SEGMENTATION ON SURFACES WITH THE CLOSEST POINT METHOD

Li (Luke) Tian[*] *Colin B. Macdonald*[†] *Steven J. Ruuth*[*]

Simon Fraser University
Department of Mathematics
Burnaby, Canada
lta12@sfu.ca

University of California, Los Angeles
Department of Mathematics
Los Angeles, USA
cbm@math.ucla.edu

Simon Fraser University
Department of Mathematics
Burnaby, Canada
sruuth@sfu.ca

ABSTRACT

We propose a method to detect objects and patterns in textures on general surfaces. Our approach applies the Chan–Vese variational model for active contours without edges to the problem of segmentation of scalar surface data. This leads to gradient descent equations which are level set equations on surfaces. These equations are evolved using the Closest Point Method, which is a recent technique for solving partial differential equations (PDEs) on surfaces. The final algorithm has a particularly simple form: it merely alternates a time step of the usual Chan–Vese model in a small 3D neighborhood of the surface with an interpolation step. We remark that the method can treat very general surfaces since it uses a closest point function to represent the underlying surface. Various experimental results are presented, including segmentation on smooth surfaces, non-smooth surfaces, open surfaces, and general triangulated surfaces.

***Index Terms*—** Image segmentation, Surfaces, Variational methods, Partial differential equations, Numerical analysis.

1. INTRODUCTION

Segmentation of objects and patterns in textures on *curved* surfaces (e.g., Figure 1) is a subject of recent and increasing interest. For example, Spira and Kimmel [1] segment out images painted on parametrically-defined manifolds using the geodesic active contour model for segmentation. Hara et al. [2] segment images on polar coordinate meshes using the Chan–Vese model. Applications of their methods arise in the analysis of earth data such as topography and remote sensing imagery. Bogdanova et al. [3] carry out segmentation via active contours for omnidirectional images defined on spherical, hyperbolical and parabolical shapes (such imagery arises using catadioptric cameras). Segmentation on surfaces via geometric active contours was also considered by Krueger et al. in [4]. Their work considers segmentation by both texture and surface geometry and leads to methods of interest in 3D human face feature segmentation.

A common theme to all these works is that they solve some PDE-based segmentation model on a surface of interest. The representation of the (static) surface is a central feature of each approach. For example, a parameterization might be used (e.g., [1, 2, 3]) but this has the deficiency of introducing distortions and singularities into the method through the parameterization. Indeed, designing a good parameterization can be a considerable challenge for many surfaces. Alternatively, a level set representation of the underlying surface may be considered (e.g., [4]). Algorithms based on level set representations have their own limitations [5]. Such methods solve a PDE in a 3D neighborhood of the surface and require the introduction of artificial boundary conditions at the boundary of the computational band. They also use non-standard PDEs in the embedding space (i.e., degenerate PDEs involving projection operators). Finally, level set representations do not give an obvious way to treat open surfaces, or any surface which lacks a clearly defined inside and outside.

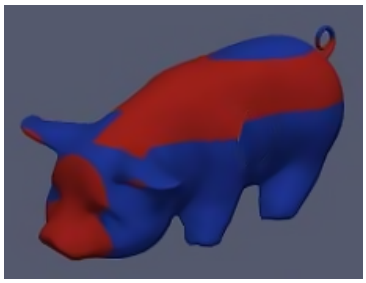

Fig. 1. The objective of this paper is to automatically segment texture on a surface into two regions: in this case, to separate the red patches from the rest of the pig.

For these reasons we are motivated to use the Closest Point Method [5, 6, 7] for solving the problem of segmentation on surfaces. The Closest Point Method uses a closest point representation of the surface, and therefore does not require the design of a parameterization nor does it require the surface to be closed or even have an orientation. The method uses entirely standard methods in a small 3D neighborhood of the surface in combination with an interpolation step. This is particularly attractive in practice since it enables us to use existing software for 3D region segmentation to segment out shapes on 2D surfaces.

All our segmentations are carried out using the Chan–Vese active contours without edges model [8], which we review in Section 2. Section 3 applies the Closest Point Method to the Chan–Vese model. This is followed by numerical experiments in Section 4.

2. THE CHAN–VESE ALGORITHM

We begin by outlining the derivation of the Chan–Vese model for segmentation on surfaces, The full details appear in [9] and follow closely that of the original Chan–Vese model [8].

Let $u_0 : S \to \mathbb{R}$ be our given scalar surface image. Segmentation will be carried out on the surface using a function $\phi : S \to \mathbb{R}$. Specifically, the zero contour of ϕ will divide the surface into two regions: $\{x \in S : \phi(x) > 0\}$ ("inside" the zero contour) and $\{x \in S : \phi(x) < 0\}$ ("outside" the zero contour). Segmentation

[*]The work of these authors was partially supported by a grant from NSERC Canada.

[†]The work of this author was partially supported by a grant from NSERC Canada and NSF grant number CCF-0321917.

will be based on minimization of the energy functional

$$\begin{aligned} F(u_1, u_2, \phi) = \mu \cdot (\text{length of } \phi\text{'s zero contour}) \\ + \lambda_1 \int_{\{x:\phi(x)>0\}} |u_0(x) - u_1|^2 dS \\ + \lambda_2 \int_{\{x:\phi(x)<0\}} |u_0(x) - u_2|^2 dS, \end{aligned}$$

where u_1 and u_2 are unknown scalars and $\mu \geq 0$, $\lambda_1, \lambda_2 > 0$ are fixed parameters. The energy can be written as an integral over the entire surface by introducing the Heaviside function $H(z)$ and the Dirac measure $\delta(z)$. Keeping ϕ fixed and minimizing the energy $F(u_1, u_2, \phi)$ with respect to the scalars u_1 and u_2 [8], gives

$$\begin{aligned} u_1(\phi) &= \frac{\int_S u_0(x) H(\phi(x)) dS}{\int_S H(\phi(x)) dS}, \\ u_2(\phi) &= \frac{\int_S u_0(x) \left(1 - H(\phi(x))\right) dS}{\int_S (1 - H(\phi(x))) dS}. \end{aligned} \tag{1}$$

That is, u_1 and u_2 are the average values of u_0 over the inside and outside respectively of the zero contour of ϕ.

Next, we introduce the regularized Heaviside function $H_\epsilon(z) = \frac{1}{2}\left(1 + \frac{2}{\pi} \arctan\left(\frac{z}{\epsilon}\right)\right)$ [8] and the associated Dirac measure $\delta_\epsilon(z)$. Let $F_\epsilon(u_1, u_2, \phi)$ denote the resulting regularized functional. Keeping u_1 and u_2 fixed, and minimizing F_ϵ with respect to ϕ gives the corresponding Euler–Lagrange equations. These will be solved by evolving the gradient descent equations

$$\begin{aligned} \frac{\partial \phi}{\partial t} = \delta_\epsilon(\phi) \Big[\mu \nabla_S \cdot \left(\frac{\nabla_S \phi}{|\nabla_S \phi|} \right) - \lambda_1 (u_0(x) - u_1)^2 \\ + \lambda_2 (u_0(x) - u_2)^2 \Big] \text{ in } (0, \infty) \times S, \end{aligned} \tag{2}$$

$$\phi(0, x) = \phi_0(x) \text{ in } S, \qquad \frac{\partial \phi}{\partial n} = 0 \text{ on } \partial S,$$

to steady state, where $t \geq 0$ is an artificial time. Finally, we remark that the regularized H_ϵ is also used in evaluating u_1 and u_2 in (1).

3. THE CLOSEST POINT METHOD FOR ACTIVE CONTOURS WITHOUT EDGES

We now give our algorithm for solving (2). A detailed description of our approach may be found in [9].

3.1. The embedding PDE

The surface PDE (2) is complicated to solve due to the surface derivative operators. The Closest Point Method is a new general technique for the numerical solution of PDEs and other processes on surfaces using standard Cartesian grid methods in $\mathbb{R}^3$. The method is based on a closest point representation of the surface. For a given surface S, the closest point function takes a point $x \in \mathbb{R}^3$ and returns a (possibly non-unique) point $cp(x) \in S$ which is closest in Euclidean distance to x. That is, $cp(x) = \text{argmin}_{q \in S} \|x - q\|_2$.

The main idea behind the Closest Point Method is that many differential operators on surfaces can be replaced by their corresponding Cartesian differential operators in a higher-dimensional space, provided the quantities are evaluated at the closest point $cp(x)$ [5]. For example, for points x on a smooth surface S, $\nabla_S u(x) = \nabla u(cp(x))$ because the function $u(cp(x))$ is constant in the normal direction and therefore only varies along the surface. Relevant to our segmentation problem is that a similar property holds for nonlinear diffusion terms [5]. Specifically, the in-surface curvature term appearing in (2) satisfies

$$\nabla_S \cdot \left(\frac{\nabla_S \phi}{|\nabla_S \phi|} \right) = \nabla \cdot \left(\frac{\nabla \phi(cp)}{|\nabla \phi(cp)|} \right),$$

for all points x on a smooth surface S. Embedding the problem in $\mathbb{R}^3$, and using the above as a replacement for the in-surface curvature term, we obtain the *embedding PDE*

$$\begin{aligned} \frac{\partial \phi}{\partial t} = \delta_\epsilon(\phi(cp)) \Big[\mu \nabla \cdot \left(\frac{\nabla \phi(cp)}{|\nabla \phi(cp)|} \right) - \lambda_1 (u_0(cp) - u_1)^2 \\ + \lambda_2 (u_0(cp) - u_2)^2 \Big] \text{ in } (0, \infty) \times \Omega. \end{aligned} \tag{3}$$

This is precisely the PDE for segmenting solid objects in 3D space, where function evaluations are carried out at the closest point, $cp(x)$, rather than at x itself. The functions u_1 and u_2 in (1) are also evaluated over the embedding space with all functions evaluated at $cp(x)$.

Note that the homogeneous Neumann boundary conditions appearing in the gradient descent equations (2) for an open surface need not be explicitly imposed with our approach: as is explained in [5], such boundary conditions arise naturally as part of the extension procedure. Furthermore, we do not need to introduce artificial boundary conditions at the edge of the computational domain: the closest point extension provides the necessary values for such points [5, 6].

3.2. Discretization

To discretize the PDE, we begin by selecting a computational domain Ω_c of discrete grid points: it is not necessary to solve on the whole domain, but just on a certain narrow band enveloping the surface [5, 6].

For each grid point in Ω_c, we need to construct a closest point function $cp(x)$ for the surface S. This will represent the surface. For simple surfaces such as a sphere, torus or cube, the closest point function has an easy analytical form. For triangulated surfaces, the closest point function can be computed efficiently by the algorithm outlined in [6].

To solve the embedding PDE (3) we alternate the following two steps:

1. Extend the solution off the surface to the computational domain using the closest point function. That is, replace ϕ by $\phi(cp)$ for each node on the computational domain. This *closest point extension* is an interpolation step, and is carried out using barycentric Lagrange interpolation [10].

2. Compute the solution to the embedding PDE (3) using standard finite differences on the Cartesian mesh for one time step. This step is just a step of the usual Chan–Vese algorithm in 3D for region segmentation.

The second step of the algorithm is discretized by a finite difference

scheme which is semi-implicit in time. Following [8], it is

$$\begin{aligned}\frac{\phi_{ijk}^{n+1}-\phi_{ijk}^{n}}{\Delta t} = \delta_h(\phi_{ijk}^n)\Bigg[& \\ & \frac{\mu}{h^2}\Delta_-^x \frac{\Delta_+^x \phi_{ijk}^{n+1}}{\sqrt{\frac{(\Delta_+^x\phi_{ijk}^n)^2}{h^2}+\frac{(\Delta_c^y\phi_{ijk}^n)^2}{4h^2}+\frac{(\Delta_c^z\phi_{ijk}^n)^2}{4h^2}}} \\ & +\frac{\mu}{h^2}\Delta_-^y \frac{\Delta_+^y \phi_{ijk}^{n+1}}{\sqrt{\frac{(\Delta_+^y\phi_{ijk}^n)^2}{h^2}+\frac{(\Delta_c^x\phi_{ijk}^n)^2}{4h^2}+\frac{(\Delta_c^z\phi_{ijk}^n)^2}{4h^2}}} \\ & +\frac{\mu}{h^2}\Delta_-^z \frac{\Delta_+^z \phi_{ijk}^{n+1}}{\sqrt{\frac{(\Delta_+^z\phi_{ijk}^n)^2}{h^2}+\frac{(\Delta_c^x\phi_{ijk}^n)^2}{4h^2}+\frac{(\Delta_c^y\phi_{ijk}^n)^2}{4h^2}}} \\ & -\lambda_1\left(u_{0,ijk}-u_1(\phi^n)\right)^2+\lambda_2\left(u_{0,ijk}-u_2(\phi^n)\right)^2\Bigg],\end{aligned} \tag{4}$$

where Δ_+^α, Δ_-^α, Δ_c^α indicate forward, backward and centered finite differences, in the direction indicated by the superscript α, on a grid spacing of h. The averages u_1 and u_2 are computed by simple quadrature over all grid cells in Ω_c.

The linear system (4) is solved by the iterative Gauss–Seidel method [11, 8]. Because we are interested in the steady state solution to (2), it is not necessary to iterate to convergence at each time step but merely to descend towards the minimum energy. In our numerical experiments, a fixed number of Gauss–Seidel iterations (15 iterations) is performed for each time step. See [9] for a detailed description of this procedure.

We remark that reinitialization is sometimes useful to prevent the level set function ϕ from becoming too flat or too steep [8]. Our numerical experiments did not make use of this option.

3.3. Efficiency improvement

Our ultimate goal is to find the steady state of the gradient descent equations (2). To achieve this objective, it is sometimes helpful to break the computation into two phases [9]. In the first phase, we are interested in a fast method which gives a qualitatively improved segmentation. In this phase, enhanced computational speed can be obtained by dropping the interpolation step in the algorithm. This gives a result which can be used as a starting condition for the second phase. The second phase needs to be consistent with the gradient descent equations (2) and therefore must include the interpolation step.

4. NUMERICAL EXPERIMENTS

In the numerical experiments that follow, we take $\lambda_1 = \lambda_2 = 1$ and $h = 1$. The range of the surface data u_0 is $[0, 255]$. In all instances, the initial contour is given by the intersection of the surface with that of a cylinder of radius 15 units. The closest point extension steps are performed using barycentric Lagrange interpolation [10] with cubic polynomials. Computations are performed in Matlab and visualizations are done using either Matlab or ParaView [12].

4.1. Segmentation on a hemisphere

The first experiment is to segment patterns on a hemisphere. The surface data consists of a greyscale bitmap image projected onto the hemisphere, and contains noise and artifacts.

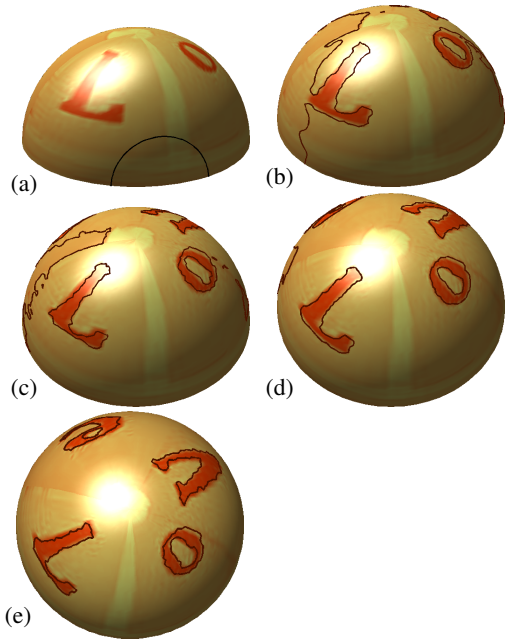

Fig. 2. Segmentation on the surface of a hemisphere, where (a)–(e) show the contour (dark curve) evolving in time to the steady state. The hemisphere is 72 units in diameter and the grid has $h = 1$. The calculation uses $\mu = 0.04 \cdot 255^2$.

Figure 2 shows the evolution of the contour ϕ as a dark curve on the hemisphere at various times. The contour converges to the desired segmentation of the letters in the image.

This example illustrates that the Closest Point Method naturally treats segmentation problems on open surfaces. We remark that no explicit imposition of boundary conditions is used at the boundary of the hemisphere; this corresponds to the desired homogeneous Neumann conditions [5].

4.2. Segmentation on a cube

The next experiment considers greyscale data on the surface of a cube. The top of the cube consists of a bitmap image of a galaxy and the other faces contain geometric shapes.

Figure 3 shows the evolution of the zero contour of ϕ at various times. With this choice of μ, the evolution segments out the core of the galaxy, as well as each of the shapes on the surface. It is noteworthy that the underlying surface is non-smooth. The analysis of the Closest Point Method on non-smooth surfaces is ongoing, but our results here suggest that the method is stable and achieves qualitatively correct results in the context of image segmentation.

4.3. Segmentation on a triangulated surface

The algorithm can also be applied to complicated triangulated shapes, such as Annie Hui's pig [13] (here smoothed via Loop's algorithm within ParaView [12]). The surface data u_0 consists of various shapes that were manually generated.

Figure 4 shows the evolution of the zero level set of ϕ (shown as a white contour) on the pig at various times. The segmentation automatically finds the boundary between the two colored regions. It is worth emphasizing that the code for segmenting the triangulated pig is identical to our other examples: segmentation on a pig,

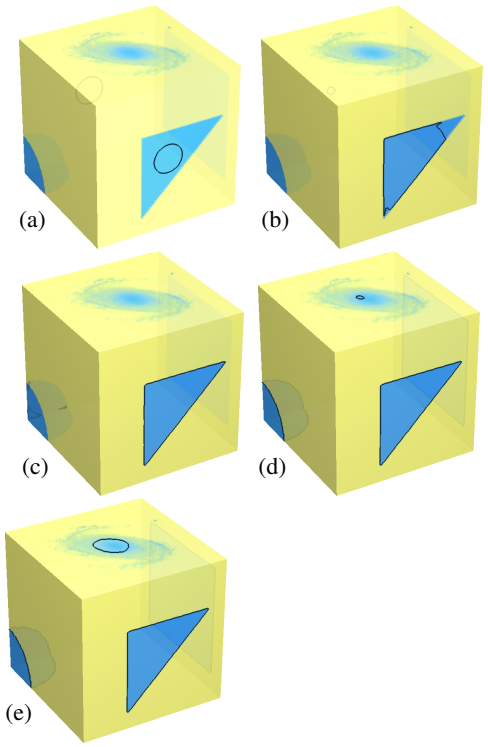

Fig. 3. Segmentation on the surface of a cube, where (a)–(e) show the contour evolving in time to the steady state. Here $\mu = 0.15 \cdot 255^2$. The cube is 80 units on each side.

cube, hemisphere or any other surface merely requires the input of the closest point function corresponding to the surface.

5. CONCLUSIONS AND SUMMARY

In this paper, the combination of the Chan–Vese segmentation algorithm and the Closest Point Method is proposed as a way to segment images defined on general surfaces, including open surfaces. The method is simple, and merely alternates a time step of the standard Chan–Vese algorithm in a small 3D neighborhood of the surface with an interpolation step.

While our experiments have only considered the Chan–Vese model, the Closest Point Method may be used to evolve other PDE-based segmentation models on surfaces as well. Another direction for future work is to design segmentation-on-surface algorithms using both surface texture and local surface geometry. See [4] for further discussion on this possibility.

We conclude by noting that our approach is not limited to segmentation models. De-noising, de-blurring, in-painting and other PDE-based image processing models may also be approximated using the Closest Point Method to give new methods for processing images defined on surfaces. We are investigating these applications as part of ongoing research.

6. REFERENCES

[1] A. Spira and R. Kimmel, "Geometric curve flows on parametric manifolds," *J. Comput. Phys.*, vol. 223, 2007.

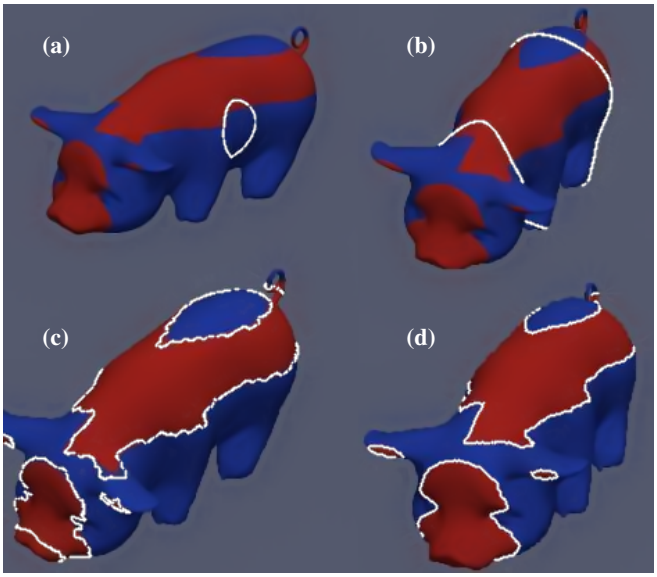

Fig. 4. Segmentation on the surface of a triangulated pig, where (a)–(d) show the contour evolving in time to the steady state. We take $\mu = 0.05 \cdot 255^2$ and the pig is roughly 75 units long.

[2] K. Hara, R. Kurazume, K. Inoue, and K. Urahama, "Segmentation of images on polar coordinate meshes," in *Proc. ICIP07, International Conference on Image Processing*, 2007.

[3] I. Bogdanova, X. Bresson, J.-P. Thiran, and P. Vandergheynst, "Scale-space analysis and active contours for omnidirectional images," *IEEE Trans. Image Process.*, vol. 16, no. 7, 2007.

[4] M. Krueger, P. Delmas, and G. Gimel'farb, "Active contour based segmentation of 3D surfaces," in *Proc. European Conference on Computer Vision (ECCV)*, 2008.

[5] S.J. Ruuth and B. Merriman, "A simple embedding method for solving partial differential equations on surfaces," *J. Comput. Phys.*, vol. 227, no. 3, 2008.

[6] C.B. Macdonald and S.J. Ruuth, "Level set equations on surfaces via the Closest Point Method," *J. Sci. Comput.*, vol. 35, no. 2–3, 2008.

[7] C.B. Macdonald and S.J. Ruuth, "The implicit Closest Point Method for the numerical solution of partial differential equations on surfaces," 2008, Submitted.

[8] T.F. Chan and L.A. Vese, "Active contours without edges," *IEEE Trans. Image Process.*, vol. 10, no. 2, 2001.

[9] L. Tian, "Segmentation on surfaces with the Closest Point Method," M.S. thesis, Department of Mathematics, Simon Fraser University, 2009.

[10] J.-P. Berrut and L.N. Trefethen, "Barycentric Lagrange interpolation," *SIAM Rev.*, vol. 46, no. 3, 2004.

[11] G. Aubert and L. Vese, "A variational method in image recovery," *SIAM J. Numer. Anal.*, vol. 34, 1997.

[12] A. Henderson, *ParaView Guide, A Parallel Visualization Application*, Kitware, Inc., third edition, 2008.

[13] A. Hui, "Annie Hui's pig in the AIM@SHAPE shape repository," `http://shapes.aimatshape.net`, 2008.