Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Computational Physics 204 (2005) 121–130
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

A topology-preserving level set method for shape optimization

Oleg Alexandrov *, Fadil Santosa

University of Minnesota School of Mathematics, Vincent Hall, 206 Church Str SE, Minneapolis MN 55455, USA

Received 25 May 2004; received in revised form 16 September 2004; accepted 2 October 2004

Abstract

The classical level set method, which represents the boundary of the unknown geometry as the zero-level set of a function, has been shown to be very effective in solving shape optimization problems. The present work addresses the issue of using a level set representation when there are simple geometrical and topological constraints. We propose a logarithmic barrier penalty which acts to enforce the constraints, leading to an approximate solution to shape design problems.

Keywords: Level set method; Optimization; Topology preservation; Steepest descent method

1. Introduction

The level set method [7,9,5], is a very powerful approach for problems involving geometry and geometric evolution. It has also been applied to solving shape optimization problems [1,10,6], and it is at this type of problems that this work is aimed. By a *shape* we mean a bounded region D in $\mathbb{R}^n$ with C^1 boundary. The level set method amounts to considering a function ϕ such that

$$D = \{x : \phi(x) > 0\}$$

(see Fig. 1) and manipulating D implicitly, through its *level set function* ϕ.

It is typical in shape optimization problems to start with an initial shape, which is then improved in an iterative process. Thus, one would start with a level set function $\phi(x)$ which is updated at each iteration.

The advantage of the level set method is that it is much easier to work with a globally defined function than to keep track of the boundary of a domain. The latter, which can be achieved by using marker points

* Corresponding author.
E-mail addresses: aoleg@math.umn.edu (O. Alexandrov), santosa@math.umn.edu (F. Santosa).

0021-9991/$ - see front matter
doi:10.1016/j.jcp.2004.10.005

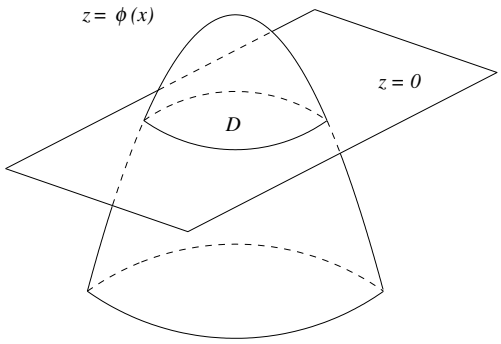

Fig. 1. The domain D and its level set function ϕ.

and spline interpolation, can become especially complicated if D has either several connected components, or is otherwise connected but has several holes. During the optimization process, the components or holes may merge or split, or even entirely disappear. The level set method, on the other hand, takes care of this kind of changes with great ease.

Given the shape D there exist of course many functions ϕ such that $D = \{x : \phi(x) > 0\}$. The most convenient ϕ to work with is the *signed distance* to ∂D, thus

$$\phi(x) = \begin{cases} \operatorname{dist}(x, \partial D), & x \in D, \\ -\operatorname{dist}(x, \partial D), & x \notin D. \end{cases} \tag{1}$$

Then ϕ will have the additional property

$$\nabla\phi(x) \cdot \nabla\phi(x) = 1 \tag{2}$$

for x in a neighborhood of ∂D. Any level set function ϕ can be reinitialized as the signed distance to the set $\{x : \phi(x) = 0\}$, so from here on we will assume that ϕ always satisfies (1), by reinitializing it if necessary.

It is very easy to describe deformations of D in terms of its level set function ϕ. For example, if $h : \mathbb{R}^n \to \mathbb{R}$ is a function with $\sup|h(x)|$ small enough, then the set $\{x : (\phi + h)(x) > 0\}$ is obtained from the set $D = \{x : \phi(x) > 0\}$ by shifting every point $x \in \partial D$ by approximately the amount $h(x)$ in the direction of the external normal to ∂D at x (which is $-\nabla\phi(x)$).

While the level set method has its strong points – one being that it gives a representation that is topology-independent – it is not obvious how to extend it to problems where there are constraints. Simple volume (area in 2-D) constraints are relatively easy to incorporate [6]. Other constraints, such as a bound on the size of a connected component of D, or the requirement that D has a fixed number of connected components, are not as easy to handle. It is towards this class of problems that this work is directed.

Our approach starts with the concept of subdomain neighborhood. The neighborhood of one subdomain will detect the nearness of other subdomains, and will thus allow us to take action to prevent geometrical or topological changes. This strategy can be formulated as a penalty functional, which we describe in Section 2. In Section 3, we deduce a descent direction which will enable us to find a minimizer for the obtained penalized optimization problem. We discuss numerical issues in implementing our method in Section 4. Lastly, in Section 5 we illustrate our method by several numerical examples.

We wish to mention the paper [3] which also suggests a way of adapting the level set method to preserve topology. The authors of this paper do it in the context of image segmentation. The key difference between our work and [3] is that their method is pixel-based. The algorithm in [3] is able to detect that a shape is about to change topology only when certain dimensions of the shape are of size comparable to the grid size.

In the context of image processing this makes a lot of sense, as then it is convenient to define a body to be connected as long as it is made up of one or more pieces joined together by at least one pixel.

We developed our topology preserving level set method having in view problems of shape design. In contrast to the pixel-based method mentioned above, we start with a continuous problem whereby we can specify certain conditions about how small, thin, or close certain features of the shape can get. The problem is then discretized using a grid as fine as needed to resolve the details of the optimal shape.

2. The penalty functional

A typical shape optimization problem is as follows. We are given a cost function F which depends on the geometry of the unknown shape. The problem is to find a shape such that the cost function is minimized (at least locally).

Let us represent the shape D as

$$D = \{x : \phi(x) > 0\}.$$

The optimization problem we wish to solve is

$$\min_{\phi} F(\phi),$$

subject to geometrical and topological constraints on D. The latter constraints are:

- *Shape topology.* The domain we design for must have, for example, a fixed number of connected components and holes.
- *Component size.* A lower bound on the size of each component and hole is prescribed.
- *Distance between components.* A lower bound on the distance between components (and holes) is prescribed. In the case of holes, we also prescribe a lower bound on the distance from each of the holes to the external boundary of the domain.

These constraints arise naturally in optimal design problems as we will illustrate in the numerical examples.

It turns out that all these constraints can be handled in a single penalty formulation. We will restrict our attention to 2-D problems, even though the same ideas will work in higher dimensions.

Assume for simplicity that D is a bounded and connected set in $\mathbb{R}^2$ with a set of holes inside of it, which are connected components of $\mathbb{R}^2 \setminus D$. If $d > 0$ and $l > 0$ are real numbers, denote

$$I_d = \{x + d\nabla\phi(x) : x \in \partial D\},$$

and

$$E_l = \{x - l\nabla\phi(x) : x \in \partial D\}$$

(see Fig. 2).

It follows from (2) that for d and l small enough, I_d and E_l are made up of points at distance d and l respectively from ∂D. (In fact, for $d = l$, the union of these two sets is exactly the set of *all* points at distance d from ∂D.) This implies that any two components of $\mathbb{R}^2 \setminus D$ (we consider the unbounded component too) are at distance more than d from each other if and only if I_d is entirely inside of D, that is, $\phi(x) > 0$ on I_d. Also, the gaps in D are neither "smaller" nor "thinner" than l if and only if E_l is a subset of $\mathbb{R}^2 \setminus D$, that is, $\phi(x) < 0$ on E_l. In view of the definitions of I_d and E_l, these translate into the conditions

$$\phi(x + d\nabla\phi(x)) > 0 \quad \text{and} \quad \phi(x - l\nabla\phi(x)) < 0 \quad \text{for } x \in \partial D.$$

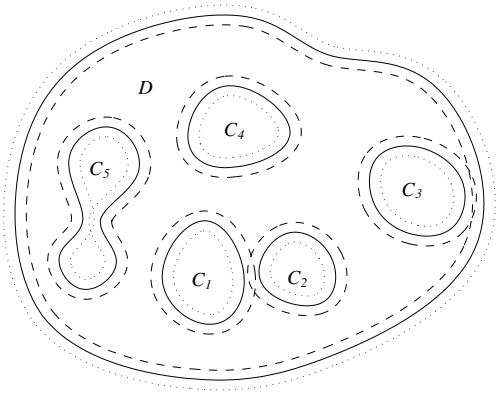

Fig. 2. The set I_d (dashed curves) and E_l (dotted curves).

To incorporate these conditions into the optimization problem we use the *logarithmic barrier method* [4]. Instead of trying to minimize $F(\phi)$, consider the problem of minimizing $F_\varepsilon(\phi) = F(\phi) + \varepsilon H(\phi)$ for $\varepsilon \ll 1$, where

$$H(\phi) = -\int_{\partial D} \log\left[\phi(x + d\nabla\phi(x))\right] \mathrm{d}s - \int_{\partial D} \log\left[-\phi(x - l\nabla\phi(x))\right] \mathrm{d}s.$$

Thus, by trying to achieve a minimal value for $F_\varepsilon(\phi)$, we keep the value of $H(\phi)$ not too large, and in particular, this functional is defined, which implies that the topological constraints are preserved.

3. The descent direction

To obtain ϕ minimizing $F_\varepsilon(\phi)$ we will use the *steepest descent method*. It amounts to calculating the Fréchet derivative of $F_\varepsilon(\phi)$, and at each iteration taking a step in the direction in which $F_\varepsilon(\phi)$ decreases fastest.

In order to calculate the derivative of $F_\varepsilon(\phi)$ we need the derivatives of $F(\phi)$ and $H(\phi)$. Let $h : \mathbb{R}^2 \to \mathbb{R}$ be a test function. Denote

$$D_\phi F(\phi) \cdot h = \left.\frac{\mathrm{d}F(\phi + th)}{\mathrm{d}t}\right|_{t=0}.$$

Let us note that, as $t \to 0$, $F(\phi + t\,h)$ will depend on the values of h only on an ever shrinking neighborhood of ∂D, as F is a function only of the set $\{x{:}(\phi + t\,h)(x) > 0\}$, and the way this set depends on h was discussed in the introduction. Then, intuitively we would expect that $D_\phi F(\phi) \cdot h$ will be a function only of $h|_{\partial D}$ and ϕ. This is indeed true. According to a result called the "Hadamard-Zolésio structure theorem" [2], if F, D, and h are sufficiently regular, then

$$D_\phi F(\phi) \cdot h = \int_{\partial D} U(x)h(x)\,\mathrm{d}s, \tag{3}$$

for some function U which depends on ϕ.

The derivative of $H(\phi)$ can be calculated explicitly. Consider a parameterization $x(s)$ of ∂D, with $x'(s)$ having unit length for all s. $H(\phi + th)$ will be a sum of two integrals over the set $\{x{:}(\phi + t\,h)(x) = 0\}$, which, if (2) holds, is approximately parameterized by $x - t\,h(x)\nabla\phi(x)$, with $x = x(s)$. One can then find that the derivative of the first integral in $H(\phi + th)$ at $t = 0$ is

$$\int_{\partial D}\left\{\frac{\left[\nabla\phi(x)h(x)+dh(x)\nabla^2\phi(x)\nabla\phi(x)-d\nabla h(x)\right]\cdot\nabla\phi(x+d\nabla\phi(x))-h(x+d\nabla\phi(x))}{\phi(x+d\nabla\phi(x))}\right.$$
$$\left.+\log\left[\phi(x+d\nabla\phi(x))\right]x'\cdot\left[(\nabla h(x)\cdot x')\nabla\phi(x)+\left(\nabla^2\phi(x)x'\right)h(x)\right]\right\}\mathrm{d}s. \tag{4}$$

A similar equality holds for the second term in $H(\phi)$.

Beside the obvious complexity of this expression, note that unlike the case of $F(\phi)$, this derivative will no longer depend on the values of the test function h only on ∂D. We will make several approximations. Recall that the purpose of $H(\phi)$ is to make sure at every step in the optimization process the domain D has the topology preserved. $H(\phi)$ will grow large only when D is close to violating the restrictions imposed on it. As far as the first integral in $H(\phi)$ is concerned, this happens when $\phi(x + d\nabla\phi(x))$ becomes close to zero. Then, the term on the first line of (4) is much larger than the second. For this reason, we will ignore the term on the second line. Also, on the first line, we have $\nabla^2\phi(x)\nabla\phi(x) = 0$, which follows from (2). Since d is supposed to be a small number, we will replace $h(x + d\nabla\phi(x))$ with its first order Taylor expansion $h(x) + d\nabla h(x)\cdot\nabla\phi(x)$. Then, the numerator of the expression on the first line of (4) becomes

$$h(x)\{\nabla\phi(x)\cdot\nabla\phi(x+d\nabla\phi(x))-1\}-d\nabla h(x)\cdot[\nabla\phi(x+d\nabla\phi(x))+\nabla\phi(x)]. \tag{5}$$

To further simplify this expression, we will need a lemma.

Lemma 3.1. *Assume that two connected components C_1 and C_2 of $\mathbb{R}^2 \setminus D$ are at distance slightly larger than d. Then, for points $x \in \partial C_1$ closest to ∂C_2 one has*

$$\nabla\phi(x) \approx -\nabla\phi(x + d\nabla\phi(x)).$$

Proof. Let us see what happens when ∂C_1 and ∂C_2 are at distance *exactly* d from one another, and $x \in \partial C_1$, $x' \in \partial C_2$ are such that $\mathrm{dist}(x, x') = \mathrm{dist}(\partial C_1, \partial C_2)$. Consider this situation in Fig. 3. Then, the segment going from x to x' will be perpendicular to the curves ∂C_1 and ∂C_2 at these points. Since $\nabla\phi(x)$ is also perpendicular to ∂C_1 and points outward C_1, and since $|\nabla\phi(x)| = 1$, it follows that $x' - x = d\nabla\phi(x)$. In the same manner one obtains $x - x' = d\nabla\phi(x')$. Thus we have $x' = x + d\nabla\phi(x)$ and

$$\nabla\phi(x) = \frac{x'-x}{d} = -\frac{x-x'}{d} = -\nabla\phi(x') = -\nabla\phi(x+d\nabla\phi(x)).$$

Clearly, if the distance between ∂C_1 and ∂C_2 is slightly larger than d, this equality will hold only approximately. □

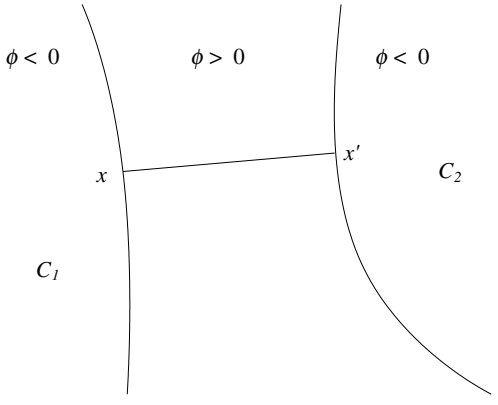

Fig. 3. The case when $\mathrm{dist}(x_1,x_2) = \mathrm{dist}(\partial C_1, \partial C_2) = d$.

With the help of this lemma, and if we recall that we care about the expression (4) only when two components of $\mathbb{R}^2 \setminus D$ are getting at distance slightly larger than d from one another, we can then drop the second term in (5), and simplify (4) to

$$\int_{\partial D} U_1(x)h(x)\,\mathrm{d}s,$$

with

$$U_1(x) = \frac{\nabla\phi(x)\cdot\nabla\phi(x+d\nabla\phi(x))-1}{\phi(x+d\nabla\phi(x))}, \quad x\in\partial D. \tag{6}$$

The derivative of the second integral in H can be calculated, and then approximated, in the same way. Make the notation

$$U_2(x) = \frac{\nabla\phi(x)\cdot\nabla\phi(x-l\nabla\phi(x))-1}{\phi(x-l\nabla\phi(x))}, \quad x\in\partial D. \tag{7}$$

We obtain

$$D_\phi H(\phi)\cdot h = \int_{\partial D}[U_1(x)+U_2(x)]h(x)\,\mathrm{d}s,$$

which gives us the following approximate equality

$$D_\phi F_\varepsilon(\phi)\cdot h = \int_{\partial D}[U(x)+\varepsilon U_1(x)+\varepsilon U_2(x)]h(x)\,\mathrm{d}s.$$

At each step in the optimization process, we will take a step in the direction

$$u(x) = -[U(x)+\varepsilon U_1(x)+\varepsilon U_2(x)], \tag{8}$$

where $x\in\partial D$. This quantity can be extended continuously to a neighborhood of ∂D in the following manner: for $x\in\mathbb{R}^2$ close to ∂D let $\widetilde{x}\in\partial D$ be the unique point such that $\mathrm{dist}(x,\partial D)=\mathrm{dist}(x,\widetilde{x})$, and set

$$u(x) = \phi(x)+u(\widetilde{x}). \tag{9}$$

Thus, the next iteration for ϕ would be $\phi+\alpha u$, where $\alpha>0$ is the length of the step to be taken in the direction u.

Since the formula for $D_\phi F_\varepsilon(\phi)\cdot h$ we found is not exact, u will not be the steepest descent direction, actually it might not be a descent direction at all. Nevertheless, we will argue below that this iterative process does its job at maintaining the topology constraints. And as far as the problem of minimizing $F(\phi)$ is concerned, it is clear that the iterative process we suggest will give us a sufficiently good approximation to the point of minimization ϕ, provided that ε is small enough.

We will show that, if the level set function ϕ is such that two components of $\{x:\phi(x)<0\}$ are at distance slightly more than d from one another, then u will act as a repelling force, and in consequence, the components of $\{x:(\phi+\alpha u)(x)<0\}$ will be further apart.

Indeed, let C_1 and C_2 be two such components, and let $x\in\partial C_1$ be a point at distance slightly larger than d from C_2. Then, from Lemma 3.1 and Eqs. (2) and (6), one obtains

$$U_1(x) = \frac{\nabla\phi(x)\cdot\nabla\phi(x+d\nabla\phi(x))-1}{\phi(x+d\nabla\phi(x))} \approx -\frac{2}{\phi(x+d\nabla\phi(x))},$$

which is large in magnitude and negative. Moreover, when the distance between x and C_2 gets quite close to d, $\varepsilon\ U_1(x)$ will be larger in magnitude than $U(x)+\varepsilon U_2(x)$. In consequence, $u(x)$ defined by (8) will be positive. Therefore, we have $\phi(x)=0$, but $(\phi+\alpha u)(x)>0$. The same reasoning applies for points $x\in\partial C_2$ close to $\partial\ C_1$. This shows that the connected components of $(\phi+\alpha\ u)(x)<0$ will be further apart.

It can be argued in the same manner that should a component of $\{x : \phi(x) < 0\}$ get too "thin" or too "small", then $U_2(x)$ will serve as a counterweight, forcing it to get "fatter".

4. Numerical implementation

The idea of the algorithm is then to perform an iterative process, at each step replacing ϕ by $\phi + \alpha u$, with $\alpha > 0$ being the step size. Let us note that in order for this to work, each step size should not be too big. Indeed, αu determines by how much the boundary of D gets shifted at a given step, and if the boundary of D moves by more than $d/2$ at a time, then two components of $\{x : \phi(x) < 0\}$ which were at distance slightly more than d can end up merging without the penalty functional noticing that. Or, if the boundary moves by more than l/2, a connected component slightly thinner or larger than l might end up splitting or disappearing. Therefore, at each step one needs to make sure that the step size α is such that

$$\alpha \max_{x \in \partial D} |u(x)| < K \min(d, l) \tag{10}$$

with $K > 0$. Theoretically K can be allowed to be as large as 1/2, but since we use a finite grid size we have to be more conservative. A value of $K = 1/4$ works in practice.

But even enforcing (10) is not enough to guarantee our geometrical and topological constraints. The penalty functional $H(\phi)$ is supposed to take care of this, but it is clear that the smaller ε is, the weaker the influence of $H(\phi)$ in $F_\varepsilon(\phi)$ will be, and the closer to violating the constraints ϕ will get, before this penalty functional kicks in. Thus, at each iteration one needs to first take a step size α satisfying (10), and still check after updating ϕ to $\phi + \alpha u$ whether $H(\phi)$ is defined. If not, one needs to decrease the step size α, for example by halving it, until $H(\phi)$ is defined. If no amount of decreasing α helps, one needs to either increase ε or decrease the grid size, and restart the algorithm.

A pseudo-code for the algorithm is shown in Fig. 4.

We note that if at some point the contour $\{x : \phi(x) = 0\}$ develops sharp angles, then the functional $H(\phi)$ might not be defined (this can be seen from Fig. 2). To prevent this from happening, one can smooth ϕ a bit at each iteration. For ϕ discretized on a square grid we used the procedure

$$\phi_{i,j} \to \frac{\phi_{i,j} + \phi_{i-1,j} + \phi_{i+1,j} + \phi_{i,j-1} + \phi_{i,j+1}}{5}.$$

Also, for fine grids it becomes expensive to reinitialize ϕ according to (1). To make this computation faster we reinitialized ϕ only in a neighborhood of the set $\{x : \phi(x) = 0\}$. For more performance one could use the fast re-distancing algorithms suggested in [8,11,12].

Lastly, sometimes one might wish to introduce additional constraints of the form $G(\phi) = \text{const.}$ in the optimization problem. An example of such a constraint is the requirement that the area of the set $\{x : \phi(x) > 0\}$ be kept fixed, which we will use in the two numerical examples below. Then one needs to modify the descent direction u as described in [6].

```
initial guess for φ
do while not optimal
 • compute the descent direction u (use (6), (7), (8), and (9))
 • choose a step size α satisfying (10) for which H(φ + αu) is
   defined
 • update φ to φ + αu
 • reinitialize φ to satisfy (1)
```

Fig. 4. The pseudo-code for the algorithm.

5. Numerical examples

In the first example, we consider the problem of finding a domain that has the smallest perimeter, subject to the constraint that the area of the domain be fixed. Thus, the functional to minimize is

$$F(\phi) = \int_{\{\phi=0\}} 1 \, \mathrm{d}s,$$

with the constraint

$$G(\phi) = \int_{\{\phi>0\}} 1 \, \mathrm{d}x = \text{const.}$$

The starting shape is a region with seven subdomains, each one an ellipse with aspect ratio 1.3, as shown in Fig. 5 on the left. The center ellipse has a slightly bigger (20%) size than the rest. The distance between the centers of the ellipses is 4, and the smallest semi-axis of the surrounding ellipses is 1.

If we do not constrain the topology or geometry, the optimal solution would be a disk whose area is equal to the area of the original seven subdomains. If we do enforce these constraints, minimizing instead the functional

$$F(\phi) + \varepsilon H(\phi),$$

we obtain the picture in Fig. 5 on the right.

For this calculation we set $d = l = 0.8$, $\varepsilon = 0.25$ and considered a square grid of size $h = 0.05$ (each square is further split into two triangles, to make it easier to keep track of the set $\{x : \phi(x) = 0\}$).

We find that the "satellite" components of the central domain do not disappear, but became of size slightly larger than l.

We note that that the resulting large domain in the center is not perfectly circular. This because the steepest descent direction for $F(\phi)$ will be $\Delta\phi = \partial^2\phi/\partial x_1^2 + \partial^2\phi/\partial x_2^2$. We need to calculate this quantity numerically, and after reinitializing ϕ according to (1) it is not smooth enough for $\Delta\phi$ to be calculated accurately. Smoothing ϕ as noted in the previous section helped a bit, this is how this picture was obtained. We found that if we perform additional smoothing then the result in Fig. 3 will look more circular. This artifact does not show up in the next example, as then one does not need to calculate second-order derivatives of ϕ.

In the second example we examine the problem of minimizing the functional

$$F(\phi) = \int_{\{\phi>0\}} (x_1^2 + x_2^2) \, \mathrm{d}x_1 \, \mathrm{d}x_2.$$

This functional, which is the second moment of area, measures how concentrated around the origin the set $\{x : \phi(x) > 0\}$ is. We again enforce the area constraint $G(\phi) = \text{const.}$, and we use the same initial shape and

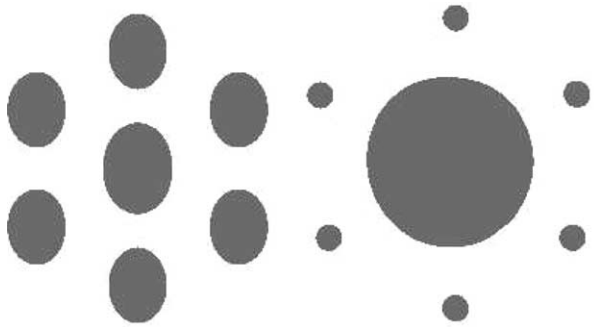

Fig. 5. The initial and optimized shape for numerical example 1.

the same values for d, l and h. We set $\varepsilon = 0.5$. In absence of topological constraints, these seven ellipses would merge to form a large circle. The topological constraints prevent them from doing so, as we see from Fig. 6.

For Example 2, we performed several experiments varying ε. For $\varepsilon = 1$, we found the optimal shape pictured in Fig. 7. The shape obtained for $\varepsilon = 0.5$ (pictured in Fig. 6) was essentially the same as the one found for $\varepsilon = 0.25$, except that in the latter case the "satellite" components were slightly smaller, which is to be expected, as for smaller ε the influence of the penalty functional is weaker. The algorithm failed to converge for $\varepsilon = 0.125$. We also found that, as expected, the smaller ε is, the smaller the value of the functional F is for the resulting shape.

We also ran several experiments varying d and l, with $\varepsilon = 0.5$ fixed. For $d = l = 0.6$ we found essentially the same optimal solution as for $d = l = 0.8$ (pictured in Fig. 6) except that the "satellite" components were slightly smaller. The algorithm failed to converge for $d = l = 1$. We then halved the grid size, to $h = 0.025$. We obtained the result shown in Fig. 8. We see that the "satellite" components are quite a bit larger than in Fig. 6.

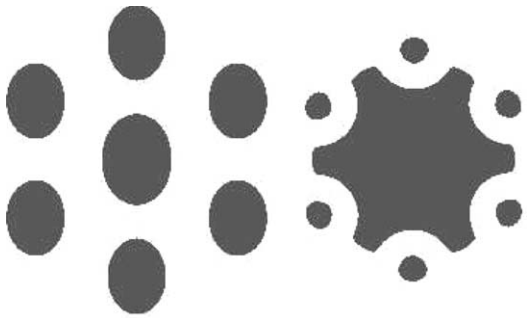

Fig. 6. The initial and optimized shape for numerical example 2.

Fig. 7. Running Example 2 for $\varepsilon = 1$.

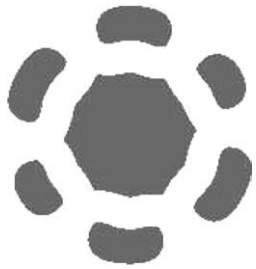

Fig. 8. Running Example 2 for $d = l = 1$ and $h = 0.025$.

6. Discussion

In this paper we introduced a penalty functional which makes it possible to use the level set method in problems with topology and geometry constraints. Our method allows for topological constraints independent of the grid size (that is, for given d and l, the grid size h can be chosen as small as desired), which is a key difference with the method suggested in [3].

Acknowledgments

We thank Grant Erdmann, whose suggestion that a logarithmic barrier method could be used to preserve constraints lead to the penalty functional we employ in this paper. We are grateful to the anonymous referees who made very helpful suggestions to the original manuscript, many of which we have incorporated in the present version. This work is supported in part by the National Science Foundation.

References

[1] Grégoire Allaire, François Jouve, Anca-Maria Toader, A level-set method for shape optimization, C. R. Math. Acad. Sci. Paris 334 (12) (2002) 1125–1130.
[2] M.C. Delfour, J.-P. Zolésio, Shapes and Geometries. Advances in Design and Control, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2001.
[3] Xiao Han, Chenyang Xu, Jerry L. Prince, A topology preserving level set method for geometric deformable models, IEEE Trans. PAMI 25 (6) (2003) 755–768.
[4] Jorge Nocedal, Stephen J. Wright, Numerical optimization Springer Series in Operations Research, Springer, New York, 1999.
[5] Stanley J. Osher, Ronald Fedkiw, Level Set Methods and Dynamic Implicit Surfaces, Applied Mathematical Sciences, vol. 153, Springer, New York, 2003.
[6] Stanley J. Osher, Fadil Santosa, Level set methods for optimization problems involving geometry and constraints. I. Frequencies of a two-density inhomogeneous drum, J. Comput. Phys. 171 (1) (2001) 272–288.
[7] Stanley J. Osher, James A. Sethian, Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations, J. Comput. Phys. 79 (1) (1988) 12–49.
[8] Giovanni Russo, Peter Smereka, A remark on computing distance functions, J. Comput. Phys. 163 (1) (2000) 51–67.
[9] James A. Sethian, Level Set Methods and Fast Marching Methods, Cambridge Monographs on Applied and Computational Mathematics, second ed., vol. 3, Cambridge University Press, Cambridge, 1999.
[10] James A. Sethian, Andreas Wiegmann, Structural boundary design via level set and immersed interface methods, J. Comput. Phys. 163 (2) (2000) 489–528.
[11] John Strain, Fast tree-based redistancing for level set computations, J. Comput. Phys. 152 (2) (1999) 664–686.
[12] Mark Sussman, Emad Fatemi, An efficient, interface-preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow, SIAM J. Sci. Comput. 20 (4) (1999) 1165–1191 (electronic).