradient flow for the energy functional

$$J(\Omega, y(\Omega)) := \int_\Omega |\nabla y(\Omega)|^2 + \gamma \int_\Gamma \mathrm{d}S \tag{11}$$

and we will show in Section 2.2.3 that G in (3) has the explicit form

$$G := |\nabla y(\Omega)|^2 + \gamma\kappa. \tag{12}$$

1.4. Shape gradient flows

We observe that in all the examples above, the shape derivative with respect to a normal velocity V is given by (3) with G of the form

$$G = g(x, \Omega)\kappa + f(x, \Omega). \tag{13}$$

This explicit expression can be exploited to deform Ω in the direction V of maximal decrease of the functional $J(\Omega, y(\Omega))$. Instead of considering the simple-minded gradient flow $V = -G$ we allow for more flexibility by introducing a scalar product $b(\cdot,\cdot)$ on Γ and letting V be the solution to

$$b(V, W) = -\int_\Gamma GW, \quad \forall W \in B(\Gamma), \tag{14}$$

where $B(\Gamma)$ is the Hilbert space induced by $b(\cdot,\cdot)$. Here Γ (and hence G) implicitly depend on $\vec{V} = V\vec{\nu}$ by means of a suitable system of ODE describing the deformation of Ω through V. Since $b(\cdot,\cdot)$ is a scalar product $0 \leqslant b(V,V) = -\int_\Gamma GW = -\mathrm{d}J(G,V)$ and thus V is a descent direction. If $\mathscr{B}$ is an (elliptic) operator such that $\langle \mathscr{B}V, W\rangle = b(V,W)$, then (14) is equivalent to solving the following elliptic PDE on the surface Γ for the normal velocity V:

$$\mathscr{B}V = -G. \tag{15}$$

In this case, the gradient flow given by (14) is called b- or $\mathscr{B}$-gradient flow.

We point out that so far we have not discretized the underlying problem but still have been able to find an equation such that its solution V turns out to be a descent direction for the domain shape, the *gradient descent* direction.

The next step is to discretize in time and space in order to obtain a fully practical algorithm. We will do first the time discretization. Since the mean curvature is the surface Laplacian of the position vector, computing it explicitly leads to instabilities similar to those found in the discretization of the heat equation, where, roughly speaking, the time discretization step is forced to be smaller than the square of the minimum space discretization parameter. When the time discretization step is above that threshold, spurious oscillations at the mesh level appear on the solution. Since we plan to use adaptive meshes for representing the surfaces with the least amount of degrees of freedom, the space discretization parameter might be very small at certain areas. Such a restriction on the time step will make computations very slow and an explicit computation of the curvature will destroy the regularity of the discrete surface Γ. We will thus use a time discretization keeping an implicit computation of curvature in (13). Such a discretization has already been successfully used for mean curvature flow [13] and surface diffusion [6]. An implicit treatment of the curvature while keeping the other geometric quantities explicit provides a natural linearization of the problem. This linearization process is fully discussed in Section 3.3 and is followed by space discretization via finite element methods in Section 3.5. The ensuing variational approach is rather flexible to accommodate several scalar products $b(\cdot,\cdot)$ depending on the application, as discussed in Section 3.3 and 4. Roughly speaking we can distinguish between applications where the gradient flow has a physical meaning (e.g. surface diffusion), and where it does not (e.g. image segmentation or optimal shape design for PDE). In the first case the choice of the scalar product is dictated by physics, whereas in the latter case it can be driven by issues concerning the well-posedness of (15), as discussed in the example of optimal shape design for PDE, or by stability and rate of convergence of the resulting numerical scheme, as described in the example of image segmentation.

In designing a numerical scheme (e.g. gradient method) for the approximation of the continuous gradient flow (14), and hence the construction of a sequence of domains $\{\Omega_n\}_{n\geqslant 0}$ aiming at convergence to $\Omega^* = \operatorname{argmin}_{\Omega\in\mathscr{U}_{\mathrm{ad}}} J(\Omega)$, the following chief question arises:

$$\text{Given a domain } \Omega, \text{ is it possible to choose a vector field } \vec{V} \text{ deforming } \Omega \text{ into } \tilde{\Omega} \text{ such that } J(\tilde{\Omega}) < J(\Omega)? \tag{16}$$

A key step in answering this question is the shape sensitivity analysis of the mapping $\Omega \mapsto J(\Omega)$, which is briefly reviewed in Section 2.

2. Shape sensitivity analysis

In Section 2.1 we introduce some elements of shape calculus, along with related references, necessary to properly carry out the shape sensitivity analysis of the model problems in Section 2.2.

2.1. Shape differential calculus

We start by briefly recalling some useful notions of differential geometry. Let us be given $h \in C^2(\Gamma)$ and an extension $\tilde{h}$ of h, $\tilde{h} \in C^2(U)$ and $\tilde{h}|_\Gamma = h$ on Γ where U is a tubular neighborhood of Γ in $\mathbb{R}^d$. Then the *tangential gradient* $\nabla_\Gamma h$ of h is defined as follows:

$$\nabla_\Gamma h = \left(\nabla\tilde{h} - \partial_\nu \tilde{h}\vec{\nu}\right)|_\Gamma,$$

where $\vec{\nu}$ denotes the normal vector to Γ. For an open set of class C^2 with boundary Γ, we define the *tangential divergence* of $\vec{W}$ by

$$\operatorname{div}_\Gamma \vec{W} = \left(\operatorname{div}\vec{W} - \vec{\nu}\cdot D\vec{W}\cdot\vec{\nu}\right)|_\Gamma, \tag{17}$$

where $D\vec{W}$ denotes the Jacobian matrix of $\vec{W}$. Finally, if $D^2\tilde{h}$ denotes the Hessian of $\tilde{h}$, then the *Laplace–Beltrami operator* Δ_Γ on Γ is defined as follows:

$$\Delta_\Gamma h = \operatorname{div}_\Gamma(\nabla_\Gamma h) = \left(\Delta\tilde{h} - \vec{\nu}\cdot D^2\tilde{h}\cdot\vec{\nu} - \kappa\partial_\nu\tilde{h}\right)|_\Gamma. \tag{18}$$

2.1.1. The velocity method

We consider now a hold-all domain $\mathscr{D}$, which contains Ω, and a vector field $\vec{V}$ defined on $\mathscr{D}$, which is used to define the continuous sequence of perturbed sets $\{\Omega_t\}_{t\geqslant 0}$, with $\Omega_0 := \Omega$. Each point $x \in \Omega_0$ is continuously deformed by an ODE defined by the field $\vec{V}$. The parameter which controls the amplitude of the deformation is denoted by t (a fictitious time).

We now consider the system of ODEs

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \vec{V}(x(t)), \quad \forall t \in [0,T], \ x(0) = X, \tag{19}$$

where $X \in \Omega_0 = \Omega$. This defines the mapping

$$x(\cdot,t) : X \in \Omega \to x(X,t) \in \mathbb{R}^d \tag{20}$$

and also the perturbed sets

$$\Omega_t = \{x(X,t) : X \in \Omega_0\}. \tag{21}$$

We recall that the family of perturbed sets has its regularity preserved for $\vec{V}$ smooth enough [24]: if Ω_0 is of class C^r, $r \leqslant k$, then for any $t \in [0,T]$, Ω_t is also of class C^r.

2.1.2. Derivative of shape functionals

Let $J(\Omega)$ be a shape functional; examples of such functionals have been given in Section 1. The Eulerian derivative, or *shape derivative*, of the functional $J(\Omega)$ at Ω, in the direction of the vector field $\vec{V}$ is defined as the limit

$$\mathrm{d}J(\Omega;\vec{V}) = \lim_{t\to 0}\frac{1}{t}(J(\Omega_t) - J(\Omega)). \tag{22}$$

Let B be a Hilbert space of perturbating vector fields. The functional $J(\Omega)$ is said to be shape differentiable at Ω in B if the Eulerian derivative $\mathrm{d}J(\Omega;\vec{V})$ exists for all $\vec{V}\in B$ and the mapping $\vec{V} \to \mathrm{d}J(\Omega;\vec{V})$ is linear and continuous on B. An analogous definition can be introduced for functionals $J(\Gamma)$ depending on a $d-1$ manifold Γ as an independent variable.

We now recall a series of results from shape differential calculus in $\mathbb{R}^d$. We start with the shape derivative of domain and boundary integrals of functions not depending on the geometry.

Lemma 2.1 [24, Prop. 2.45]. *Let $\phi \in W^{1,1}(\mathbb{R}^d)$ and $\Omega \subset \mathbb{R}^d$ be open and bounded. Then the functional*

$$J(\Omega) = \int_\Omega \phi\,\mathrm{d}x \tag{23}$$

is shape differentiable. The shape derivative of J is given by

$$\mathrm{d}J(\Omega;\vec{V}) = \int_\Omega \mathit{div}\,(\phi\vec{V})\mathrm{d}x. \tag{24}$$

If $\Gamma = \partial\Omega$ is of class C^1 and $V = \vec{V}\cdot\vec{\nu}$, then

$$\mathrm{d}J(\Omega;\vec{V}) = \int_\Gamma \phi V \mathrm{d}s. \tag{25}$$

Lemma 2.2 [24, Prop. 2.50 and (2.145)]. *Let $\psi \in W^{2,1}(\mathbb{R}^d)$ and Γ be of class C^1. Then the functional*

$$J(\Gamma) = \int_\Gamma \psi \mathrm{d}S \tag{26}$$

is shape differentiable and

$$\begin{aligned}\mathrm{d}J(\Gamma;\vec{V}) &= \int_\Gamma \left(\nabla\psi\cdot\vec{V} + \psi \mathrm{div}_\Gamma \vec{V}\right)\mathrm{d}S \\ &= \int_\Gamma (\partial_\nu \psi + \psi\kappa) V \mathrm{d}S.\end{aligned} \tag{27}$$

Let us now consider more general functionals $J(\Omega)$, which are useful when we consider problems of optimal shape design for partial differential equations, like the one introduced in Section 1.2. In particular we are interested in computing sensitivities for functionals of the form

$$J(\Omega) = \int_\Omega \phi(x,\Omega)\mathrm{d}x, \quad \text{or} \quad J(\Gamma) = \int_\Gamma \psi(x,\Gamma)\,\mathrm{d}s, \tag{28}$$

where the functions $\phi(\cdot,\Omega):\Omega\to\mathbb{R}$ and $\psi(\cdot,\Gamma):\Gamma\to\mathbb{R}$ themselves depend on the geometric variables Ω and Γ, respectively. To handle the computation of the sensitivities of such functionals we need to take care of the derivatives of ϕ and ψ with respect to Ω and Γ.

First of all we recall the notion of *material derivative* $\dot{\phi}(\Omega;\vec{V})$ of ϕ at Ω in direction $\vec{V}$. It is defined as follows [24, Def. 2.71]:

$$\dot{\phi}(\Omega;\vec{V}) = \lim_{t\to 0}\frac{1}{t}(\phi(\Omega_t)\circ x(\cdot,t) - \phi(\Omega_0)), \tag{29}$$

where the mapping $x(\cdot,t)$ is defined as in (20). A similar definition holds for functions $\psi(\cdot,\Gamma)$ which are defined on boundaries Γ instead of domains Ω.

Let us now now recall the notion of *shape derivative* $\phi\prime(\Omega;\vec{V})$ of ϕ at Ω in the direction $\vec{V}$. It is defined to be [24, Def. 2.85]

$$\phi'(\Omega;\vec{V}) = \dot{\phi}(\Omega;\vec{V}) - \nabla\phi\cdot\vec{V}. \tag{30}$$

Accordingly, for boundary functions $\psi(\Gamma):\Gamma\to\mathbb{R}$, the shape derivative is defined to be [24, Def. 2.88]

$$\psi'(\Gamma;\vec{V}) = \dot{\psi}(\Gamma;\vec{V}) - \nabla_\Gamma\psi\cdot\vec{V}|_\Gamma. \tag{31}$$

With these notions we are able to calculate the Eulerian derivatives for the above shape functionals.

Proposition 2.1 [24, Sect. 2.31, 2.33]. *Let $\phi = \phi(\Omega,x)$ be so that the material derivative $\dot{\phi}(\Omega;\vec{V})$ and the shape derivative $\phi'(\Omega;\vec{V})$ exist. Then, the cost functional in (28) is shape differentiable and we have*

$$\mathrm{d}J(\Omega;\vec{V}) = \int_\Omega \phi'(\Omega;\vec{V})\mathrm{d}x + \int_\Gamma \phi V \mathrm{d}S. \tag{32}$$

For boundary functions $\varphi(\Gamma)$ we get

$$\mathrm{d}J(\Gamma;\vec{V}) = \int_\Gamma \varphi'(\Gamma;\vec{V})\mathrm{d}S + \int_\Gamma \kappa\varphi V \mathrm{d}S, \tag{33}$$

whereas if $\varphi(\cdot,\Gamma) = \psi(\cdot,\Omega)|_\Gamma$, then we obtain

$$\mathrm{d}J(\Gamma;\vec{V}) = \int_\Gamma \psi'(\Omega;\vec{V})|_\Gamma \mathrm{d}S + \int_\Gamma (\partial_\nu\psi + \kappa\psi)V\mathrm{d}S. \tag{34}$$

To use this Proposition we need to be able to compute the shape derivative of solutions $y(\Omega)$ to elliptic boundary value problems. We consider now a simple case, though sufficient for our later developments: let f, g, h be functions defined on $\mathbb{R}^d$, i.e. they do not depend on Ω, and let $y(\Omega)$ satisfy

$$\begin{aligned}&-\Delta y(\Omega) = f \quad \text{in } \Omega,\\ &y(\Omega) = g \quad \text{on } \Sigma, \qquad \partial_\nu y(\Omega) = h \quad \text{on } \Gamma.\end{aligned} \tag{35}$$

Lemma 2.3 ([26,24, Sect. 3.1 and 3.2]). *The shape derivative of $y(\Omega)$ in (35), $y' := y'(\Omega,\vec{V})$, satisfies the following boundary value problem*

$$\begin{cases} -\Delta y' &= 0, & \text{in } \Omega,\\ y' &= V\partial_\nu(g - y(\Omega)) & \text{on } \Sigma,\\ \partial_\nu y' &= \mathrm{div}_\Gamma(V\nabla_\Gamma y(\Omega)) + (\kappa h + \partial_\nu h + f)V & \text{on } \Gamma.\end{cases} \tag{36}$$

Let us conclude this part with a Riesz representation theorem, the Hadamard–Zolésio formula (3). We state the theorem without regularity assumptions and give below explicit expressions for G.

Lemma 2.4 ([26, Prop. 2.21,24, Sect 2.11 and Th. 2.27]). *The Eulerian derivative of a domain or boundary functional always has a representation of the form*

$$\mathrm{d}J(\Omega;\vec{V}) = \langle G, V\rangle_\Gamma, \tag{37}$$

where we denote by $\langle\cdot,\cdot\rangle_\Gamma$ a duality pair on Γ; that is, the Eulerian derivative is concentrated on Γ.

2.2. Shape derivatives of the model problems

For each shape functional introduced in Section 1.2 we compute the shape derivative and thereby obtain the explicit expressions (5), (8), and (12) of the Riesz representative G.

2.2.1. Image segmentation

According to Lemmas 2.1 and 2.2, we can write the shape derivative of $J(\Omega)$ as follows:

$$\mathrm{d}J(\Omega;V) = \int_\Gamma ((\kappa+\lambda)H(x) + \partial_\nu H(x))V\,\mathrm{d}S. \tag{38}$$

Then the shape gradient is

$$G = (\kappa+\lambda)H(x) + \partial_\nu H(x), \tag{39}$$

and has the form (13) with $g = H(x)$ and $f = \lambda H(x) + \partial_\nu H(x)$.

2.2.2. Optimal shape design for PDE

Let us now compute the shape derivative of the functional

$$J(\Omega, y(\Omega)) = J_1(\Omega, y(\Omega)) + J_2(\Omega) = \frac{1}{2}\int_D (y(\Omega) - z_g)^2 \mathrm{d}x + \gamma \int_\Gamma \mathrm{d}\Gamma, \tag{40}$$

where $y(\Omega)$ solves the elliptic problem (7).

Let us first consider the shape derivative $y' := y'(\Omega; \vec{V})$ at Ω in the direction $\vec{V}$, where we allow $\vec{V}$ to be nonzero only in a neighborhood of Γ (i.e. D and Σ are both assumed to be fixed). According to Lemma 2.3, y' is the solution of the following elliptic problem:

$$\begin{cases} -\Delta y' &= 0 & \text{in } \Omega, \\ y' &= -V\partial_\nu y = 0 & \text{on } \Sigma, \\ \partial_\nu y' &= \mathrm{div}_\Gamma(V\nabla_\Gamma y) + \kappa V & \text{on } \Gamma. \end{cases} \tag{41}$$

In order to relate the L^2-norm in $J_1(\Omega, y(\Omega))$ it is customary to introduce an adjoint problem

$$\begin{cases} -\Delta p &= \chi_D(y - z_g) & \text{in} \quad \Omega, \\ p &= 0 & \text{on} \quad \Sigma, \\ \partial_\nu p &= 0 & \text{in} \quad \Gamma, \end{cases} \tag{42}$$

whence

$$\begin{aligned} \mathrm{d}J_1(\Omega, V) &= \int_D (y - z_g)y' = -\int_\Omega \Delta p y' \\ &= \int_\Omega \nabla p \nabla y' - \int_\Gamma \partial_\nu p y' - \int_\Sigma \partial_\nu p y' \\ (y' = 0 \quad \text{on } \Sigma, \quad \partial_\nu p = 0 \quad \text{on } \Gamma) &= -\int_\Omega \Delta y' p + \int_\Gamma \partial_\nu y' p + \int_\Sigma \partial_\nu y' p \\ (p = 0 \quad \text{on } \Sigma) &= -\int_\Omega \Delta y' p + \int_\Gamma \partial_\nu y' p \\ &= \int_\Gamma (\mathrm{div}_\Gamma(V\nabla_\Gamma y) + \kappa V)p \\ &= \int_\Gamma (-\nabla_\Gamma y \nabla_\Gamma p + \kappa p)V. \end{aligned}$$

On using Lemma 2.2 we have

$$\mathrm{d}J_2(\Omega, V) = \gamma \int_\Gamma \kappa V \mathrm{d}\Gamma. \tag{43}$$

Hence Lemma 2.4 holds with the Riesz representation function

$$G = -\nabla_\Gamma y \nabla_\Gamma p + \kappa p + \kappa\gamma, \tag{44}$$

which has the form (13) with $g = p + \gamma$ and $f = -\nabla_\Gamma y \nabla_\Gamma p$.

2.2.3. Surface diffusion and epitaxially stressed solids

We now compute the shape derivative of the functional (11), namely,

$$J(\Omega, y(\Omega)) = J_1(\Omega, y(\Omega)) + J_2(\Omega) = \int_\Omega |\nabla y(\Omega)|^2 + \gamma \int_\Gamma \mathrm{d}S, \tag{45}$$

with $y(\Omega)$ satisfying (10). It follows from (36) that the shape derivative $y' := y'(\Omega; \vec{V})$ of $y(\Omega)$ satisfies

$$\begin{aligned} -\Delta y' &= 0 \quad \text{in } \Omega, \\ y' &= 0 \quad \text{on } \Sigma, \\ \partial_\nu y' &= \mathrm{div}_\Gamma(V\nabla_\Gamma y) \quad \text{on } \Gamma. \end{aligned} \tag{46}$$

Consequently, using (32), we obtain

$$\mathrm{d}J_1(\Omega; \vec{V}) = 2\int_\Omega \nabla y \nabla y' + \int_\Gamma |\nabla y|^2 V$$

and

$$\int_\Omega \nabla y \nabla y' = -\langle \Delta y, y' \rangle + \int_\Gamma y' \partial_\nu y = 0$$

because of (10). Since the shape derivative for $J_2(\Omega)$ obeys (43), we have thus derived the expression

$$\mathrm{d}J(\Omega; \vec{V}) = \int_\Gamma \left(|\nabla y|^2 + \gamma\kappa\right) V \,\mathrm{d}s.$$

This implies that the shape gradient G is

$$G = |\nabla y|^2 + \gamma\kappa$$

and has the form (13) with $g = \gamma$ and $f = |\nabla y|^2$.

When considering $b(\cdot,\cdot)$ as the $H^{-1}(\Gamma)$ inner product, the gradient flow, written in strong form, reads $V = \Delta_\Gamma(|\nabla y|^2 + \gamma\kappa)$, which coincides with Eq. (9). So in this case, the H^{-1} gradient flow corresponding to the functional (45) results in the surface diffusion flow, which had been previously derived from physics.

3. Discrete gradient flows

Now we are ready to answer the chief question (16) and provide a strategy to build a sequence $\{\Omega_n\}_{n\geqslant 0}$ such that $J(\Omega_{n+1}) \leqslant J(\Omega_n)$. We first consider in Section 3.1 an implicit time discretization, as proposed by Luckhaus [18] and Almgren et al. [1] for the Stefan problem with Gibbs–Thomson law. This technique is also related to the idea of minimizing movements first introduced by E. De Giorgi, see e.g. [2,3].

Since the resulting scheme is not practical, we then study an explicit time linearization in Section 3.2, followed in Section 3.3 by a semi-implicit time discretization which keeps geometric quantities such as velocity and curvature implicit but the rest of the geometry explicit.

3.1. Implicit time discretization

Let τ_n be a given variable time step and $t_{n+1} = t_n + \tau_n$. Before introducing the time discretization we observe that one step of the implicit Euler method for the heat equation coincides with the minimization of the functional

$$v \to \frac{1}{2}\int_\Omega |\nabla v|^2 + \frac{1}{2\tau_n}\int_\Omega |u_n - v|^2$$

(see [18,1]). Motivated by this, we let the domain Ω_{n+1} be the solution of the following penalized minimization problem:

$$\Omega_{n+1} = \operatorname{argmin}_{\Omega}\left(J(\Omega) + \frac{1}{2\tau_n} d^2(\Omega, \Omega_n)\right), \tag{47}$$

where the "distance term" $\frac{1}{2\tau_n} d^2(\Omega, \Omega_n)$ penalizes the distance between Ω and Ω_n. In order to specify the distance function $d(\cdot,\cdot)$, we consider $\vec{V}_{n+1} := \vec{V}(\vec{X}_{n+1})$ to be the *implicit* Euler approximation of (19):

$$\vec{X}_{n+1} = \vec{X}_n + \tau_n \vec{V}_{n+1}. \tag{48}$$

Note that Ω_{n+1} is described by the set of points $\vec{X}_{n+1}$ and that $\vec{V}_{n+1}$ is defined in Ω_{n+1}, so (48) does not specify $\vec{V}_{n+1}$ directly.

Let $V_{n+1} \in B(\Gamma_{n+1})$, where $(B(\Gamma_{n+1}), b_{n+1}(\cdot,\cdot), \|\cdot\|_{\Gamma_{n+1}})$ is a Hilbert space defined on the deformable part Γ_{n+1} of the boundary of Ω_{n+1}, thereby measuring the (boundary) smoothness of the vector fields. The natural choice

$$d(\Omega_{n+1}, \Omega_n) = \|\tau_n V_{n+1}\|_{B(\Gamma_{n+1})}$$

converts (47) into the following minimization problem:

$$\vec{V}_{n+1} = \operatorname{argmin}_{\vec{V}}\left(J(\Omega_n + \tau_n \vec{V}) + \frac{1}{2\tau_n}\|\tau_n V\|^2_{B(\Gamma_{n+1})}\right), \tag{49}$$

where $\vec{V} = V\vec{v}_{n+1}$. The optimality condition reads as follows:

$$\mathrm{d}J(\Omega_{n+1}; \tau_n \vec{W}) + \frac{1}{\tau_n} b_{n+1}(\tau_n V_{n+1}, \tau_n W) = 0, \quad \forall W \in B(\Gamma_{n+1}), \tag{50}$$

in terms of the variation $\vec{W} = W\vec{v}_{n+1}$ of $\vec{V}_{n+1}$. Such a condition, via Lemma 2.4, is equivalent to

$$\langle G_{n+1}, \tau_n W\rangle_{\Gamma_{n+1}} = -\frac{1}{\tau_n} b_{n+1}(\tau_n V_{n+1}, \tau_n W), \quad \forall W \in B(\Gamma_{n+1}).$$

This yields the following ideal equation for V_{n+1}

$$b_{n+1}(V_{n+1}, W) = -\langle G_{n+1}, W\rangle_{\Gamma_{n+1}}, \quad \forall W \in B(\Gamma_{n+1}), \tag{51}$$

which is *implicit* in that the domain Ω_{n+1} is unknown and thus part of the problem (see also [7]). A crucial consequence of (49) important for theory is

$$J(\Omega_{n+1}) \leqslant J(\Omega_n + \tau_n \vec{V}_{n+1}) + \frac{\tau_n}{2}\|V_{n+1}\|^2_{B(\Gamma_{n+1})} \leqslant J(\Omega_n) \tag{52}$$

as results from taking $\vec{V}_{n+1} = 0$ in (49). Consequently,

$$J(\Omega_k) + \frac{1}{2}\sum_{i=1}^{k} \tau_i \|\vec{V}_i\|^2_{B(\Gamma_i)} \leqslant J(\Omega_0), \quad \forall k \geqslant 1.$$

Solving the *implicit* nonlinear problem (51) is unaffordable directly and would require iteration. The following linearization technique may either replace the implicit solve or be used as one step in an iterative process.

3.2. Explicit linearization

The key idea is to replace in (51) the new domain Ω_{n+1} and its deformable boundary Γ_{n+1}, which are unknown, by the current ones Ω_n and Γ_n. This gives rise to the following linear elliptic PDE on Γ_n: find $V_{n+1} \in B(\Gamma_n)$ such that

$$b_n(V_{n+1}, W) = -\langle G_n, W\rangle_{\Gamma_n}, \quad \forall W \in B(\Gamma_n). \tag{53}$$

Then Ω_{n+1} results from the *explicit* update $\vec{X}_{n+1} = \vec{X}_n + \tau\vec{V}_{n+1}$, but the energy decrease property (52) is no longer valid. Nevertheless, (53) still provides a weaker energy decrease property. In fact, if one chooses $\vec{V}_{n+1} = V_{n+1}\vec{v}_n$, with V_{n+1} being the solution of

$$b_n(V_{n+1}, W) = -\langle G_n, W\rangle_{\Gamma_n}, \quad \forall W \in B(\Gamma_n), \tag{54}$$

then there holds that

$$\begin{aligned}\mathrm{d}J(\Omega_n; \vec{V}_{n+1}) &= \langle G_n, V_{n+1}\rangle_{\Gamma_n} = -b_n(V_{n+1}, V_{n+1}) \\ &\leqslant -\|V_{n+1}\|^2_{B(\Gamma_n)},\end{aligned} \tag{55}$$

that is $\vec{V}_{n+1}$ provides a descent direction for the energy $J(\Omega_n)$ (see [10]). However, (52) may not be valid, thereby leading to the bisection of τ_n (backtracking) until (52) holds. This is guaranteed by (55) which expresses the instantaneous decrease of energy.

3.3. Semi-implicit time discretization

To derive an effective algorithm, we still need to address two critical issues:

Computing geometric quantities such as curvature and normal velocity implicitly, but most of the geometry explicitly, thereby reaching a compromise between the schemes of Sections 3.1 and 3.2; (56)

Providing a variational method to compute curvature that could be used in unstructured meshes. (57)

To deal with (56) we let $\mathscr{B}_n$ denote the linear operator defined by the scalar product $b_n(\cdot,\cdot)$ on Γ_n, namely,

$$\langle \mathscr{B}_n V, W\rangle = b_n(V, W) \quad \forall V, W \in B(\Gamma_n).$$

Recalling the special form (13) of G, we thus propose the following *semi-implicit* computation of V_{n+1} and κ_{n+1}:

$$\mathscr{B}_n V_{n+1} + g(\Omega_n)\kappa_{n+1} = -f(\Omega_n). \tag{58}$$

This relation satisfies neither (52) nor (55) but tends to (15) for $\tau_n \to 0$. So backtracking must be employed to guarantee energy decrease.

To assess (57) we resort to basic differential geometry which asserts that the Laplace–Beltrami operator Δ_Γ of the position vector $\vec{X}$ of Γ is the vector mean curvature $\vec{\kappa}$ of Γ, namely,

$$-\Delta_\Gamma \vec{X} = \vec{\kappa} = \kappa\vec{v}. \tag{59}$$

Hereafter we use the minus sign to be consistent with the convention that a circle with outward unit normal $\vec{v}$ has positive curvature. The use of (59) for computation is due to Dziuk [13]. Since we need the scalar curvature κ, instead of $\vec{\kappa}$, we proceed as Bänsch et al. [6] and consider the four unknowns $\vec{\kappa}, \kappa, V, \vec{V}$ along with their algebraic relations:

$$\kappa = \vec{\kappa}\cdot\vec{v}, \quad \vec{V} = V\vec{v}. \tag{60}$$

If we take $\Gamma = \Gamma_{n+1}$, then for consistency with (58) we enforce (59) and (60) semi-implicitly

$$-\Delta_{\Gamma_n}\vec{X}_{n+1} = \vec{\kappa}_{n+1}, \quad \kappa_{n+1} = \vec{\kappa}_{n+1}\cdot\vec{\nu}_n, \quad \vec{V}_{n+1} = V_{n+1}\vec{\nu}_n.$$

Finally, to close the system we must relate position $\vec{X}_{n+1}$ of Γ_{n+1} and velocity $\vec{V}_{n+1}$. We impose

$$\vec{X}_{n+1} = \vec{X}_n + \tau_n\vec{V}_{n+1}, \tag{61}$$

whence we get the *semi-implicit scheme*: find $(\vec{\kappa}_{n+1}, \kappa_{n+1}, V_{n+1}, \vec{V}_{n+1})$, the solution on Γ_n of the following system of linear PDE:

$$\vec{\kappa}_{n+1} + \tau_n\Delta_{\Gamma_n}\vec{V}_{n+1} = -\Delta_{\Gamma_n}\vec{X}_n, \tag{62}$$

$$\kappa_{n+1} - \vec{\kappa}_{n+1}\cdot\vec{\nu}_n = 0, \tag{63}$$

$$\mathscr{B}_n V_{n+1} + g(\Omega_n)\kappa_{n+1} = -f(\Omega_n), \tag{64}$$

$$\vec{V}_{n+1} - V_{n+1}\vec{\nu}_n = 0. \tag{65}$$

3.4. Choosing the scalar product

Depending on the application of interest (e.g. image segmentation, optimal shape design for PDE or surface diffusion), there are several possibilities for the space $B(\Gamma)$ and the associated scalar product $b(\cdot,\cdot)$.

A first possibility is to choose $b(\cdot,\cdot)$ to coincide with the $L^2(\Gamma)$ scalar product. Then (64) takes the form

$$V_{n+1} + g(\Omega_n)\kappa_{n+1} = -f(\Omega_n), \tag{66}$$

which is a *backward–forward* parabolic-type equation, depending on the sign of the function g. In fact, (66) is locally backward (ill-posed) in regions where $g < 0$ and forward otherwise. This issue is crucial in optimal shape design for PDE, where the sign of $g = p + \gamma$ is unknown beforehand (see Section 4).

A second possibility is to choose $b(\cdot,\cdot)$ to coincide with a weighted $H^1(\Gamma)$ scalar product. In this case (64) reads

$$-\mathrm{div}_{\Gamma_n}(\alpha\nabla_{\Gamma_n}V_{n+1}) + \beta V_{n+1} + g(\Omega_n)\kappa_{n+1} = -f(\Omega_n), \tag{67}$$

where β and α are some positive functions. In Section 4 we will see that this choice (for suitable β's and α's) is well suited to stabilize the (ill-posed) L^2 gradient flow in the case of optimal shape design for PDE and it is helpful in increasing the rate of convergence of the numerical scheme in the case of image segmentation.

A third option is to choose $b(\cdot,\cdot)$ to coincide with the $H^{-1}(\Gamma)$ scalar product. If $g(\Omega_n) = \gamma$ and $f(\Omega_n) = |y(\Omega_n)|^2$, then (64) becomes

$$V_{n+1} - \gamma\Delta_{\Gamma_n}\kappa_{n+1} = \Delta_{\Gamma_n}f(\Omega_n). \tag{68}$$

This is the case of epitaxially stressed solids. Their dynamics is either described by the physical law (9) or by the H^{-1} gradient flow of the energy functional (11). In this application the choice of scalar product is thus dictated by consistency with the physics of the underlying phenomena.

To derive a weak formulation, we proceed as in the case of $H^{-1}(\Gamma)$; see [6] for details. For the weighted $H^1(\Gamma)$ scalar product, we multiply by suitable test functions, integrate by parts $-\mathrm{div}_{\Gamma_n}(\alpha\nabla_{\Gamma_n}V_{n+1})$, and ignore boundary terms either because Γ_n is closed or we assume Dirichlet boundary conditions: $-\langle\mathrm{div}_{\Gamma_n}(\alpha\nabla_{\Gamma_n}V_{n+1}), W\rangle = \langle\alpha\nabla_{\Gamma_n}V_{n+1}, \nabla_{\Gamma_n}W\rangle$. In the following we describe the finite element formulation for the weighted $H^1(\Gamma)$ case.

3.5. Finite element discretization

We now discuss the finite element discretization of (62)–(65). Let $\mathscr{T} = \mathscr{T}_n$ be a shape regular but possibly graded mesh of triangular finite elements over the surface $\Gamma = \Gamma_n$, which, from now on, is assumed to be polyhedral. We assume that Γ_n is defined to be a list of triangles (or segments), that is information on coordinates and connectivity. We stress that this is the minimal amount of info needed for planar domains. No explicit parametrization such as local or global charts are needed. To simplify the notations we hereafter drop the subscripts n and $n+1$. Let $T \in \mathscr{T}$ be a typical triangle and let $\vec{\nu}_T = (\nu_T^i)_{i=1}^d$ be the unit normal to T pointing outwards. We denote by $\vec{\nu}$ the outward unit normal to Γ, which satisfies $\vec{\nu}|_T = \vec{\nu}_T$ for all $T \in \mathscr{T}$, and is thus discontinuous across inter-element boundaries. Let $\{\phi_i\}_{i=1}^I$ be the set of canonical basis functions of the finite element space $\mathscr{V}(\Gamma)$ of continuous piecewise polynomials P^k of degree $\leqslant k$ over $\mathscr{T}$ for $k \geqslant 1$; we also set $\vec{\mathscr{V}}(\Gamma) := \mathscr{V}(\Gamma)^d$. We thus have a conforming approximation $\mathscr{V}(\Gamma)$ of $H^1(\Gamma)$.

We now multiply Eqs. (62)–(65) by test functions $\phi \in \mathscr{V}(\Gamma)$ and $\vec{\phi} \in \vec{\mathscr{V}}(\Gamma)$ and integrate by parts those terms involving Δ_Γ. We thus arrive at the fully discrete problem: seek $\vec{V}, \vec{\kappa} \in \vec{\mathscr{V}}(\Gamma)$, $V, \kappa \in \mathscr{V}(\Gamma)$, such that

$$\langle\vec{\kappa}, \vec{\phi}\rangle - \tau\langle\nabla_\Gamma\vec{V}, \nabla_\Gamma\vec{\phi}\rangle = \langle\nabla_\Gamma\vec{X}, \nabla_\Gamma\vec{\phi}\rangle, \quad \forall\vec{\phi} \in \vec{\mathscr{V}}(\Gamma), \tag{69}$$

$$\langle\kappa, \phi\rangle - \langle\vec{\kappa}\cdot\vec{\nu}, \phi\rangle = 0, \quad \forall\phi \in \mathscr{V}(\Gamma), \tag{70}$$

$$\langle\alpha\nabla_\Gamma V, \nabla_\Gamma\phi\rangle + \langle\beta V, \phi\rangle + \langle g\kappa, \phi\rangle = -\langle f, \phi\rangle, \quad \forall\phi \in \mathscr{V}(\Gamma), \tag{71}$$

$$\langle\vec{V}, \vec{\phi}\rangle - \langle V, \vec{\phi}\cdot\vec{\nu}\rangle = 0, \quad \forall\vec{\phi} \in \vec{\mathscr{V}}(\Gamma). \tag{72}$$

3.6. Matrix formulation

We turn our attention to an equivalent matrix formulation to the fully discrete problem. Given the matrix entries

$$M_{g_{i,j}} := \langle g\phi_i, \phi_j\rangle, \quad M_{\beta_{i,j}} := \langle\beta\phi_i, \phi_j\rangle, \quad M_{i,j} := \langle\phi_i, \phi_j\rangle,$$

$$\vec{M}_{i,j} := M_{i,j}\vec{Id}, \quad \vec{N}_{i,j} := (N_{i,j}^k)_{k=1}^d := \langle\phi_i, \phi_j\nu^k\rangle_{k=1}^d,$$

$$A_{i,j} := \langle\nabla_\Gamma\phi_i, \nabla_\Gamma\phi_j\rangle, \quad A_{\alpha i,j} := \langle\alpha\nabla_\Gamma\phi_i, \nabla_\Gamma\phi_j\rangle, \quad \vec{A}_{i,j} := A_{i,j}\vec{Id},$$

with $\vec{Id} \in \mathbb{R}^{d\times d}$ being the identity matrix and $(\vec{e}_k)_{k=1}^d$ the canonical basis of $\mathbb{R}^d$, the mass and stiffness matrices are

$$M_g := (M_{g_{i,j}})_{i,j=1}^I, \quad M_\beta := (M_{\beta_{i,j}})_{i,j=1}^I, \quad M := (M_{i,j})_{i,j=1}^I,$$

$$\vec{M} := (\vec{M}_{i,j})_{i,j=1}^I, \quad \vec{N} := (\vec{N}_{i,j})_{i,j=1}^I, \quad A_\alpha := (A_{\alpha i,j})_{i,j=1}^I,$$

$$A := (A_{i,j})_{i,j=1}^I, \quad \vec{A} := (\vec{A}_{i,j})_{i,j=1}^I.$$

We point out that $\vec{M}, \vec{A}$ and $\vec{N}$ possess matrix-valued entries and therefore the matrix–vector product is understood in the following sense:

$$\vec{M}\mathbf{V} = \left(\sum_{j=1}^{I} \vec{M}_{i,j}\vec{V}_j\right)_{i=1}^{I}, \tag{73}$$

each component $\vec{V}_i$ of $\vec{\mathbf{V}}$, as well as each of $\vec{M}\mathbf{V}$, is itself a vector in $\mathbb{R}^d$.

We use the convention that a vector of nodal values of a finite element function is written in bold face: $\mathbf{V} = (V_i)_{i=1}^{I} \in \mathbb{R}^I$ is equivalent to $V = \sum_{i=1}^{I} V_i\phi_i \in \mathscr{V}(\Gamma)$. We are now in a position to write the matrix formulation of (62)–(65). Upon expanding the unknown scalar functions $V, K \in \mathscr{V}(\Gamma)$ and vector functions $\vec{V}, \vec{K} \in \vec{\mathscr{V}}$ in terms of the basis functions and setting $\phi = \phi_i$ and $\vec{\phi}^k = \phi\vec{e}_k$, we easily arrive at the *linear* system of equations

$$\begin{aligned}
-\tau\vec{A}\vec{\mathbf{V}} + \vec{M}\vec{\mathbf{K}} &= \vec{A}\vec{\mathbf{X}},\\
M\mathbf{K} - \vec{N}^{\mathrm{T}}\vec{\mathbf{K}} &= \mathbf{0},\\
(A_\alpha + M_\beta)\mathbf{V} + M_g\mathbf{K} &= -\mathbf{f},\\
\vec{M}\vec{\mathbf{V}} - \vec{N}\mathbf{V} &= \vec{\mathbf{0}}.
\end{aligned} \tag{74}$$

3.7. Schur complement approach

In this section, following [6], we substitute variables until we arrive at a system of equations for the scalar velocity vector $\mathbf{V}$. The resulting system is what is actually solved in the simulations using an iterative algorithm, such as GMRES.

Let us rewrite the system (74) in the following way:

$$\begin{pmatrix} \vec{M} & 0 & 0 & -\vec{N} \\ 0 & M & -\vec{N}^{\mathrm{T}} & 0 \\ -\tau\vec{A} & 0 & \vec{M} & 0 \\ 0 & M_g & 0 & A_\alpha + M_\beta \end{pmatrix} \begin{pmatrix} \vec{\mathbf{V}} \\ \mathbf{K} \\ \vec{\mathbf{K}} \\ \mathbf{V} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vec{A}\vec{\mathbf{X}} \\ -\mathbf{f} \end{pmatrix}, \tag{75}$$

or equivalently, with obvious notation and $\mathbf{X}_1 = (\vec{\mathbf{V}}, \mathbf{K})^{\mathrm{T}}$, $\mathbf{X}_2 = (\vec{\mathbf{K}}, \mathbf{V})^{\mathrm{T}}$,

$$\begin{pmatrix} Z & N \\ C & \tilde{A} \end{pmatrix} \begin{pmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{pmatrix} = \begin{pmatrix} 0 \\ \mathbf{h} \end{pmatrix}. \tag{76}$$

Invoking the equalities

$$\mathbf{X}_1 = -Z^{-1}N\mathbf{X}_2, \tag{77}$$

$$(-CZ^{-1}N + \tilde{A})\mathbf{X}_2 = \mathbf{h}, \tag{78}$$

we see that (78) is equivalent to

$$\begin{pmatrix} \vec{M} & -\tau\vec{A}\vec{M}^{-1}\vec{N} \\ M_gM^{-1}\vec{N}^{\mathrm{T}} & A_\alpha + M_\beta \end{pmatrix} \begin{pmatrix} \vec{\mathbf{K}} \\ \mathbf{V} \end{pmatrix} = \begin{pmatrix} \vec{A}\vec{\mathbf{X}} \\ -\mathbf{f} \end{pmatrix}. \tag{79}$$

Finally the Schur complement reads as follows:

$$\begin{aligned}
&(\tau M_gM^{-1}\vec{N}^{\mathrm{T}}\vec{M}^{-1}\vec{A}\vec{M}^{-1}\vec{N} + A_\alpha + M_\beta)\mathbf{V}\\
&\quad = -\mathbf{f} - M_gM^{-1}\vec{N}^{\mathrm{T}}\vec{M}^{-1}\vec{A}\vec{\mathbf{X}}.
\end{aligned} \tag{80}$$

Invertibility of the Schur complement matrix is equivalent to solvability of (74). This is guaranteed for τ small. It is worth mentioning that the Schur complement matrix is never directly assembled. We assemble the individual sparse matrices M_g, M, $\vec{N}$, $\vec{M}$, $\vec{A}$, A_α, M_β; and write a matrix–vector product routine, which is then passed to GMRES as an argument. To solve the linear system $Ax = b$ iteratively, GMRES only requires the routine $\xi \to A\xi$. Inside the matrix–vector product routine, we never invert M or $\vec{M}$, but solve linear systems using conjugate gradient (CG) because M and $\vec{M}$ are symmetric and positive definite.

4. Numerical experiments

The numerical experiments presented here were implemented and carried out within the finite element toolbox ALBERTA of Schmidt and Siebert [22]. Curve or surface evolution of the deformable part Γ of the domain Ω is what drives the algorithm. To solve the linear system (74) and compute the vector velocity $\vec{\mathbf{V}}$, in order to update the mesh, we first solve for the scalar velocity $\mathbf{V}$ the system (80) by the iterative method GMRES, and finally compute $\vec{\mathbf{V}}$ from the last equation of (74). To solve the elliptic PDE in Ω, as in Sections 4.3 and 4.4.2, we invoked the 2D mesh generator TRIANGLE of Shewchuk [23], which partitions Ω into shape regular triangles and exhibits a superior performance with respect to mesh deformation techniques in 2D. The situation in 3D is quite different and needs to be explored further. Finally, we used GEOMVIEW [14] for visualization.

4.1. Implementation

It is typical of surface evolution undergoing large deformations that triangles may tangle and cross, and that their angles may become large. These mesh distortions limit resolution and approximability, as well as impair computations, thereby leading to numerical artifacts. We have resorted to a number of geometric enhancements as proposed by Bänsch et al. for surface diffusion [6]. An additional feature for curves in 2D is the capability of handling topological changes. We list these features below and briefly comment on them.

- *Mesh regularization*: This is a procedure to maintain mesh quality, namely to keep all angles on element stars approximately of the same size. Mesh regularization is a redistribution of nodes on the surface, which entails a tangential flow and does not affect the normal motion. We use the volume preserving Gauss–Seidel type iteration of [6].
- *Time adaptivity*: This is a procedure to allow large time steps when the normal velocity does not exhibit large variations, and to force small time steps when the change of position of nodes of an element may exceed the element size. This accounts for very disparate time

scales and prevent mesh distortion and even node crossing. Following [6], we impose a geometric restriction that limits the tangential motion of nodes: if $\mathbf{z}_0, \mathbf{z}_1$ are nodes belonging to a triangle T on Γ_n, and $\vec{\tau}_T$ is any unit tangent vector to T, then

$$\tau\left|\left(\vec{V}_{n+1}(\mathbf{z}_0) - \vec{V}_{n+1}(\mathbf{z}_1)\right) \cdot \vec{\tau}_T\right| \leqslant C\tau h_T\left|\nabla_\Gamma \vec{V}_{n+1}|_T\right| \leqslant \epsilon\tau h_T,$$

with $C, \epsilon > 0$ mesh independent constants.

- *Backtracking*: This simple procedure ensures energy decrease and improves significantly the algorithm performance. After the time step has been accepted according to the previous criterion we check that $J(\Omega_{n+1}) < J(\Omega_n)$. If this is not the case, we divide the time step by two and re-compute, repeating if necessary until the functional value is smaller than the previous one. The algorithm stops when the time step necessary for functional decrease is smaller than a pre-assigned minimum time step. There is still some room for improvement of this algorithm, but this is beyond the scope of this article.
- *Space adaptivity*: This procedure keeps, via refining/coarsening, an accurate representation of Γ_n with least computational cost in the sense that the node density correlates with the local variation (regularity) of Γ_n. We measure the latter intrinsically by monitoring the variation of normals $\vec{v}$. Since the pointwise accuracy of a mesh in representing a smooth surface is proportional to $h_S^2|\nabla_\Gamma \vec{v}|$, we impose [6]

$$h_S^2|\nabla_\Gamma \vec{v}| \approx h_S^2\frac{|\vec{v}_1 - \vec{v}_2|}{h_S} \approx \beta_S h_S \le \epsilon;$$

  here T_1, T_2 are two adjacent elements in Γ_n with unit normals $\vec{v}_1, \vec{v}_2$ and common side (node in 2D) S, β_S is the angle between $\vec{v}_1, \vec{v}_2$, and $\epsilon > 0$ is a given constant.
- *Angle width control*: This procedure consists of a single splitting (one bisection) of those elements with angles wider than a certain threshold $\beta_{\max}$, followed by a few mesh regularization sweeps [6]. This procedure is important close to pinch-off where mesh distorsion increases dramatically (see Fig. 10).
- *Topological changes in 2D*: This procedure is a set of algorithms that carry out topological changes, such as merging and splitting, in 2D; see [12] for details. At each time step, we check for element intersections that signal topological changes. If there are intersections, we adjust the time step and local resolution, reconnect elements at the intersection locations and delete superfluous elements if necessary. Level set methods can handle topological changes also in 3D and could be combined with our variational approach. This coupling is being investigated.

4.2. Image segmentation

In this section we perform numerical experiments with a number of synthetic images, by minimizing the geodesic active contour functional

$$J(\Omega) := \int_\Gamma H(x)\,\mathrm{d}S + \lambda\int_\Omega H(x)\,\mathrm{d}x.$$

The process starts with an initial curve in 2D or surface in 3D, which is iteratively deformed to decrease its energy at each step, and should terminate at the boundaries of the objects in the image.

4.2.1. Two gradient flows: L^2 vs weighted H^1

In our first experiment, we have a simple image with one connected, but nonconvex, object in it and we want to compare the L^2 gradient flow with the one resulting from the following weighted H^1 scalar product:

$$b(V, W) = \int_\Gamma \alpha\nabla_\Gamma V\nabla_\Gamma W + \beta VW, \tag{81}$$

where

$$\alpha = H, \quad \beta = \left(v\cdot D^2H\cdot v + (2\kappa+\lambda)\partial_v H + \lambda\kappa H\right)_+$$

and $(\cdot)_+ = \max(\cdot, \epsilon), \epsilon > 0$. This scalar product has been obtained in [17] by taking the second order shape derivative into account, thereby resulting in a Newton-type flow.

Both the L^2 and weighted H^1 Newton-type flows successfully detect the object, but the latter exhibits a smaller number of iterations and thus a higher rate of convergence (see Fig. 1). In this case the choice of the "good" scalar product has been essentially dictated by issues concerning stability and rate of convergence of the descent method.

In the rest of this Section 4.2 we present numerical experiments computed with the weighted H^1 scalar product (81). We should remark that the choice of threshold ϵ is a subtle issue. Hintermüller and Ring report in [17] that a small value suffices in general. Our experiments also show that $\epsilon = 0.1$ is well-behaved for a 100×100 image if we take unit interpixel distance, thereby giving rise to a computational domain $[0, 100]^2$. However, we prefer to take $\tilde{x} = 0.01x$ and rescale the domain to $[0, 1]^2$. This change of variables scales α by a factor 10^4 through D^2H, ∂H_v and κ. To preserve the time scale, ϵ must also be replaced with $\tilde{\epsilon} = (0.01)^{-2}\epsilon = 10^3$. This is the threshold used in our simulations.

4.2.2. Multiple objects

In this case the image has multiple objects in it. We start with a single closed curve that evolves and breaks into four smaller curves, which in turn eventually converge to the boundaries of the objects (see Fig. 2). The topological changes have been handled by using numerical tools developed in [12].

4.2.3. A simple 3D image

We finally consider a 3D image consisting of two spheres touching each other. The initial surface is a sphere that changes curvature as it evolves. This is reflected in the mesh grading (see Fig. 3).

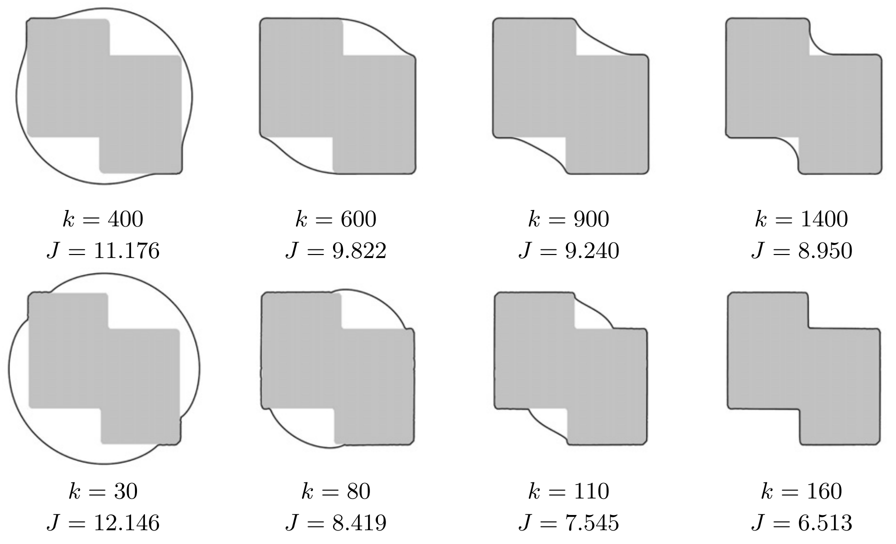

Fig. 1. Detection of a simple nonconvex object via the L^2 gradient descent (top) and weighted H^1 gradient descent (bottom) with $\lambda = 30$. The weighted H^1 flow is faster and more accurate than the L^2 gradient flow, as reflected in the number of iterations k, functional value J and resolution of reentrant corners.

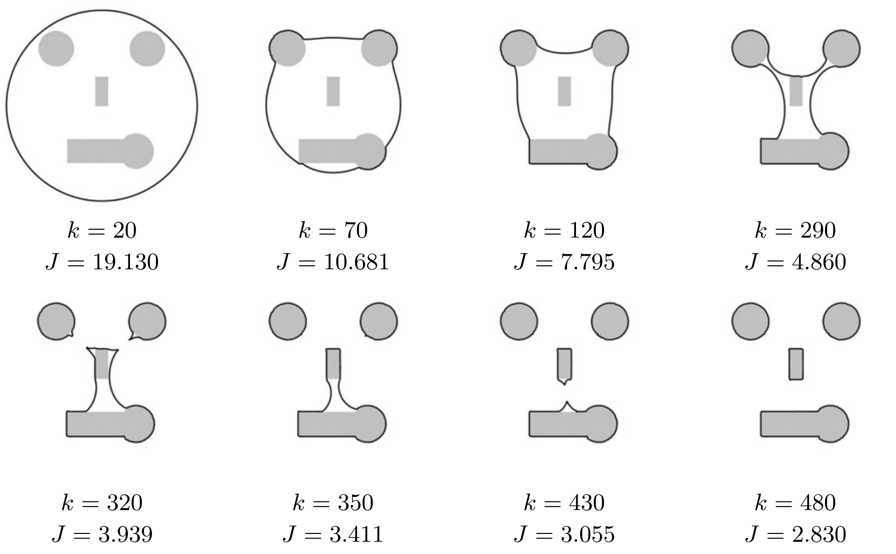

Fig. 2. Detection of multiple objects with weighted H^1 flow ($\lambda = 40$), showing splitting into several curves which recover all objects in the image.

4.3. Optimal shape design for PDE

In this section we present some numerical experiments for the model problem of Sections 1.2 and 2.2.2. Here, the energy functional reads

$$J(\Omega, y(\Omega)) := \frac{1}{2}\int_D \left(y(\Omega) - z_g\right)^2 \mathrm{d}x,$$

where z_g is a given *target* function on a subdomain D of Ω. The goal is to have the solution $y(\Omega)$ of (7) closest to z_g inside D in the least squares sense. We assume that both Ω and D are polygonal domains, and force the mesh generator TRIANGLE to match the boundary of D exactly when solving elliptic problems in Ω. This minimizes the quadrature error for the right-hand side evaluation in the equations for both $y = y(\Omega)$ and $p = p(\Omega)$.

4.3.1. The choice of scalar product

We perform simulations with a bilinear form $b(\cdot,\cdot)$ corresponding to a weighted H^1 scalar product, which gives rise to the elliptic PDE (67) on Γ_n:

$$-\mathrm{div}_\Gamma(\alpha\nabla_\Gamma V_{n+1}) + \beta V_{n+1} + g(\Omega_n)\kappa_{n+1} = -f(\Omega_n), \qquad (82)$$

where $g = p$, $f = -\nabla_\Gamma y \nabla_\Gamma p$, and the weight functions β and α were chosen appropriately.

The first attempt simply employs the $L^2(\Gamma)$ scalar product, which is obtained by taking $\alpha = 0$ and $\beta = 1$. In this case (82) reduces to

$$V_{n+1} + p\kappa_{n+1} = \nabla_\Gamma y \nabla_\Gamma p, \qquad (83)$$

which is a *backward–forward* parabolic-type equation, depending on the sign of the adjoint solution p. More precisely, (83) is locally backward parabolic (and so ill-posed)

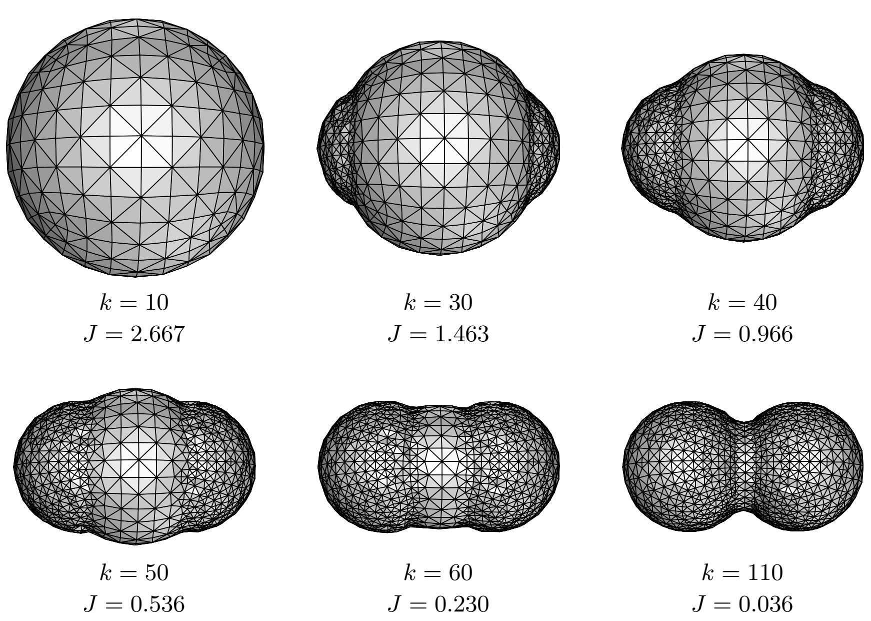

Fig. 3. Detection of a 3D object consisting of two touching balls with weighted H^1 flow ($\lambda = 0$). Notice the mesh grading depending on surface curvature.

in regions where $p < 0$ and forward otherwise. The ill-posedness of (83) gives rise to strong oscillations on the evolving curve at the mesh level, which can be observed in Fig. 4 where $p < 0$ (lower left). This ruled out the simple-minded option of L^2 flow because it is unstable.

The key idea behind the choice of an adequate scalar product is to set $\beta = 1$ on the whole curve Γ and take α variable to smooth out the evolution and thereby prevent oscillations. We thus take $\alpha = 1$ where $p < 0$ and α small where $p < 0$. In order to avoid spurious singularities in the velocity we make α vary smoothly from one element to a neighbor. This is reflected in the definition of α, which is constant on each edge e of Γ, but such constant depends on $p_M := \max_e p$ and $p_m := \min_e p$ as follows: we define the average $\bar{p} = \frac{1}{2}(p_M + p_m)$ and the oscillation $\text{osc} = p_M - p_m$ of p over e, and let $\alpha|_e$ be

$$\alpha|_e = \begin{cases} 1, & \text{if } p_m < 0, \\ 1 - \frac{\bar{p}(1-h_e^2)}{\text{osc}}, & \text{if } p_m \geqslant 0 \text{ and } \bar{p} \leqslant \text{osc}, \\ h_e^2, & \text{if } p_m \geqslant 0 \text{ and } \bar{p} > \text{osc}, \end{cases} \tag{84}$$

where h_e denotes the length of e. That is, in the transition region where p changes sign, $\alpha|_e$ is a sort of linear interpolation between $+1$ and h_e^2. The effective behavior of factor h_e^2 is to mimic an L^2 scalar product. Therefore, the resulting mesh dependent scalar product $b(\cdot,\cdot)$ combines L^2 and H^1 scalar products smoothly.

4.3.2. Test 1: exact solution

In this section we present an example with a known optimal shape. To build this example, we let $z_g = \log\frac{3}{|x|_2}$ be the exact solution of Laplace's equation on the ring

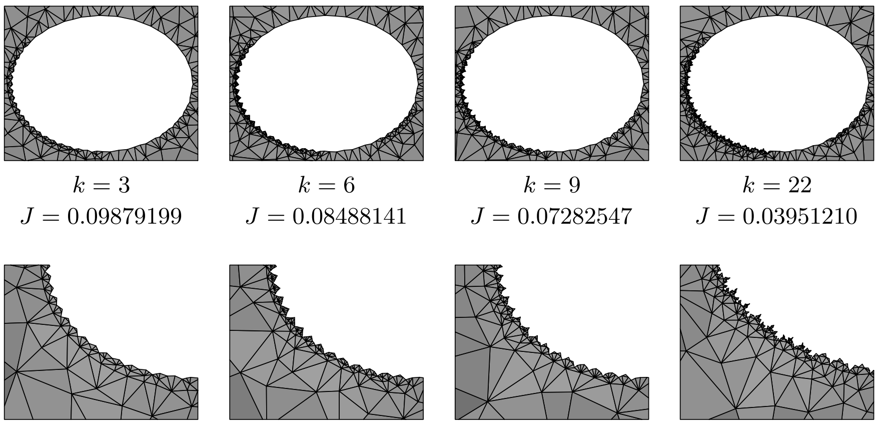

Fig. 4. L^2 gradient flow from a noncentered ellipse to a centered circle. The exact solution is a circle of radius one centered at the origin, and the initial configuration is a small ellipse centered at $(0.5, 0.7)$; see Fig. 5. The L^2 scalar product yields a dynamics which is locally stable where $p > 0$ (upper right) and locally unstable where $p < 0$ (lower left). The bottom row is a zoom of the unstable region exhibiting oscillations at the mesh level.

$\{1 < |x|_2 < 3\}$ with homogeneous Dirichlet data on $\{|x|_2 = 3\}$, and outer normal derivative equal to 1 on $\{|x|_2 = 1\}$. We let $D = \{2 \leqslant |x|_2 \leqslant 2.5\}$, and we point out that the global minimizer of $J(\Omega, y(\Omega))$ is $\Omega^* := \{1 < |x|_2 < 3\}$. The weighted H^1-scalar product with α as in (84) turns out to be a reasonable compromise between numerical stability and rate of convergence. We present a sequence of stable computations in Fig. 5 starting from a noncentered ellipse.

We also study the evolution from different initial configurations and observe that the algorithm always reaches a local minimum, but not necessarily the global minimizer Ω^*. In Fig. 6 we show the evolution obtained from an initial configuration consisting of two disjoint squares. This evolution leads to merging and stops at a local minimum different from the optimal configuration Ω^*. Nevertheless, the functional decreases several orders of magnitude which illustrates the flat energy landscape of $J(\Omega, y(\Omega))$.

An observation common to all simulations is that the reduction of $J(\Omega, y(\Omega))$, as well as the change of shape of Γ, is fast at the beginning but somewhat slow close to the optimal shape. This is typical of gradient flows.

4.3.3. Test 2: unknown solutions

We present here two examples with unknown minimizer. The first example consists of the same initial configuration and the same ring D as in the examples of Section 4.3.2, but the objective function z_g is now given in polar coordinates by

$$z_g(r, \theta) = 0.45 + 0.4\cos(6\theta).$$

That is, z_g is oscillates with respect to the angle θ. This experiment is to check whether the H^1 gradient flow is able to capture geometries other than circles.

In Fig. 7 we present snapshots of the approximating domains Ω_k together with values of the functional J_k. It is interesting to notice that the functional value does not decrease as drastically as in the previous examples. However, it is worth mentioning that the flow is able to capture the direction of energy decrease, even when such quantity is very small in relative terms. For example, between the fifth and sixth pictures, the energy decrease is just 0.03%, but the method is still able to detect how the curve should be modified to get an energy reduction.

Next, we consider a problem which resembles a real application. We take the starting domain to be

$$\Omega = \left\{x \in \mathbb{R}^2 : |x|_\infty < 3, \text{ and } |x|_2 > 0.5\right\},$$

and consider homogeneous Dirichlet boundary condition on the outer boundary, which is kept fixed, and the outer normal derivative equal to +1 in the “inner boundary”. The objective function is now $z_g \equiv 0.45$ in the domain $D := \{x \in \mathbb{R}^2 : 2.0 < |x|_\infty < 2.5\}$. Ideally the solution y should equal 0.45 in the (square) ring D, but this cannot happen with a harmonic function. We present a sequence of computations in Fig. 8 starting from a small centered circle (of radius 0.5). This circle evolves first into a bigger circle, and later into a smoothed-out square, yielding an energy reduction that goes from an initial value of 0.97481 to a final value of 0.19826. There is again a quick energy reduction at the beginning and a slower reduction in the end, which is typical of gradient flows.

4.4. Surface diffusion and epitaxially stressed solids

We present a couple of simulations exhibiting pinch-off in finite time, for the model problem of Sections 1.3 and 2.2.3:

$$J(\Omega, y(\Omega)) = \int_\Omega |\nabla y(\Omega)|^2 + \int_\Gamma \mathrm{d}S.$$

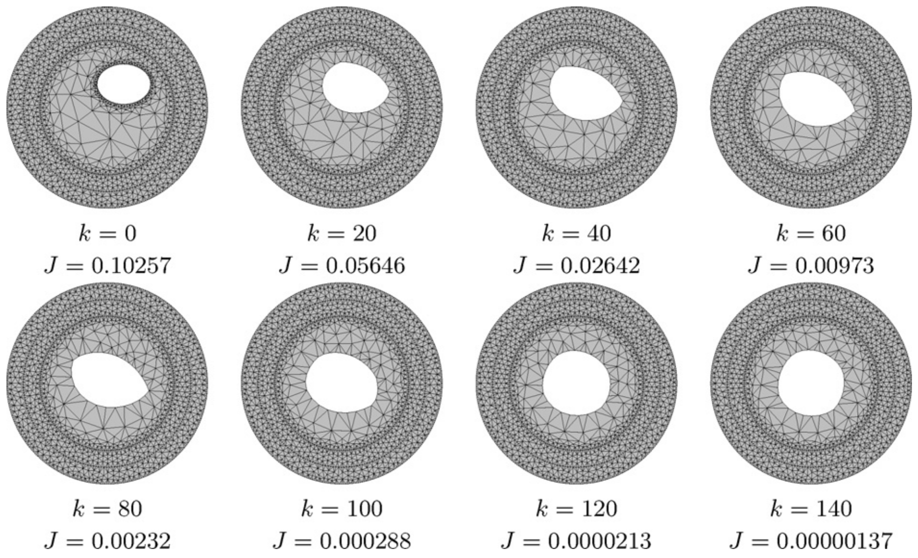

Fig. 5. Weighted H^1 flow from a noncentered ellipse to a centered circle. This dynamics is stable and efficient to detect a local minimizer.

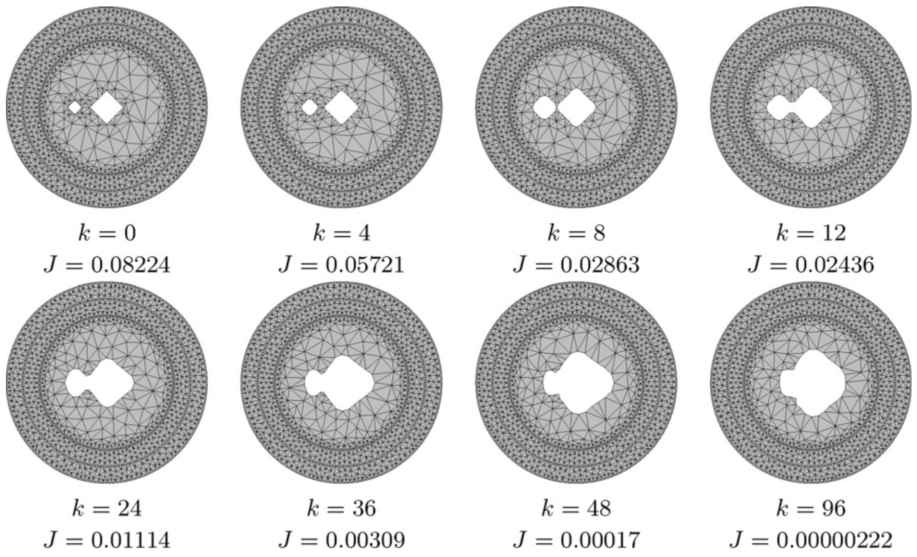

Fig. 6. Evolution of two disjoint squares merging via H^1 flow. The evolution stops at a local minimum before reaching the optimal circular configuration.

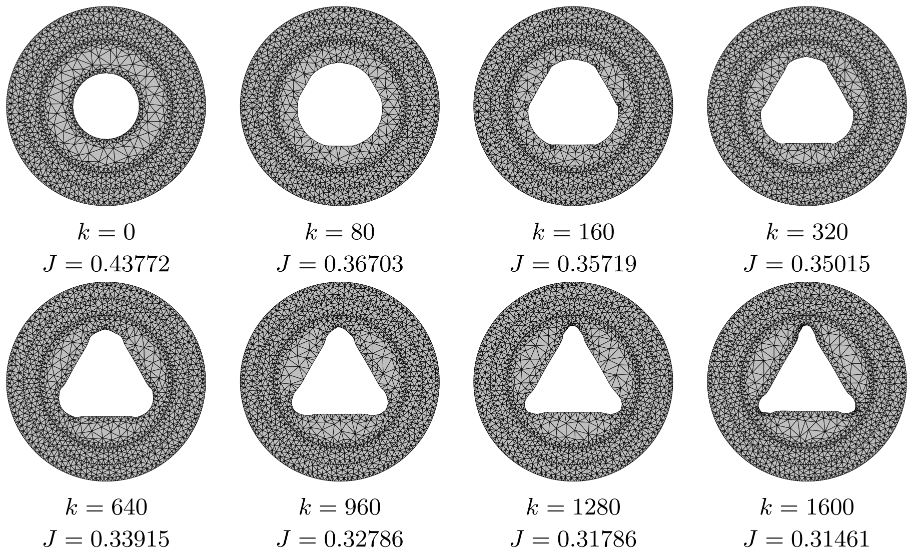

Fig. 7. Evolution of a small circle towards an unknown smoothed-out triangle via H^1 flow. The goal is to have the solution y as close as possible to an oscillating function z_g in the ring $D = \{x \in \mathbb{R}^2 : 2.0 < |x|_2 < 2.5\}$.

We first consider in Section 4.4.1 the pure geometric motion by surface diffusion, namely $y(\Omega) = 0$, and next the coupled problem in Section 4.4.2:

$$V = \Delta_\Gamma(\kappa + |\nabla_\Gamma y(\Omega)|^2).$$

We illustrate the use of space–time adaptivity as well as mesh smoothing and angle width control, explained in Section 4.1, to maintain mesh quality.

4.4.1. Test 1: surface diffusion with pinch-off

Let the initial surface $\Gamma(0)$ be an $8 \times 1 \times 1$ prism. This surface evolution is geometric and, even though it is regularizing, it leads to pinch-off in finite time as depicted in Fig. 9. This is a key example for the use of mesh smoothing and space–time adaptivity to avoid mesh distortion. However, close to the pinch-off some elements would tend to degenerate if it were not for the angle width control. Their combined effect is displayed in Fig. 10.

4.4.2. Test 2: formation of an inclusion

We now couple surface diffusion of the free surface of a film with the Laplace equation in the bulk, as explained in Section 1.3 and 2.2.3. We imposed the Dirichlet boundary condition $y = x$ on both the bottom and lateral boundary. The initial free surface $\Gamma(0)$ is a small cosine perturbation of the flat case, and its evolution $\Gamma(t)$ is periodic in x. We observe that $\Gamma(t)$ retains the graph property for a while

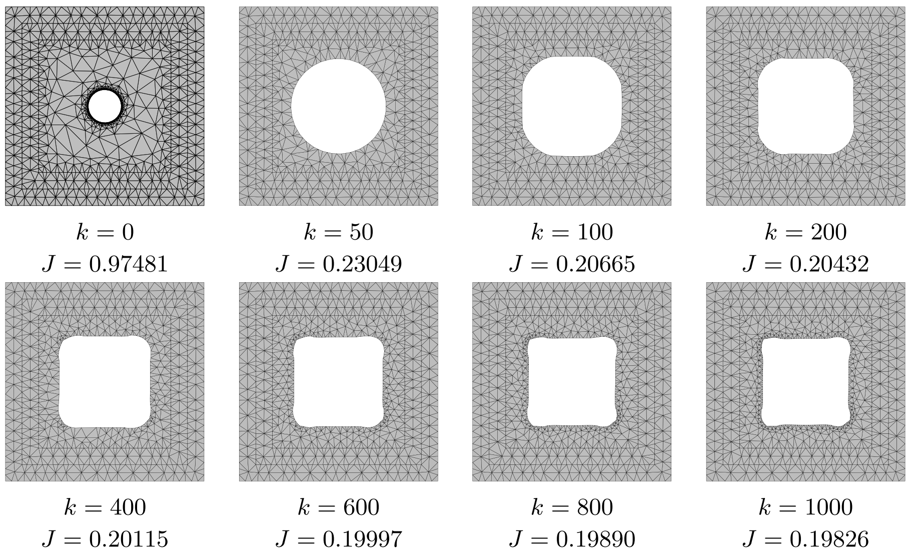

Fig. 8. Evolution of a small circle towards an unknown smoothed-out square via H^1 flow. The goal is to have the solution y as close to 0.45 as possible in the region $D = \{x \in \mathbb{R}^2 : 2.0 < |x|_\infty < 2.5\}$.

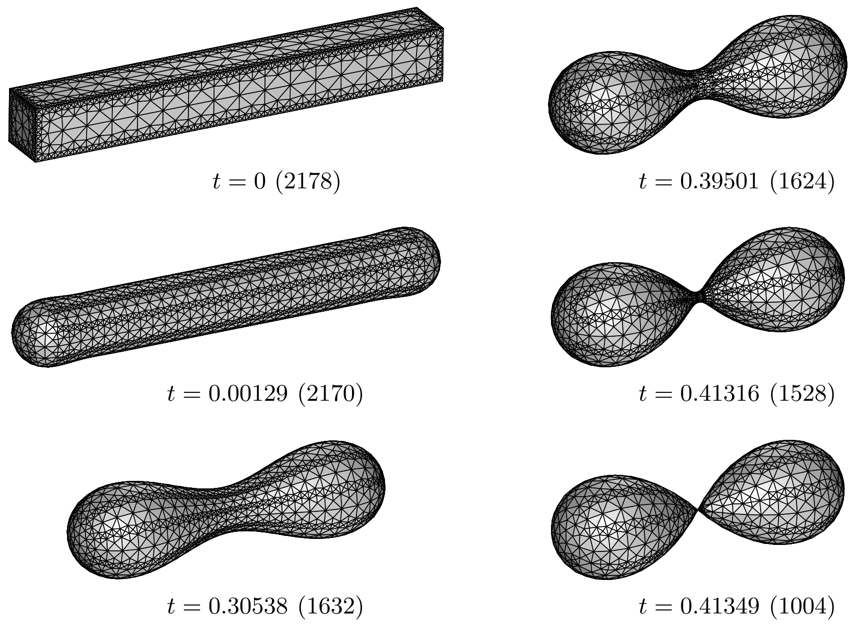

Fig. 9. Surface diffusion with pinch-off in finite time. Evolution of an $8 \times 1 \times 1$ prism at various time instants leading to a dumbbell and cusp formation (between parentheses we indicate the number of vertices used to represent the surface). The evolution was computed using time step control, mesh regularization, mesh refinement/coarsening, and a routine for controlling angle width (taken from [6]).

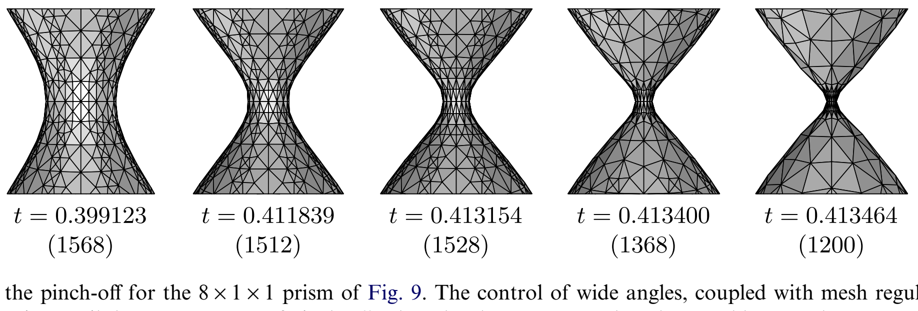

Fig. 10. Detailed view of the pinch-off for the $8 \times 1 \times 1$ prism of Fig. 9. The control of wide angles, coupled with mesh regularization, refinement and coarsening cure mesh distortion until the very moment of pinch-off, when the elements are rather elongated but not degenerate. An angle is considered to be wide when bigger than 120° (taken from [6]).

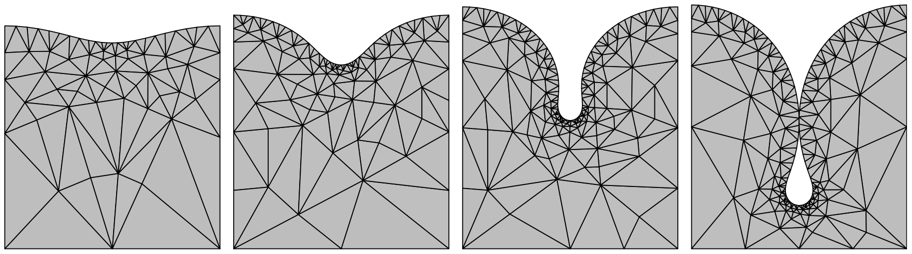

Fig. 11. Coupling surface diffusion with the Laplace operator in the bulk leads to a mushroom-like free surface that gives rise to an inclusion in finite time.

and eventually develops into a mushroom-like shape which closes up forming an insertion; see Fig. 11.

5. Conclusions

We presented a novel variational framework for numerically solving shape optimization problems. This framework is based on shape differential calculus, a semi-implicit time discretization and a finite element method for space discretization. Three examples were put into this framework: Image segmentation, optimal shape design for PDE, and surface diffusion. Some essential features of our approach are the following:

- Flexibility in choosing different descent directions by varying the scalar product used for the computation of normal velocity. We exploit this in applications either to provide efficiency (image segmentation, Section 4.2), stability (optimal shape design for PDE, Section 4.3), or to obtain an evolution consistent with physics (surface diffusion, Section 4.4).
- No need of an explicit parametrization of the deformable surface or curve. A description through vertex coordinates and element connectivity is sufficient.
- Well fitted for time and space adaptivity, mesh smoothing and backtracking, which are explored in the paper (see Section 4.1 and the simulations thereafter).
- Implicit (and therefore stable) computation of the curvature and normal velocity of the discrete surfaces without a CFL constraint. On the other hand level set-based approaches in shape optimization often handle these quantities explicitly in the time stepping scheme, hence in a less stable manner [7,17].
- Explicit handling of geometry (surface and normals), which give rise to a system of linear elliptic PDE on surfaces at every time step. A Schur complement approach is employed to solve the linear algebraic system for the scalar normal velocity at each time step.
- Capability to handle topological changes in 2D (see Sections 4.2 and 4.3). A hybrid variational/level set approach for 3D is being investigated.

Even though the presentation is formal, this work constitutes a concrete basis for the development and implementation of efficient numerical algorithms for general shape optimization problems. Further research includes dealing with PDE and topological changes in 3D.

Acknowledgements

We would like to thank E. Bänsch for allowing us to quote joint results from [6] in Section 4.4.1 and those of Section 4.4.2 developed in collaboration in Oberwolfach in March 2004 within the program *Research in Pairs*.

References

[1] F. Almgren, J.E. Taylor, L. Wang, Curvature-driven flows: a variational approach, SIAM J. Control Optim. 31 (2) (1993) 387–438.
[2] L. Ambrosio, Minimizing movements, Rend. Accad. Naz. Sci. XL Mem. Mat. Appl. 19 (5) (1995) 191–246.
[3] L. Ambrosio, N. Gigli, G. Savaré, Gradient Flows in Metric Spaces and in the Space of Probability Measures, Lectures in Mathematics ETH Zürich, Birkhäuser Verlag, Basel, 2005.
[4] R.J. Asaro, W.A. Tiller, Surface morphology development during stress corrosion cracking: Part i: via surface diffusion, Metall. Trans. 3 (1972) 1789–1796.
[5] G. Aubert, P. Kornprobst, Mathematical Problems in Image Processing, Applied Mathematical Sciences, vol. 147, Springer-Verlag, New York, 2002, Partial differential equations and the calculus of variations, with a foreword by Olivier Faugeras.
[6] E. Bänsch, P. Morin, R.H. Nochetto, A nite element method for surface diffusion: the parametric case, J. Comput. Phys. 203 (1) (2005) 321–343.
[7] M. Burger, A framework for the construction of level set methods for shape optimization and reconstruction, Interfaces Free Bound. 5 (3) (2003) 301–329.
[8] J. Cahn, J. Taylor, Surface motion by surface diffusion, Acta Metall. Mater. 42 (1994) 1045–1063.
[9] V. Caselles, R. Kimmel, G. Sapiro, Geodesic active contours, IJCV 22 (1) (1997) 61–79.
[10] J. Céa, Numerical methods for optimal shape design, in: E.J. Haug, J. Céa (Eds.), Proceedings of the NATO Advanced Study Institute on Optimization of Distributed Parameter Structural Systems, Martinus Nijho Publishers, 1981.
[11] M.C. Delfour, J.-P. Zolésio, Shapes and Geometries, Advances in Design and Control, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2001.
[12] G. Doğan, Topological changes and adaptivity for curve evolution, in preparation.
[13] G. Dziuk, An algorithm for evolutionary surfaces, Numer. Math. 58 (6) (1991) 603–611.
[14] Geometry Center at the University of Minnesota, Geomview, Interactive 3D Viewing Program. www.geomview.org.
[15] A. Henrot, M. Pierre, Variation et Optimisation de Formes, Mathematiques et Applications, vol. 48, Springer, 2005.

[16] A. Henrot, G. Villemin, An optimum design problem in magnetostatics, M2AN Math. Model. Numer. Anal. 36 (2) (2002) 223–239.
[17] M. Hintermüller, W. Ring, A second order shape optimization approach for image segmentation, SIAM J. Appl. Math. 64 (2) (2003/04) 442–467.
[18] S. Luckhaus, Solutions for the two-phase Stefan problem with the Gibbs–Thomson law for the melting temperature, Eur. J. Appl. Math. 1 (2) (1990) 101–111.
[19] B. Mohammadi, O. Pironneau, Applied Shape Optimization for Fluids, Numerical Mathematics and Scientific Computation, The Clarendon Press, Oxford University Press, Oxford Science Publications, New York, 2001.
[20] A. Novruzi, J.R. Roche, Newton's method in shape optimisation: a three-dimensional case, BIT 40 (1) (2000) 102–120.
[21] O. Pironneau, Optimal Shape Design for Elliptic Systems, Springer Series in Computational Physics, Springer-Verlag, New York, 1984.
[22] A. Schmidt, K. Siebert, Design of Adaptive Finite Element Software, Lecture Notes in Computational Science and Engineering, vol. 42, Springer-Verlag, Berlin, 2005, The finite element toolbox ALBERTA, With 1 CD-ROM (Unix/Linux).
[23] J.R. Shewchuk, Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator, in: M. Lin, D. Manocha (Eds.), Applied Computational Geometry: Towards Geometric Engineering, Lecture Notes in Computer Science, vol. 1148, Springer-Verlag, 1996.
[24] J. Sokołowski, J.-P. Zolésio, Introduction to Shape Optimization, Springer Series in Computational Mathematics, vol. 16, Springer-Verlag, Berlin, 1992, Shape sensitivity analysis.
[25] B.J. Spencer, S.H. Davis, P.W. Voorhees, Morphological instability in epitaxially-strained dislocation-free solid films: nonlinear evolution, Phys. Rev. B 47 (2) (1993) 9760–9777.
[26] J.-P. Zolésio, The material derivative (or speed method for shape optimization), in: E.J. Haug, J. Céa (Eds.), Proceedings of the NATO Advanced Study Institute on Optimization of Distributed Parameter Structural Systems, Martinus Nijhoff Publishers, 1981.
[27] P. Zunino, Multidimensional pharmacokinetic models applied to the design of drug eluting stents, J. Cardiovasc. Engrg. 4 (2) (2004).