Journal of Computational Physics **174,** 759–780 (2001)
doi:10.1006/jcph.2001.6937, available online at http://www.idealibrary.com on IDEAL®

Variational Problems and Partial Differential Equations on Implicit Surfaces

Marcelo Bertalmío

Electrical and Computer Engineering, University of Minnesota, Minneapolis, Minnesota 55455

Li-Tien Cheng and Stanley Osher

Mathematics Department, University of California, Los Angeles, California 90095

and

Guillermo Sapiro

Electrical and Computer Engineering, University of Minnesota, Minneapolis, Minnesota 55455
E-mail: guille@ece.umn.edu

A novel framework for solving variational problems and partial differential equations for scalar and vector-valued data defined on surfaces is introduced in this paper. The key idea is to implicitly represent the surface as the level set of a higher dimensional function and to solve the surface equations in a fixed Cartesian coordinate system using this new embedding function. The equations are then both intrinsic to the surface and defined in the embedding space. This approach thereby eliminates the need for performing complicated and inaccurate computations on triangulated surfaces, as is commonly done in the literature. We describe the framework and present examples in computer graphics and image processing applications, including texture synthesis, flow field visualization, and image and vector field intrinsic regularization for data defined on 3D surfaces.

Key Words: variational problems; partial differential equations; level set method; implicit surfaces; image processing; computer graphics; flow visualization; regularization; pattern formation; texture synthesis.

1. INTRODUCTION

In a number of applications, variational problems and partial differential equations need to be intrinsically solved for data defined on arbitrary manifolds, three-dimensional surfaces in particular. Examples of this exist in the areas of mathematical physics, fluid dynamics,

0021-9991/01 $35.00

image processing, medical imaging, computer graphics, and pattern formation. In computer graphics, examples of this include texture synthesis [63, 67], vector field visualization [20], and weathering [21]. In other numerous applications, data defined on surfaces often need to be regularized, e.g., as part of a vector field computation or interpolation process [49, 64], for inverse problems [27], or for surface parameterization [22]. These last regularization examples can be addressed by solving a variational problem defined on the surface, or its corresponding gradient-descent flow on the surface, using, for example, the well-developed theory of harmonic maps [24, 25], which has recently been demonstrated to be of use for image processing and computer graphics applications as well (e.g., [13, 22, 47, 54, 57, 69]). All these equations are generally solved on triangulated or polygonal surfaces. That is, the surface is given in polygonal (triangulated) form, and the data are discretely defined on it. Solving the problems then in this representation involves the nontrivial discretization of the equations in general polygonal grids, as well as the difficult numerical computation of other quantities such as projections onto the discretized surface (when computing gradients and Laplacians for example). Although the use of triangulated surfaces is extremely popular in all areas dealing with 3D models, especially in computer graphics, there is still no widely accepted technique for computing simple differential characteristics such as tangents, normals, principal directions, and curvatures; see, for example, [19, 41, 60] for a few of the approaches in this direction. On the other hand, it is widely accepted that computing these objects for iso-surfaces (implicit representations) is straightforward and much more accurate and robust. This problem in polygonal surfaces becomes even bigger when in addition to computes these first- and second-order differential characteristics of the surface we also have to use them to solve variational problems and PDEs for data defined on the surface. Moreover, virtually no analysis of numerical PDEs on nonuniform grids exists in the generality needed for the wide range of applications mentioned above, making it difficult to understand the behavior of the numerical implementation and its proximity (or lack thereof) to the continuous model.

In this paper we present a new framework for solving variational problems and PDEs for scalar and vector-valued data defined on surfaces. We use, instead of a triangulated/polygonal representation, an implicit representation: our surface will be the zero level set of a higher dimensional *embedding* function (i.e., a 3D volume with real values, positive outside the surface and negative inside it). Implicit surfaces have been widely used in many areas including computational physics [43], computer graphics [7, 28, 66], and image processing [52], as an alternative efficient representation to polygonal surfaces. We smoothly extend the original (scalar or vector-valued) data lying on the surface to the 3D volume, adapt our PDEs accordingly, and then perform all the computations on the fixed Cartesian grid corresponding to the embedding function. These computations are nevertheless intrinsic to the surface. The advantages of using the Cartesian grid instead of a triangulated mesh are many: we can use well-studied numerical techniques, with accurate error, stability, and robustness measures; the topology of the underlying surface is not an issue; and we can derive simple, accurate, robust, and elegant implementations. If the original surface is not already in implicit form, and it is for example triangulated, we can use any of a number of implicitation algorithms that achieve this representation given a triangulated input, e.g., [23, 37, 59, 68]. For example, the public domain software [38] can be used. If the data are just defined on the surface, an extension of them to the whole volume is also easily achieved using a PDE, as we will later see. Therefore, the method proposed here works as well for nonimplicit surfaces after the preprocessing is performed. This preprocessing

is quite simple and no complicated regridding needs to be done to go from one surface representation to another (see below). Finally, we will solve the variational problem or PDE only in a band surrounding the zero level set (a classical approach; see [46]). Therefore, although we will be increasing the dimension of the space by one, the computations remain of the same complexity, while significantly improved accuracy and greater simplicity are obtained.

1.1. *The Background and Our Contribution*

Representing *deforming* surfaces as level sets of higher dimensional functions was introduced in [43] as a very efficient technique for numerically studying the deformation (see [44] for a review of this technique and also [66] for studies on the deformation and manipulation of implicit surfaces for graphics applications). The idea is to represent the surface deformation via the embedding function deformation, which adds accuracy, robustness, and, as expected, topological liberty. When the velocity of the deformation is given by the minimization of an energy, the authors in [70] proposed a "variational level set" method, where they extended the energy (originally defined only on the surface) to the whole space. This allows the implementation to be made in the Cartesian grid. The key to this approach is to go from a "surface energy" to a "volume energy" by using a Dirac's delta function that concentrates the penalization on the given surface.

We will follow this general direction with our fixed, nondeforming surfaces. In our case, what is being deformed is the (scalar or vector-valued) data on the surface. If this deformation is given by an energy-minimization problem (as is the case in data smoothing applications), we will extend the definition of the energy to the whole 3D space, and its minimization will be achieved with a PDE, which, despite its being intrinsic to the underlying surface, is also defined in the whole space. Therefore, it is easily implementable. This is straightforward, as opposed to approaches where one maps the surface data onto the plane, performs the required operations there, and then maps the results back onto the triangulated representation of the surface or to approaches that attempt to solve the problem directly on a polygonal surface.

The new framework proposed here also tells us how to translate into surface terms PDEs that we know work on the plane but which do not necessarily minimize an energy (e.g., texture synthesis or flow visualization PDEs). Instead of running these PDEs on the plane and then mapping the results onto a triangulated representation of the surface, or running them directly on the triangulated domain, we obtain a 3D straightforward Cartesian grid realization that implements the equation intrinsically on the surface and has an accuracy that depends only on the degree of spatial resolution.

Moreover, we consider that for computing differential characteristics and solving PDEs even for triangulated surfaces, it might be appropriate to run an implicitation algorithm as any of the ones used for the examples in this paper and then work on the implicit representation. Current algorithms for doing this, some of them publicly available [38], are extremely accurate and efficient.

The contribution of this paper is therefore a new technique to efficiently solve a common problem in many computational physics and engineering applications: the implementation of variational problems and PDEs on 3D surfaces. In particular, we show how to transform any intrinsic variational or PDE equation into its corresponding one for implicit surfaces. In this paper we are then proposing a new framework to better solve existent problems and to

help in building up the solutions for new ones. To exemplify the technique and its generality, we implement and extend popular equations previously reported in the literature. Here we solve them with our framework, while in the literature they were solved with elaborated discretizations on triangulated representations.

Before proceeding, we should comment on a few of the basic characteristics of our framework. First, as stated before, although we solve the equations in the embedding space, the basic computational complexity of our technique is not increased, since all operations are performed on a narrow band surrounding the given surface. Second, since the work is now on a Cartesian grid, classical numerical analysis results on issues such as robustness and stability apply here as well. Note that for triangulated representations, new theoretical results are needed to justify the common methods proposed in the literature, while with our framework, classical and well-established numerical techniques can be made as accurate, robust, and computationally efficient as dictated by the application.

2. THE FRAMEWORK

2.1. *Surface and Data Representation*

As mentioned before, our approach requires us to have an implicit representation of the given fixed surface, and the data must be defined in a band surrounding them and not just on the surface. The implicit surfaces used in this paper have been derived from public-domain triangulated surfaces via the computation of a (signed) distance function $\psi(x, y, z)$ to the surface $\mathcal{S}$. Arriving at an implicit representation from a triangulated one is currently not a significant issue; there are publicly available algorithms that achieve it in a very efficient fashion. To exemplify this, in our paper we have used several of these techniques. For some surfaces the classical Hamilton–Jacobi equation $\|\nabla\psi\| = 1$ was solved on a predefined grid enclosing the given surface via the computationally optimal approach devised in [61]. Accurate implicit surfaces from triangulations of the order of 10^6 triangles are obtained in less than 2 min of CPU time with this technique. Alternatively, we used the implementation of the Closest Point Transform available in [38]. The teapot and knot surfaces were obtained from unorganized data points using the technique devised in [71]. We therefore assume from now on that the three-dimensional surface $\mathcal{S}$ of interest is given in implicit form as the zero level set of a given function $\psi : \mathbb{R}^3 \to \mathbb{R}$. This function is negative inside the closed bounded region defined by $\mathcal{S}$; it is positive outside and is Lipschitz continuous a.e., with $\mathcal{S} \equiv \{x \in \mathbb{R}^3 : \psi(x) = 0\}$. To ensure that the data, which need not be defined outside of the surface originally, are now defined in the whole band, one simple possibility is to extend the data u defined on $\mathcal{S}$ (i.e., the zero level set of ψ) in such a form that they are constant normal to each level set of ψ. This means that the extension satisfies $\nabla u \cdot \nabla\psi = 0$. (For simplicity, we now assume u to be a scalar function, although we will also address in this paper problems where the data defined on $\mathcal{S}$ are vector-valued. This is solved in an analogous fashion.) To solve this we numerically search for the steady-state solution of the Cartesian PDE

$$\frac{\partial u}{\partial t} + \text{sign}(\psi)(\nabla u \cdot \nabla\psi) = 0.$$

This technique was first proposed and used in [12]. Note that this keeps the given data u on the zero level set of ψ (the given surface) unchanged.

Both the implicitation and data extension (if required at all by the given data) need to be done only once off line. Moreover, they will remain for all applications that need this type of data.

2.2. *A Simple Example: Heat Flow on Implicit Surfaces*

We will exemplify our framework with the simplest case, the heat flow or Laplace equation for scalar data defined on a surface. For scalar data u defined on the plane, that is, $u(x, y) : \mathbb{R}^2 \to \mathbb{R}$, it is well known that the heat flow

$$\frac{\partial u}{\partial t} = \Delta u, \tag{1}$$

where $\Delta := \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$ is the Laplacian, is the gradient-descent flow of the Dirichlet integral

$$\frac{1}{2} \int_{\mathbb{R}^2} \|\nabla u\|^2 \, dx \, dy, \tag{2}$$

where ∇ is the gradient.

Equation (1) performs smoothing of the scalar data u, and this smoothing process progressively decreases the energy defined in Eq. (2). If we now want to smooth scalar data u defined on a surface $\mathcal{S}$, we must find the minimizer of the *harmonic* energy given by

$$\frac{1}{2} \int_{\mathcal{S}} \|\nabla_{\mathcal{S}} u\|^2 \, d\mathcal{S}. \tag{3}$$

The equation that minimizes this energy is its gradient-descent flow:

$$\frac{\partial u}{\partial t} = \Delta_{\mathcal{S}} u. \tag{4}$$

Here $\nabla_{\mathcal{S}}$ is the intrinsic gradient and $\Delta_{\mathcal{S}}$ is the intrinsic Laplacian or Laplace–Beltrami operator. These are classical concepts in differential geometry and basically mean the natural extensions of the gradient and Laplacian respectively, considering all derivatives intrinsic to the surface (with the natural metric). For instance, the intrinsic gradient is just the projection onto $\mathcal{S}$ of the regular 3D gradient while the Laplace–Beltrami operator is the projected divergence of it [53].

Classically, Eq. (4) would be implemented in a triangulated surface, giving rise to sophisticated and elaborated algorithms even for such simple flows. We now show how to simplify this when considering implicit representations.

Recall that $\mathcal{S}$ is given as the zero level set of a function $\psi : \mathbb{R}^3 \to \mathbb{R}$; ψ is negative inside the region bounded by $\mathcal{S}$ and is positive outside with $\mathcal{S} \equiv \{x \in \mathbb{R}^3 : \psi(x) = 0\}$. We proceed now to redefine the above energy and compute its corresponding gradient descent flow. Let $\mathbf{v}$ be a generic three-dimensional vector and let $P_{\mathbf{v}}$ be the operator that projects a given three-dimensional vector onto the plane orthogonal to $\mathbf{v}$:

$$P_{\mathbf{v}} := I - \frac{\mathbf{v} \otimes \mathbf{v}}{\|\mathbf{v}\|^2}. \tag{5}$$

It is then easy to show that the harmonic energy (3) is equivalent to (see, for example, [53])

$$\frac{1}{2}\int_{\mathcal{S}} \|P_{\mathbf{N}}\,\nabla u\|^2\, d\mathcal{S}, \tag{6}$$

where $\mathbf{N}$ is the normal to the surface $\mathcal{S}$. In other words, $\nabla_{\mathcal{S}} u = P_{\mathbf{N}}\nabla u$. That is, the gradient intrinsic to the surface ($\nabla_{\mathcal{S}}$) is just the projection onto the surface of the 3D Cartesian (classical) gradient ∇.[1] We now embed this in the function ψ to get

$$\frac{1}{2}\int_{\mathcal{S}} \|\nabla_{\mathcal{S}} u\|^2\, d\mathcal{S} = \frac{1}{2}\int_{\mathcal{S}} \|P_{\mathbf{N}}\nabla u\|^2\, d\mathcal{S} = \frac{1}{2}\int_{\Omega\in\mathbb{R}^3} \|P_{\nabla\psi}\nabla u\|^2 \delta(\psi)\|\nabla\psi\|\, dx,$$

where $\delta(\cdot)$ stands for the Dirac delta function, and all the expressions above are considered in the sense of distributions. Note that first we got rid of intrinsic derivatives by replacing $\nabla_{\mathcal{S}}$ by $P_{\mathbf{N}}\nabla u$ (or $P_{\nabla\psi}\nabla u$) and then we replaced the intrinsic integration ($\int_{\mathcal{S}} d\mathcal{S}$) by the explicit one ($\int_{\Omega\in\mathbb{R}^3} dx$) using the delta function. Intuitively, although the energy lives in the full space, the delta function forces the penalty to be effective only on the level set of interest. The last equality includes the embedding, and it is based on the following simple facts:

1. $\nabla\psi \parallel \mathbf{N}$.
2. $\int_{\Omega} \delta(\psi) \parallel \nabla\psi\parallel dx = \int_{\mathcal{S}} d\mathcal{S} =$ surface area.

In Appendix A we show that the gradient descent of this energy is given by

$$\frac{\partial u}{\partial t} = \frac{1}{\|\nabla\psi\|}\nabla\cdot(P_{\nabla\psi}\nabla u\|\nabla\psi\|). \tag{7}$$

In other words, this equation corresponds to the intrinsic heat flow or Laplace–Beltrami for data on an implicit surface. But all the gradients in this PDE are defined in the three-dimensional Cartesian space, not in the surface $\mathcal{S}$ (this is why we need the data to be defined at least on a band around the surface). The numerical implementation is then straightforward. This is the beauty of the approach! Basically, for this equation we use a classical scheme of forward differences in time and a succession of forward and backward differences in space (see Appendix B for details). The other equations in this paper are similarly implemented. This follows techniques such as those in [50]. Once again, due to the implicit representation and embedding in a Cartesian grid, classic numerical techniques are used, avoiding elaborate projections onto discrete surfaces and discretization on general meshes, e.g., [19, 32]. Classical numerical approaches and theoretical findings on robustness, accuracy, and error bounds apply then for our framework.

It is easy to show a number of important properties of this equation:

1. For any second embedding function $\phi = \phi(\psi)$, with $\phi(0) = 0$ and $\phi' \neq 0$, we obtain the same gradient descent flow. Since both ψ and ϕ have to share the zero level set, and we are only interested in the flow around this zero level set, this means that the flow is (locally) independent of the embedding function.[2]

[1] Note that, using this fact, we have transformed the computation of the norm of the intrinsic 2D gradient into an equivalent 3D Euclidean computation.

[2] We thank F. Mémoli for helping with this fact.

2. If ψ is the signed distance function, a very popular implicit representation of surfaces (obtained, for example, from the implicitation algorithms previously mentioned), the gradient descent simplifies to

$$\frac{\partial u}{\partial t} = \nabla \cdot (P_{\nabla\psi} \nabla u). \tag{8}$$

We have then obtained the basic approach for embedding intrinsic variational problems. We proceed now to embed general PDEs.

2.3. *From Variational Problems to PDEs*

We note that we could also have derived Eq. (7) directly from the harmonic maps flow,

$$\frac{\partial u}{\partial t} = \Delta_{\mathcal{S}} u,$$

via the simple geometry exercise of computing the Laplace–Beltrami $\Delta_{\mathcal{S}} u$ for $\mathcal{S}$ in implicit form (this is simply done by means of the projected derivatives as explained above, e.g., [53]). That is, the same equation is obtained when embedding the energy and then computing the gradient descent and when first looking at the gradient descent followed by the embedding of all of its components. This property is of particular significance. It basically hints at how to solve other PDEs, not necessarily gradient-descent flows, for data defined on implicit surfaces. All that we need to do is to recompute the components of the PDE for implicit representations of the surface. Note that in this way, conceptually, we can redefine classical planar PDEs on implicit surfaces, making them both intrinsic to the underlying surface and defined on the whole space.

2.4. *Anisotropic Diffusion on Implicit Surfaces*

From this very simple example on the Laplace–Beltrami flow we have seen the key point of our approach. If the process that we want to implement comes from the minimization of an energy, we derive a PDE for the whole space by computing the gradient-descent of the whole-space extension of that energy. Otherwise, given a planar PDE we recompute its components for an implicit representation of the surface. For instance, anisotropic diffusion can be performed on the plane by

$$\frac{\partial u}{\partial t} = \nabla \cdot \left(\frac{\nabla u}{\|\nabla u\|} \right), \tag{9}$$

which minimizes the TV energy $\int_{\mathbb{R}^2} \|\nabla u\| \, dx \, dy$ (see [50] and also [2, 6, 48] for related formulations). If we now want to perform intrinsic anisotropic diffusion of scalar data on a surface $\mathcal{S}$, we can either recompute the gradient-descent flow for the intrinsic TV energy $\int_{\mathcal{S}} \|\nabla_{\mathcal{S}} u\| \, d\mathcal{S}$, which for $\mathcal{S}$ in implicit form becomes $\int_{\Omega \in \mathbb{R}^3} \|P_{\nabla\psi} \nabla u\| \delta(\psi) \|\nabla \psi\| \, dx$, or just substitute into Eq. (9) the corresponding expressions as explained in the previous section. Either way we obtain the same result,

$$\frac{\partial u}{\partial t} = \frac{1}{\|\nabla \psi\|} \nabla \cdot \left(\frac{P_{\nabla\psi} \nabla u}{\|P_{\nabla\psi} \nabla u\|} \|\nabla \psi\| \right), \tag{10}$$

which is valid in the embedding Euclidean space. Note that general *p-harmonic maps*, that is, maps for L_p, $p \neq 2$, norms of the intrinsic surface gradient, have been studied in the literature as well, e.g., [13, 17, 29, 31, 57]. In the following section additional equations will be presented.

3. EXPERIMENTAL EXAMPLES

We now exemplify the framework just introduced for a number of important cases. The numerical implementation used is quite simple and requires a few lines of $C++$ code. The CPU time required for the diffusion examples is a few seconds on a PC (512 Mb RAM, 1 GHz) under Linux. For the texture synthesis examples, the CPU time ranges from a few minutes to one hour, depending on the pattern and parameters chosen. All the volumes used contain roughly 128^3 voxels. Note once again that due to the use of only a narrow band surrounding the zero level set, the order of the algorithmic complexity remains the same. On the other hand, the use of straightforward Cartesian numerics reduces the overall algorithmic complexity, improving accuracy and simplifying the implementation.

3.1. *Diffusion of Scalar Images on Surfaces*

The use of PDEs for image enhancement has become one of the most active research areas in image processing [11, 52]. In particular, diffusion equations are commonly used for image regularization, denoising, and multiscale representations (representing the image simultaneously at several scales or levels of resolution). This started with the works in [36, 65], where the authors suggested the use of the linear heat flow (1) for this task, where u represents the image gray values (the original image is used as initial condition). Note of course that this is the basic regularization needed for inverse problems defined on surfaces, e.g., [27]. By deriving the heat flow or Laplace–Beltrami equation on implicit surfaces we then derive the basic PDE used for image processing as well as the fundamental data regularization energy/flow. As we have seen, this flow is the gradient descent of (2), and the generalizations of these equations for data on the surface are given by (4) and (3) respectively. In implicit form, the heat flow on surfaces is given by (7). Figure 1 shows a simple example of image diffusion on a surface. Please note that this is *not* equivalent to performing 3D smoothing of the data and then looking to see what happened on $\mathcal{S}$. Our flow, though using extended 3D data, performs smoothing *directly* on the surface and is an

FIG. 1. Intrinsic isotropic diffusion. Left: original image. Middle: after 15 diffusion steps. Right: after 50 diffusion steps.

FIG. 2. Intrinsic anisotropic diffusion with constraints (automatic stop term). Left: original noisy image. Middle: after 50 diffusion steps. Right: after 90 diffusion steps. Notice how the diffusion stops and information is not smeared.

intrinsic heat flow (Laplace–Beltrami on the surface and not Laplace on the 3D space). The complete details of the numerical implementation of this flow are given in Appendix B (this will once again show how the implementation is significantly simplified with the framework described here).

In Fig. 2 we show an example for the anisotropic flow (10). In this case, we have a noisy image with known variance. We can then easily add this constraint to the flow and the corresponding variational formulation. The energy corresponding to this constraint is given by

$$\frac{\lambda}{2}\int_{\mathcal{S}}(u-u_0)^2\,d\mathcal{S}$$

($\lambda \in I\!R$ is a parameter and u_0 is the given noisy image), which after it is made intrinsic and implicit becomes

$$\frac{\lambda}{2}\int_{I\!R^3}(u-u_0)^2\delta(\psi)\,\|\nabla\psi\|\,dx.$$

To incorporate the constraint on the noise variance into the diffusion/denoising process, we add to the flow (10) the corresponding Euler–Lagrange of this energy, given by

$$\lambda(u-u_0).$$

Note in the figure how the noise is removed while the image details are preserved, as expected from an anisotropic flow. The parameter λ is estimated extending the technique suggested in [50] (see Appendix C).

The same approach, that of anisotropic diffusion with a stopping term given by the constraint, may be used to perform intrinsic *deblurring*; see [15].

We should note before proceeding that [35] also showed how to regularize images defined on a surface. The author's approach is limited to graphs (not generic surfaces) and only applies to level set based motions. The approach is simply to project the deformation of the data on the surface onto a deformation on the plane.

3.2. *Diffusion of Directional Data on Surfaces*

A particularly interesting example is obtained when we have unit vectors defined on the surface. That is, we have data of the form $u : \mathcal{S} \to S^{n-1}$. When $n = 3$ our unit vectors lie on

the sphere. Examples of this sort of data include principal directions (or general directional fields on 3D surfaces) and chromaticity vectors (normalized RGB vectors) for color images defined on the surface. This is also one of the most studied cases of the theory of harmonic maps due to its physical relationship with liquid crystals, and it was introduced in [57] for the regularization of directional data unit vectors on the plane (see also [13, 47, 54]). This framework of harmonic maps was used in computer graphics for texture mapping and surface parameterization, as pointed out earlier.

We still want to minimize an energy of the form

$$\int_{\mathcal{S}} \|\nabla_{\mathcal{S}} u\|^p \, d\mathcal{S},$$

though in this case $\nabla_{\mathcal{S}}$ is the vectorial gradient and the minimizer is restricted to be a unit vector. It is easy to show, e.g., [8, 55] that the gradient descent of this energy is given by the coupled system of PDEs

$$\frac{\partial u_i}{\partial t} = \mathrm{div}_{\mathcal{S}}(\|\nabla_{\mathcal{S}} u\|^{p-2} \nabla_{\mathcal{S}} u_i) + u_i \|\nabla_{\mathcal{S}} u\|^p, \quad 1 \le i \le n.$$

This flow guarantees that the initial unit vector $u(x, y, z, 0)$ remains a unit vector $u(x, y, z, t)$ all the time, thereby providing an equation for isotropic ($p = 2$) and anisotropic ($p = 1$) diffusion and regularization of unit vectors on a surface.

We can now proceed as before and embed the surface $\mathcal{S}$ into the zero level set of ψ, obtaining the following gradient-descent flows (see Appendix D):

$$\frac{\partial u_i}{\partial t} = \frac{1}{\|\nabla \psi\|} \nabla \cdot \left(\frac{P_{\nabla \psi} \nabla u_i}{\|P_{\nabla \psi} \nabla u\|^{2-p}} \|\nabla \psi\| \right) + u_i \|P_{\nabla \psi} \nabla u\|^p. \tag{11}$$

Note once again that although the regularization is done intrinsically on the surface, this equation only contains Cartesian gradients. An example of this flow for anisotropic diffusion of principal direction vectors is given in Fig. 3. On the top, we see the surface of a bunny with its correspondent vector field for the major principal direction. Any irregularity on the surface produces a noticeable alteration of this field, as can be seen in the details a and b. In the details a′ and b′, we see the result of applying the flow (11). Once again, the implementation of this flow with our framework is straightforward, while it would require very sophisticated techniques on triangulated surfaces (techniques that, in addition, are not supported by theoretical results).

Following also the work [57, 58] for color images defined on the plane, we show in Fig. 4 how to denoise a color image painted on an implicit surface. The basic idea is to normalize the RGB vector (a three-dimensional vector) to a unit vector representing the chroma and to diffuse this unit vector with the harmonic maps flow (11).[3] The corresponding magnitude, representing the brightness, is smoothed separately via scalar diffusion flows as those presented before (e.g., the intrinsic heat flow or the intrinsic anisotropic heat flow). That is, we have to regularize a map onto S^2 (the chroma) and another one onto $\mathbb{R}$ (the brightness).

[3] We renormalize at every discrete step of the numerical evolution to address deviations from the unit norm due to numerical errors [16]. We could also extend the framework in [1] and apply it to our equations.

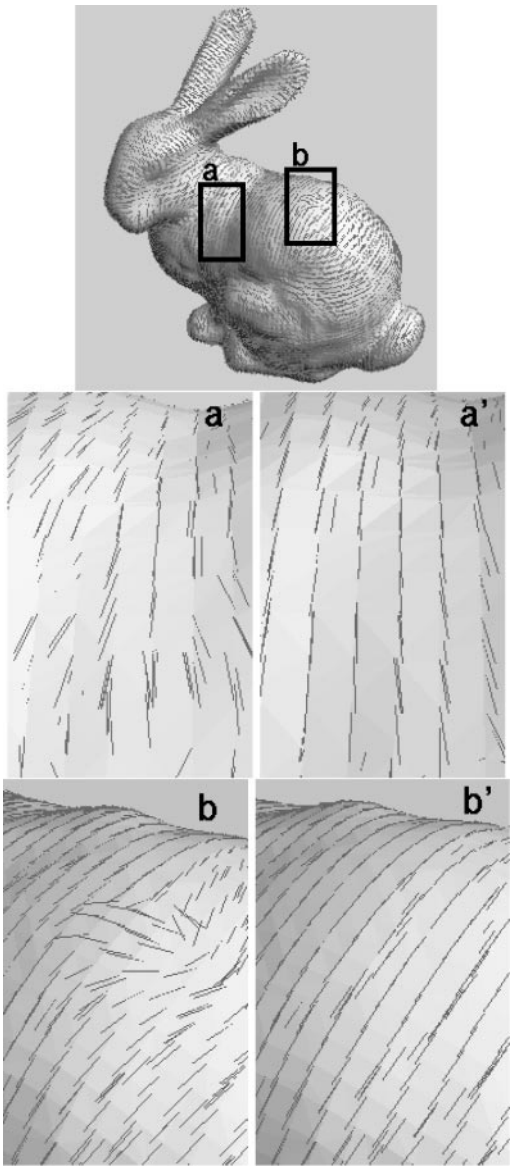

FIG. 3. Intrinsic vector field regularization. Top: original field of major principal direction of the surface. Details *a* and *b*: original field. Details a' and b': after anisotropic regularization.

3.3. *Pattern Formation on Surfaces via Reaction–Diffusion Flows*

The use of reaction–diffusion equations for texture synthesis became very popular in computer graphics following the works of Turk [63] and Witkin and Kass [67]. These works follow original ideas by Turing [62], who showed how reaction–diffusion equations can be used to generate patterns. The basic idea in these models is to have a number of "chemicals" that diffuse at different rates and that react with each other. The pattern is then synthesized by assigning a brightness value to the concentration of one of the chemicals. The authors in [63, 67] used their equations for planar textures and textures on triangulated surfaces. By

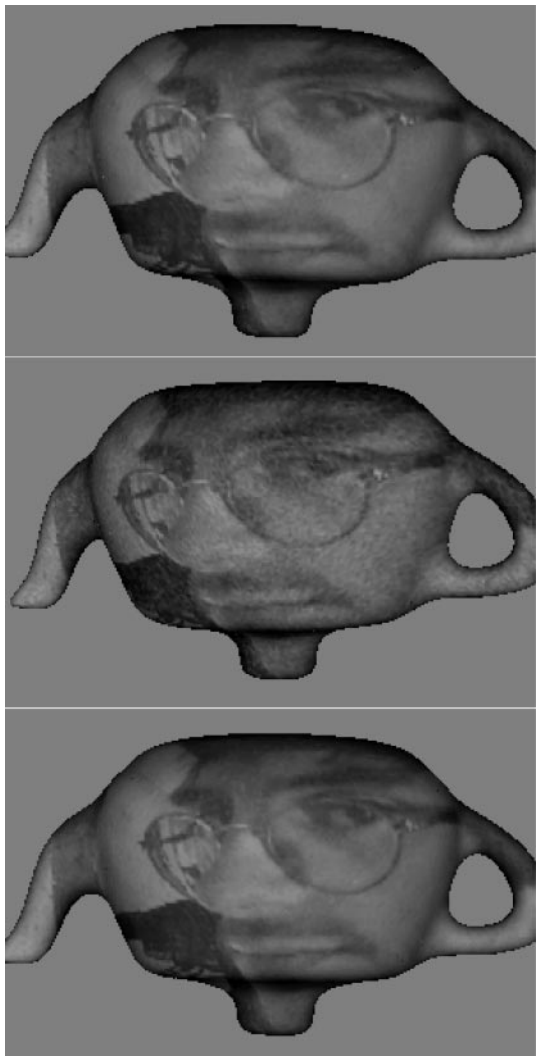

FIG. 4. Intrinsic vector field regularization. Top: original color image. Middle: heavy noise has been added to the three color channels. Bottom: color image reconstructed after 20 steps of anisotropic diffusion of the chroma vectors. Please see the journal website to view the color image.

using the framework described here, we can simply create textures on (implicit/implicitized) surfaces, without the elaborated schemes developed in those papers.[4]

Assuming a simple isotropic model with just two chemicals u_1 and u_2, we have

$$\frac{\partial u_1}{\partial t} = F(u_1, u_2) + D_1 \Delta u_1,$$
$$\frac{\partial u_2}{\partial t} = G(u_1, u_2) + D_2 \Delta u_1,$$

[4] Note that this is not the scheme proposed in [45], where the texture is created in the full 3D space. Here, the texture is created via reaction–diffusion flows intrinsic to the surface, whereas just the implementation is on the embedding 3D space.

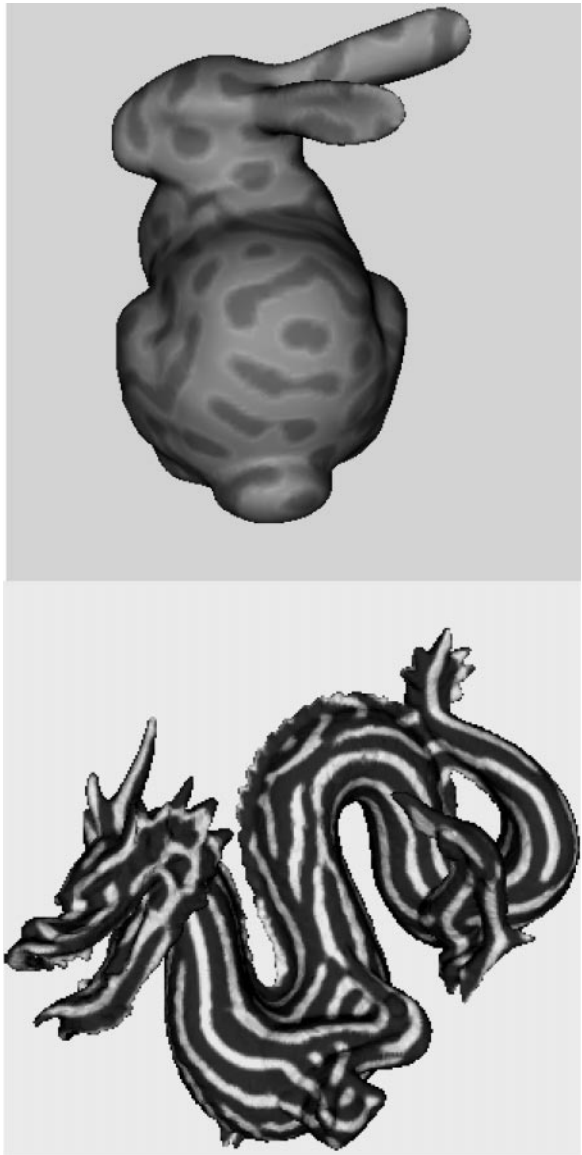

FIG. 5. Texture synthesis via intrinsic reaction–diffusion flows on implicit surfaces. Top: isotropic. Bottom: anisotropic. Pseudo-color representation of scalar data is used. The numerical values used in the computations were $D_1 = 1.0$, $D_2 = 0.0625$, $s = 0.025$, $\beta = 12.0 \pm 0.1$, and $u_1(0) = u_2(0) = 4.0$.

where D_1 and D_2 are two constants representing the diffusion rates and F and G are the functions that model the reaction.

Introducing our framework, if u_1 and u_2 are defined on a surface $\mathcal{S}$ implicitly represented as the zero level set of ψ we have

$$\frac{\partial u_1}{\partial t} = F(u_1, u_2) + D_1 \frac{1}{\|\nabla \psi\|} \nabla \cdot (P_{\nabla \psi} \nabla u_1 \|\nabla \psi\|), \tag{12}$$

$$\frac{\partial u_2}{\partial t} = G(u_1, u_2) + D_2 \frac{1}{\|\nabla \psi\|} \nabla \cdot (P_{\nabla \psi} \nabla u_2 \|\nabla \psi\|). \tag{13}$$

For simple isotropic patterns, Turk [63] selected

$$F(u_1, u_2) = s(16 - u_1 u_2),$$
$$G(u_1, u_2) = s(u_1 u_2 - u_2 - \beta),$$

where s is a constant and β is a random function representing irregularities in the chemical concentration. Examples of this, for implicit surfaces, are given in Fig. 5 (the coupled PDEs shown above are run until steady state is achieved). To simulate anisotropic textures, instead of using additional chemicals as in [63], we use anisotropic diffusion, as suggested in [67]. For this purpose, we replace Eq. (12) with

$$\frac{\partial u_1}{\partial t} = F(u_1, u_2) + D_1 \frac{1}{\|\nabla \psi\|} \nabla \cdot ((\mathbf{d} \cdot P_{\nabla \psi} \nabla u_1)\mathbf{d} \|\nabla \psi\|), \tag{14}$$

where $\mathbf{d}$ is a vector field tangent to the surface, e.g., the field of the major principal direction (which for our examples has been also accurately computed directly on the implicit surface, using the technique proposed in [40]). Note how this particular selection of the anisotropic reaction–diffusion flow direction provides a texture that helps the shape perception of the object. Additional patterns can be obtained with different combinations of the reaction and diffusion parts of the flow.

3.4. *Flow Visualization on 3D Surfaces*

Inspired by the work on line integral convolution [9] and that on anisotropic diffusion [48], the authors of [20] suggested using anisotropic diffusion to visualize flows in two and three dimensions. The basic idea is, starting from a random image, one anisotropically diffuses it in the directions dictated by the flow field. The authors presented very nice results both in two dimensions (flows on the plane) and three dimensions (flows on a surface), but once again using triangulated surfaces which introduce many computational difficulties.

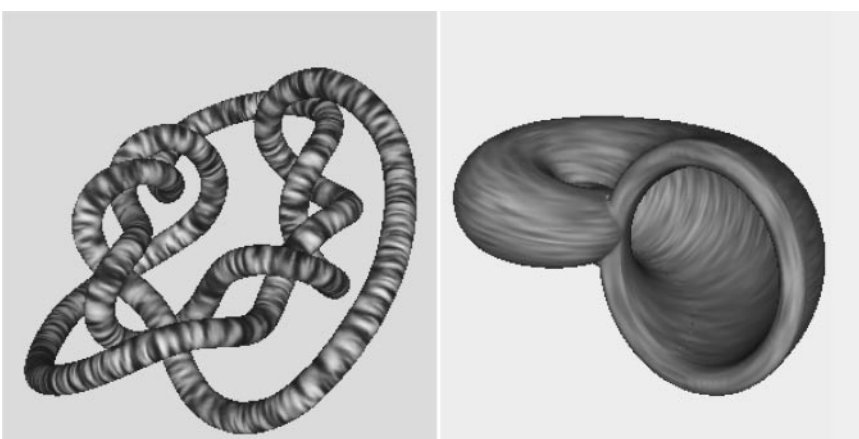

FIG. 6. Flow visualization on implicit 3D surfaces via intrinsic anisotropic diffusion flows. Left: flow aligned with the major principal direction of the surface. Right: flow aligned with the minor principal direction of the surface. Pseudo-color representation of scalar data is used.

In a straightforward fashion we can compute these anisotropic diffusion equations on the implicit surfaces with the framework introduced here, and some results are presented in Fig. 6. Note the complicated topology and how both the inside and outside parts of the surfaces are easily handled with our implicit approach. Also note that, when we choose the vector field to be that of one of the principal directions, the result emphasizes the surface shape.

4. CONCLUDING REMARKS

In this paper, we have introduced a novel framework for solving variational problems and PDEs for data defined on surfaces. The technique borrows ideas from the level set theory and the theory of intrinsic flows via harmonic maps. The surface is embedded in a higher dimensional function, and the Euler–Lagrange flow or PDE is solved in the Cartesian coordinate system of this embedding function. The equations are simultaneously intrinsic to the implicit surface and defined on the embedding Cartesian space. With this framework we enjoy accuracy, robustness, and simplicity, as expected from the computation of differential characteristics on iso-surfaces (implicit surfaces) and the use of classical and well-established numerical techniques in Cartesian grids. In addition to presenting the general approach, we have exemplified it with equations arising in image processing and computer graphics.

We believe that this new framework opens up a large number of theoretical and practical questions. In the theoretical arena, we need to extend the large amount of results available for harmonic maps (see, for example, [57] for a review on this) to the "implicit harmonic maps" equations introduced here. We would also like to investigate the effect of perturbations on the surface (zero level set) on the solutions of the intrinsic PDE. This is crucial to understanding the desired accuracy of surface implicitation algorithms. We expect that, as with the level set theory (e.g., [14, 26]), these theoretical results will follow after the presentation of the framework in this paper. On the practical side, we are currently addressing other related equations that appear in the mathematical physics, image processing, and computer graphics literature. For example, we are investigating how to extend the use of harmonic maps for texture mapping (and not just texture synthesis). This was done for triangulated surfaces in [3, 22, 30], and we plan to extend this to implicit surfaces via the implicit framework introduced here. We are also interested in investigating threshold dynamics and convolution-generated motions [34, 42, 51] for implicit surfaces. Other PDEs, such as those for image inpainting [4] or image segmentation [10], can be extended to work on implicit surfaces following the theory introduced here, and we would expect the same quality of results that were obtained on the plane. We could also use this framework to experimentally study results as those in [33]. Finally, the use of the approach presented here for regularization in inverse problems, e.g., [27], is of interest as well. These issues will be reported on elsewhere.

In conclusion, we should note that it is natural to ask about the target manifold for the most general form of harmonic maps, when this target is not just the Euclidean space or a unit ball, but a general hypersurface. In [39] we have shown how to extend the framework introduced here to arbitrary *implicit* target surfaces. Note also that general motion of curves on implicit surfaces is studied in [5, 15]. These works, together with the one here presented, provide then the basic framework for solving generic PDEs on implicit surfaces.

APPENDIX A

Heat Flow on Implicit Surfaces

Considering

$$E(u) := \frac{1}{2}\int_{\Omega\in I\!R^3} \|P_{\nabla\psi}\nabla u\|^2 \delta(\psi)\|\nabla\psi\|\,dx,$$

and with μ a perturbation of u, we have

$$\begin{aligned}
\frac{d}{dt}\Big|_{t=0} E(u+t\mu) &= \int_\Omega (P_{\nabla\psi}\nabla u \cdot P_{\nabla\psi}\nabla\mu)\delta(\psi)\|\nabla\psi\|\,dx \\
&= \int_\Omega \left(P_{\nabla\psi}\nabla u \cdot \left(\nabla\mu - \frac{\nabla\psi\cdot\nabla\mu}{\|\nabla\psi\|^2}\nabla\psi\right)\right)\delta(\psi)\|\nabla\psi\|\,dx \\
&= \int_\Omega (P_{\nabla\psi}\nabla u\cdot\nabla\mu)\delta(\psi)\|\nabla\psi\|\,dx \\
&\quad - \int_\Omega (P_{\nabla\psi}\nabla u\cdot\nabla\psi)\frac{\nabla\psi\cdot\nabla\mu}{\|\nabla\psi\|^2}\delta(\psi)\|\nabla\psi\|\,dx \\
&= \int_\Omega (P_{\nabla\psi}\nabla u\cdot\nabla\mu)\delta(\psi)\|\nabla\psi\|\,dx \\
&= -\int_\Omega \nabla\cdot(P_{\nabla\psi}\nabla u\delta(\psi)\|\nabla\psi\|)\mu\,dx \\
&= -\int_\Omega \nabla\cdot(P_{\nabla\psi}\nabla u\|\nabla\psi\|)\delta(\psi)\mu\,dx \\
&\quad - \int_\Omega (P_{\nabla\psi}\nabla u\cdot\nabla\psi)\delta'(\psi)\|\nabla\psi\|\mu\,dx \\
&= -\int_\Omega \nabla\cdot(P_{\nabla\psi}\nabla u\|\nabla\psi\|)\delta(\psi)\mu\,dx \\
&= -\int_{\mathcal{S}\equiv\{\psi=0\}} \frac{1}{\|\nabla\psi\|}\nabla\cdot(P_{\nabla\psi}\nabla u\|\nabla\psi\|)\mu\,d\mathcal{S}.
\end{aligned}$$

Since this expression has to be zero for all μ, we conclude that at the zero level set of ψ,

$$\frac{1}{\|\nabla\psi\|}\nabla\cdot(P_{\nabla\psi}\nabla u\|\nabla\psi\|) = 0,$$

and we make a natural extension to the whole domain Ω by considering this to hold on it.[5] We then obtain the gradient descent for the "implicit harmonic energy":

$$\frac{\partial u}{\partial t} = \frac{1}{\|\nabla\psi\|}\nabla\cdot(P_{\nabla\psi}\nabla u\|\nabla\psi\|). \tag{15}$$

[5] We have assumed that $\|\nabla\psi\| \neq 0$, at least on a band surrounding the zero level set. This assumption is valid since we can make the embedding function a distance function ($\|\nabla\psi\| = 1$), or we can simply multiply ψ by another function that guarantees that the zero level set $\mathcal{S}$ is preserved and that the gradient of the new embedding function is not zero.

APPENDIX B

Numerical Implementation of the Heat Flow on Implicit Surfaces

We now provide details on the numerical implementation of the intrinsic heat flow on implicit surfaces. All other equations reported in this paper are similarly implemented. Recall that all equations are on Cartesian grids, thereby permitting the use of classical numerics. We work on a cubic grid ($\Delta x = \Delta y = \Delta z = 1$), using an explicit scheme, where we compute the value of $u^n_{i,j,k} = u(i\Delta x,\, j\Delta y, k\Delta z, n\Delta t)$ based only on previous values ($t = (n-1)\Delta t$) of its neighbors, i.e., forward time differences.

First, we compute the 3D gradient of u using forward differences:

$$\mathbf{v}^n_{i,j,k} = \nabla_+ u^n_{i,j,k} = \left(u^n_{i+1,j,k} - u^n_{i,j,k}, u^n_{i,j+1,k} - u^n_{i,j,k}, u^n_{i,j,k+1} - u^n_{i,j,k}\right).$$

We compute the vector $\mathbf{N}(i, j, k)$, which gives the direction of the (outward) normal to the isosurface of ψ at the point (i, j, k):

$$\mathbf{N}_{i,j,k} = \nabla\psi_{i,j,k} = \frac{1}{2}(\psi_{i+1,j,k} - \psi_{i-1,j,k}, \psi_{i,j+1,k} - \psi_{i,j-1,k}, \psi_{i,j,k+1} - \psi_{i,j,k-1}).$$

Here we have used central differences. Note that, since ψ is fixed, $\mathbf{N}(i, j, k)$ does not change in time and we need only to compute it once. Its norm is $\|\mathbf{N}_{i,j,k}\| = (\sum_{m=1}^{3}(N_{i,j,k}[m])^2)^{\frac{1}{2}}$. The square brackets denote the components of the vector. Then we compute the intrinsic gradient, i.e., we project ∇u onto the plane normal to $\mathbf{N}$:

$$(P_{\mathbf{N}}\mathbf{v})^n_{i,j,k} = \mathbf{v}^n_{i,j,k} - \left(\frac{\sum_{m=1}^{3} N_{i,j,k}[m] \cdot v_{i,j,k}[m]}{\|\mathbf{N}_{i,j,k}\|^2}\right)\mathbf{N}_{i,j,k}.$$

Finally, we use a backward-differences implementation of the divergence:

$$\nabla_- \mathbf{w}_{i,j,k} = w_{i,j,k}[1] - w_{i-1,j,k}[1] + w_{i,j,k}[2] - w_{i,j-1,k}[2] + w_{i,j,k}[3] - w_{i,j,k-1}[3].$$

With forward time differences, the numerical implementation of the heat flow on implicit surfaces (Eq. (7)) is then

$$u^{n+1}_{i,j,k} = u^n_{i,j,k} + \Delta t\left[\frac{1}{\|\mathbf{N}_{i,j,k}\|}\nabla_- \cdot \left((P_{\mathbf{N}}\mathbf{v})^n_{i,j,k}\|\mathbf{N}_{i,j,k}\|\right)\right].$$

If the embedding function is a signed distance function, yielding Eq. (8), this expression for the numerical implementation of the intrinsic heat flow is simplified even further, making it virtually trivial.

APPENDIX C

Anisotropic Diffusion with Stopping Term on Implicit Surfaces

We have added an extra term to Eq. (10) so that the evolution stops by itself, given an estimate of the amount of noise that the original image has. The resulting PDE is

$$\frac{\partial u}{\partial t} = \frac{1}{\|\nabla\psi\|}\nabla \cdot \left(\frac{P_{\nabla\psi}\nabla u}{\|P_{\nabla\psi}\nabla u\|}\|\nabla\psi\|\right) - \lambda(u - u_0). \tag{16}$$

Assuming Gaussian noise with known variance σ^2 on the surface, $\sigma^2 = \int_{\Omega \in \mathbb{R}^3} (u - u_0)^2 \delta(\psi) \|\nabla \psi\| \, dx$, we need then to estimate the parameter λ. A way to do this, as suggested in [50], is the following. We merely multiply Eq. (16) by $(u - u_0)\delta(\psi)\|\nabla\psi\|$ and integrate by parts over Ω. If the steady state has been reached, the left side of Eq. (16) vanishes, and we have

$$\lambda = -\frac{1}{2\sigma^2} \int_{\mathcal{S}} (\nabla u - \nabla u_0) \cdot \frac{P_{\nabla\psi} \nabla u}{\|P_{\nabla\psi} \nabla u\|} \, d\mathcal{S}. \tag{17}$$

This gives us a dynamic value $\lambda(t)$.

APPENDIX D

Unit Norm Flows on Implicit Surfaces

We consider u to be data of the form $u : \mathcal{S} \to S^{n-1}$. For $n = 3$ we have unit vectors on the sphere: $u = (u_1, u_2, u_3)$, $\|u\| = 1$.

We define the norm of the vectorial gradient of u,

$$\|\nabla u\| = (\|\nabla u_1\|^2 + \|\nabla u_2\|^2 + \|\nabla u_3\|^2)^{\frac{1}{2}},$$

and also the norm of the intrinsic vectorial gradient,

$$\|P_{\nabla\psi} \nabla u\| = (\|P_{\nabla\psi} \nabla u_1\|^2 + \|P_{\nabla\psi} \nabla u_2\|^2 + \|P_{\nabla\psi} \nabla u_3\|^2)^{\frac{1}{2}}.$$

We want to minimize the energy

$$E_A(u) := \frac{1}{2} \int_{\Omega \in \mathbb{R}^3} \|P_{\nabla\psi} \nabla u\|^2 \delta(\psi) \|\nabla\psi\| \, dx,$$

with the constraint that u is of unit norm,

$$E_B(u) := \frac{1}{2} \gamma \int_{\Omega \in \mathbb{R}^3} (u^2 - 1) \delta(\psi) \|\nabla\psi\| \, dx,$$

so in practice we want to minimize the energy $E(u) = E_A(u) + E_B(u)$.

Applying to $E_A(u)$ the method described in Appendix A,

$$\begin{aligned} \left. \frac{d}{dt} \right|_{t=0} E(u + t\mu) = & \int_{\Omega} (P_{\nabla\psi} \nabla u_1 \cdot P_{\nabla\psi} \nabla \mu_1) \delta(\psi) \|\nabla\psi\| \, dx \\ & + \int_{\Omega} (P_{\nabla\psi} \nabla u_2 \cdot P_{\nabla\psi} \nabla \mu_2) \delta(\psi) \|\nabla\psi\| \, dx \\ & + \int_{\Omega} (P_{\nabla\psi} \nabla u_3 \cdot P_{\nabla\psi} \nabla \mu_3) \delta(\psi) \|\nabla\psi\| \, dx, \end{aligned}$$

we obtain the gradient descent for E_A,

$$\frac{\partial u_i}{\partial t} = \frac{1}{\|\nabla\psi\|} \nabla \cdot (P_{\nabla\psi} \nabla u_i \|\nabla\psi\|), \tag{18}$$

for each of the three components of u.
The gradient descent for E_B is simply

$$\frac{\partial u_i}{\partial t} = -\gamma u. \tag{19}$$

So the composed gradient descent, for E, is

$$\frac{\partial u_i}{\partial t} = \frac{1}{\|\nabla\psi\|}\nabla\cdot(P_{\nabla\psi}\nabla u_i\|\nabla\psi\|) - \gamma u_i. \tag{20}$$

We must find the value of γ. Multiplying both sides of Eq. (20) by u_i, making a summation over i, and bearing in mind that $\|u\| = 1$, we get

$$\gamma = \frac{1}{\|\nabla\psi\|}\Sigma u_i\nabla\cdot(\|\nabla\psi\|P_{\nabla\psi}\nabla u_i). \tag{21}$$

Using the equalities

$$u_i\nabla\cdot(\|\nabla\psi\|P_{\nabla\psi}\nabla u_i) = \nabla\cdot(u_i\|\nabla\psi\|P_{\nabla\psi}\nabla u_i) - \nabla u_i\cdot\|\nabla\psi\|P_{\nabla\psi}\nabla u_i,$$

$$u_i P_{\nabla\psi}\nabla u_i = u_i\left(\nabla u_i - \frac{\nabla\psi\nabla u_i}{\|\nabla\psi\|^2}\nabla\psi\right) = \frac{1}{2}\left(\nabla u_i^2 - \frac{\nabla\psi\nabla u_i^2}{\|\nabla\psi\|^2}\nabla\psi\right)$$

$$\nabla u_i\cdot\|\nabla\psi\|P_{\nabla\psi}\nabla u_i = \|\nabla\psi\|\nabla u_i\cdot P_{\nabla\psi}\nabla u_i = \|\nabla\psi\|\|P_{\nabla\psi}\nabla u_i\|^2$$

and the fact that

$$\Sigma\nabla u_i^2 = \nabla\left(\Sigma u_i^2\right) = \nabla(1) = 0, \tag{22}$$

we finally obtain

$$\gamma = -\|P_{\nabla\psi}\nabla u\|^2, \tag{23}$$

and so the gradient descent for E is

$$\frac{\partial u_i}{\partial t} = \frac{1}{\|\nabla\psi\|}\nabla\cdot(P_{\nabla\psi}\nabla u_i\|\nabla\psi\|) + u_i\|P_{\nabla\psi}\nabla u\|^2. \tag{24}$$

ACKNOWLEDGMENTS

We thank Facundo Mémoli for interesting conversations during this work. The implicitation software was provided by Santiago Betelu and H. Zhao. This work was partially supported by grants from the Office of Naval Research ONR-N00014-97-1-0509 and ONR-N00014-97-1-0027, the Office of Naval Research Young Investigator Award, the Presidential Early Career Awards for Scientists and Engineers (PECASE), a National Science Foundation CAREER Award, the National Science Foundation Learning and Intelligent Systems Program (LIS), NSF-DMS-9706827, ARO-DAAG-55-98-1-0323, an NDSEG Fellowship, and IIE-Uruguay.

REFERENCES

1. F. Alouges, An energy decreasing algorithm for harmonic maps, *in Nematics*, edited by Jean-Michel Coron, NATO ASI Series (Kluwer Academic, Dordrecht, Netherlands, 1991), pp. 1–13.

2. L. Alvarez, P. L. Lions, and J. M. Morel, Image selective smoothing and edge detection by nonlinear diffusion, *SIAM J. Numer. Anal.* **29**, 845 (1992).
3. S. Angenent, S. Haker, A. Tannenbaum, and R. Kikinis, *Laplace-Beltrami Operator and Brain Flattening*, Univ. of Minnesota ECE Report (Summer 1998).
4. M. Bertalmío, G. Sapiro, V. Caselles, and C. Ballester, Image inpainting, in *ACM Computer Graphics (SIGGRAPH 2000 Proc.)*, New Orleans, July 2000, pp. 417–424.
5. M. Bertalmío, G. Sapiro, and G. Randall, Region tracking on level set methods, *IEEE Trans. Med. Imaging* **18**, 448 (1999).
6. M. Black, G. Sapiro, D. Marimont, and D. Heeger, Robust anisotropic diffusion, *IEEE Trans. Image Proc.* **7**(3), 421 (1998).
7. J. Bloomenthal, *Introduction to Implicit Surfaces* (Morgan Kaufmann, San Francisco, 1997).
8. H. Brezis, J. M. Coron, and E. H. Lieb, Harmonic maps with defects, *Commun. Math. Phys.* **107**, 649 (1986).
9. B. Cabral and C. Leedom. Imaging vector fields using line integral convolution, *ACM Comput. Graphics (SIGGRAPH '93)* **27** (4), 263 (1993).
10. V. Caselles, R. Kimmel, and G. Sapiro, Geodesic active contours, *Int. J. Comput. Vision* **22** (1), 61 (1997).
11. V. Caselles, J. M. Morel, G. Sapiro, and A. Tannenbaum, Eds., Special issue on partial differential equations and geometry-driven diffusion in image processing and analysis, *IEEE Trans. Image Process.* **7**, March (1998).
12. S. Chen, B. Merriman, S. Osher, and P. Smereka, A simple level set method for solving Stefan problems, *J. Comput. Phys.* **135**, 8 (1995).
13. T. Chan and J. Shen, *Variational Restoration of Non-flat Image Features: Models and Algorithms*, UCLA CAM-TR 99-20 (June 1999).
14. Y. G. Chen, Y. Giga, and S. Goto, Uniqueness and existence of viscosity solutions of generalized mean curvature flow equations, *J. Differential Geom.* **33**, 749 (1991).
15. L. T. Cheng, *The Level Set Method Applied to Geometrically Based Motion, Material Science, and Image Processing*, Ph.D. dissertation (UCLA, June 2000).
16. R. Cohen, R. M. Hardt, D. Kinderlehrer, S. Y. Lin, and M. Luskin, Minimum energy configurations for liquid crystals: Computational results, in *Theory and Applications of Liquid Crystals*, edited by J. L. Ericksen and D. Kinderlehrer, IMA Volumes in Mathematics and Its Applications (Springer-Verlag, New York, 1987), pp. 99–121.
17. Y. Chen, M. C. Hong, and N. Hungerbuhler, Heat flow of p-harmonic maps with values into spheres, *Math. Z.* **205**, 25 (1994).
18. J. M. Coron and R. Gulliver, Minimizing p-harmonic maps into spheres, *J. Reine Angew. Math*. **401**, 82 (1989).
19. M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr, *Discrete differential-geometry operators in nD*, Multi-res Modeling Group TR (Caltech, September 2000; obtained from www.multires.caltech.edu).
20. U. Diewald, T. Preufer, and M. Rumpf, Anisotropic diffusion in vector field visualization on Euclidean domains and surfaces, *IEEE Trans. Visualization Comput. Graphics* **6**, 139 (2000).
21. J. Dorsey and P. Hanrahan, Digital materials and virtual weathering, *Sci. Am.* **282** (2), 46 (2000).
22. M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle. Multi-resolution analysis of arbitrary meshes, in *Computer Graphics (SIGGRAPH '95 Proc.)*, Los Angeles, 1995, pp. 173–182.
23. M. Eck and H. Hoppe, Automatic reconstruction of B-spline surfaces of arbitrary topological type, *Comput. Graphics (SIGGRAPH '96 Proc.)*, New Orleans, 1996, pp. 325–334.
24. J. Eells and L. Lemarie, A report on harmonic maps, *Bull. London Math. Soc.* **10**(1), 1 (1978).
25. J. Eells and L. Lemarie, Another report on harmonic maps, *Bull. London Math. Soc.* **20**(5), 385 (1988).
26. L. C. Evans and J. Spruck, Motion of level sets by mean curvature, I, *J. Differential Geom.* **33**, 635 (1991).
27. O. Faugeras, F. Clément, R. Deriche, R. Keriven, T. Papadopoulo, J. Gomes, G. Hermosillo, P. Kornprobst, D. Lingrad, J. Roberts, T. Viéville, and F. Devernay, *The inverse EEG and MEG Problems: The Adjoint State Approach I: The Continuous Case*, INRIA Research Report **3673** (June 1999).

28. S. F. Frisken, R. N. Perry, A. Rockwood, and T. Jones, Adaptively sampled fields: A general representation of shape for computer graphics, in *ACM Computer Graphics (SIGGRAPH 2000 Proc.)*, New Orleans, July 2000.
29. M. Giaquinta, G. Modica, and J. Soucek, Variational problems for maps of bounded variation with values in S^1, *Cal. Var.* **1**, 87 (1993).
30. S. Haker, S. Angenent, A. Tannenbaum, R. Kikinis, G. Sapiro, and M. Halle, *Conformal Surface Parameterization for Texture Mapping*, Univ. of Minnesota IMA Preprint Series Vol. **1611**, April (1999).
31. R. M. Hardt, "Singularities of harmonic maps," *Bull. Am. Math. Soc.* **34**(1), 15 (1997).
32. G. Huiskamp, Difference formulas for the surface Laplacian on a triangulated surface, *J. Comput. Phys.* **95**, 477 (1991).
33. T. Ilmanen, Generalized flows of sets by mean curvature on a manifold, *Indiana J. Math.* **4**, 671 (1992).
34. H. Ishii, A generalization of Bence, Merriman, and Osher algorithm for motion by mean curvature, in *Curvature Flows and Related Topics*, edited by A. Damlamian, J. Spruck, and A. Visintin (Gakkôtosho, Tokyo, 1995) pp. 111–127.
35. R. Kimmel, Intrinsic scale space for images on surfaces: The geodesic curvature flow, *Graph. Models Image Proces.* **59**, 365 (1997).
36. J. J. Koenderink, The structure of images, *Biol. Cybernet.* **50**, 363 (1984).
37. V. Krishnamurthy and M. Levoy, Fitting smooth surfaces to dense polygon meshes, *Comput. Graphics (SIGGRAPH '96 Proc.)*, New Orleans, 1996, p. 313.
38. S. Mauch, Closest point transform, www.ama.caltech.edu/∼seanm/software/cpt/cpt.html.
39. F. Mémoli, G. Sapiro, and S. Osher, Variational problems and PDE's onto arbitrary target surfaces, University of Minnesota, IMA preprint, (November 2001).
40. O. Monga, S. Benayoun, and O. Faugeras, From partial derivatives of 3D density images to ridge lines, in *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, (1992), pp. 354–359.
41. H. P. Moreton and C. H. Séquin, Functional minimization or fair surface design, in *Computer Graphics (SIGGRAPH'92 Proc.)*, 1992, pp. 167–176.
42. B. Merriman, J. Bence, and S. Osher, Diffusion generated motion by mean curvature, in *Computational Crystal Growers Workshop*, edited by J. E. Taylor (American Mathematical Society, Providence, 1992), pp. 73–83.
43. S. J. Osher and J. A. Sethian, Fronts propagation with curvature dependent speed: Algorithms based on Hamilton–Jacobi formulations, *J. Comput. Phys.* **79**, 12 (1988).
44. S. J. Osher and R. P. Fedkiw, *Level Set Methods*, UCLA CAM Report **00-07** (February 2000), *J. Comput. Phys*, to appear.
45. K. Perlin, An image synthesizer, *Comput. Graphics* **19**, 287 (1985).
46. D. Peng, B. Merriman, S. Osher, H. Zhao, and M. Kang, A PDE-based fast local level set method, *J. Comput. Phys.* **155**, 410 (1999).
47. P. Perona, Orientation diffusion, *IEEE Trans. Image Process.* **7**, 457 (1998).
48. P. Perona and J. Malik, Scale-space and edge detection using anisotropic diffusion, *IEEE Trans. Pattern. Anal. Machine Intell.* **12**, 629 (1990).
49. E. Praun, A. Finkelstein, and H. Hoppe, Lapped textures, in *ACM Computer Graphics (SIGGRAPH 2000 Proc.)*, New Orleans, July 2000, pp. 465–470.
50. L. I. Rudin, S. Osher, and E. Fatemi, Nonlinear total variation based noise removal algorithms, *Physica D* **60**, 259 (1992).
51. S. J. Ruuth, B. Merriman, and S. Osher, Convolution generated motion as a link between cellular automata and continuum pattern dynamics, *J. Comput. Phys.* **151**, 836 (1999).
52. G. Sapiro, *Geometric Partial Differential Equations and Image Processing* (Cambridge Univ. Press, New York, 2001).
53. L. Simon, *Lectures on Geometric Measure Theory* (Australian National Univ., Canberra, 1984).
54. N. Sochen, R. Kimmel, and R. Malladi, A general framework for low level vision, *IEEE Trans. Image Process.* **7**, 310 (1998).

55. M. Struwe, On the evolution of harmonic mappings of Riemannian surfaces, *Comment. Math. Helvetici* **60**, 558 (1985).
56. M. Struwe, *Variational Methods* (Springer-Verlag, New York, 1990).
57. B. Tang, G. Sapiro, and V. Caselles, Diffusion of general data on non-flat manifolds via harmonic maps theory: The direction diffusion case, *Int. J. Comput. Vision* **36**(2), 149 (2000).
58. B. Tang, G. Sapiro, and V. Caselles, Chromaticity diffusion, *IEEE Trans. Image Process.* May, 701 (2001).
59. G. Taubin, Estimation of planar curves, surfaces, and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation, *IEEE Trans. Pattern Analysis Machine Intelligence* **13**(11), 1115 (1991).
60. G. Taubin, Estimating the tensor of curvature of a surface from a polyhedral approximation, in *IEEE Int. Conf. Computer Vision*, Boston, MA, 1995, pp. 902–907.
61. J. N. Tsitsiklis, Efficient algorithms for globally optimal trajectories, *IEEE Trans. Automatic Control* **40**, 1528 (1995).
62. A. Turing, The chemical basis of morphogenesis, *Philos. Trans. R. Soc. B* **237**, 37 (1952).
63. G. Turk, Generating textures on arbitrary surfaces using reaction–diffusion, *Comput. Graphics* **25**(4), 289 (1991).
64. G. Winkenbach and D. H. Salesin, Rendering parametric surfaces in pen and ink, in *Computer Graphics (SIGGRAPH '96)*, New Orleans, 1996, pp. 469–476.
65. A. P. Witkin, Scale-space filtering, *Int. Joint Conf. Artificial Intelligence* **2**, 1019 (1983).
66. A. Witkin and P. Heckbert, Using particles to sample and control implicit surfaces, *Computer Graphics (SIGGRAPH '94)*, Orlando, 1994, pp. 269–278.
67. A. Witkin and M. Kass, Reaction–diffusion textures, *Computer Graphics (SIGGRAPH)* **25**(4), 299 (1991).
68. G. Yngve and G. Turk, Creating Smooth Implicit Surfaces from Polygonal Meshes, Technical Report GIT-GVU-99-42 (Graphics, Visualization, and Usability Center. Georgia Institute of Technology, 1999) (obtained from www.cc.gatech.edu/gvu/geometry/publications.html).
69. D. Zhang and M. Hebert, "Harmonic maps and their applications in surface matching," in *Proc. IEEE CVPR '99*, Los Alamitos, Colorado, (June 1999).
70. H. K. Zhao, T. Chan, B. Merriman, and S. Osher, A variational level set approach to multi phase motion, *J. Comput. Phys.* **127**, 179 (1996).
71. H. Zhao, S. Osher, B. Merriman, and M. Kang, Implicit, non-parametric shape reconstruction from unorganized points using a variational level set method, *Comput. Vision Image Understanding* **80**, 295 (2000).