HAL
open science

Flip procedure in geometric approximation of multiple-component shapes – Application to multiple-inclusion detection

Pierre Bonnelie, Loïc Bourdin, Fabien Caubet, Olivier Ruatta

To cite this version:

Pierre Bonnelie, Loïc Bourdin, Fabien Caubet, Olivier Ruatta. Flip procedure in geometric approximation of multiple-component shapes – Application to multiple-inclusion detection. SMAI Journal of Computational Mathematics, 2016, 2, pp.255-276. hal-01710845

HAL Id: hal-01710845

https://hal.science/hal-01710845v1

Submitted on 25 Feb 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

SMAI-JCM
SMAI Journal of Computational Mathematics

Flip procedure in geometric approximation of multiple-component shapes – Application to multiple-inclusion detection

Pierre Bonnelie, Loïc Bourdin, Fabien Caubet & Olivier Ruatta

Volume 2 (2016), p. 255-276.

<http://smai-jcm.cedram.org/item?id=SMAI-JCM_2016__2__255_0>

© Société de Mathématiques Appliquées et Industrielles, 2016
Certains droits réservés.

cedram
Article mis en ligne dans le cadre du
Centre de diffusion des revues académiques de mathématiques
http://www.cedram.org/

SMAI Journal of Computational Mathematics
Vol. 2, 255-276 (2016)

Flip procedure in geometric approximation of multiple-component shapes – Application to multiple-inclusion detection

Pierre Bonnelie [1]
Loïc Bourdin [2]
Fabien Caubet [3]
Olivier Ruatta [4]

[1] Institut de recherche XLIM. Pôle Mathématiques-Informatique-Image. UMR CNRS 7252. Université de Limoges, France.
E-mail address: pierre.bonnelie@etu.unilim.fr
[2] Institut de recherche XLIM. Pôle Mathématiques-Informatique-Image. UMR CNRS 7252. Université de Limoges, France.
E-mail address: loic.bourdin@unilim.fr
[3] Institut de Mathématiques de Toulouse. UMR CNRS 5219. Université de Toulouse, France.
E-mail address: fabien.caubet@math.univ-toulouse.fr
[4] Institut de recherche XLIM. Pôle Mathématiques-Informatique-Image. UMR CNRS 7252. Université de Limoges, France.
E-mail address: olivier.ruatta@unilim.fr.

Abstract. We are interested in geometric approximation by parameterization of two-dimensional multiple-component shapes, in particular when the number of components is *a priori* unknown. Starting a standard method based on successive shape deformations with a one-component initial shape in order to approximate a multiple-component target shape usually leads the deformation flow to make the boundary evolve until it surrounds all the components of the target shape. This classical phenomenon tends to create double points on the boundary of the approximated shape.

In order to improve the approximation of multiple-component shapes (without any knowledge on the number of components in advance), we use in this paper a piecewise Bézier parameterization and we consider two procedures called *intersecting control polygons detection* and *flip procedure*. The first one allows to prevent potential collisions between two parts of the boundary of the approximated shape, and the second one permits to change its topology by dividing a one-component shape into a two-component shape.

For an experimental purpose, we include these two processes in a basic geometrical shape optimization algorithm and test it on the classical inverse obstacle problem. This new approach allows to obtain a numerical approximation of the unknown inclusion, detecting both the topology (*i.e.* the number of connected components) and the shape of the obstacle. Several numerical simulations are performed.

Math. classification. 68U05, 68W25, 49Q10, 65N21.

Keywords. Shape approximation; free-form shapes; multiple-component shapes; Bézier curves; intersecting control polygons detection; flip procedure; inverse obstacle problem; shape optimization.

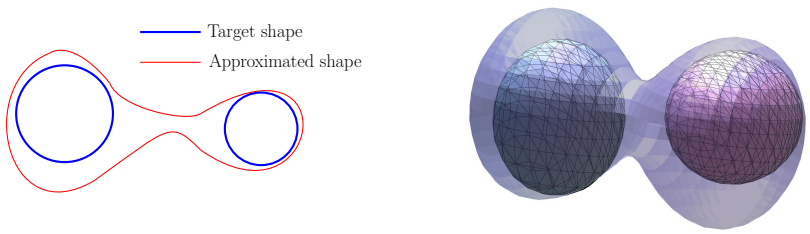
1. Introduction

Geometric shape approximation methods are commonly based on successive shape deformations, where the boundary of the approximated shape is parameterized and evolves at each step in a direction given by the deformation flow. This technique is widely used for example in shape optimization problems where the flow is given by the so-called *shape gradient* (see, e.g., Chapter 5 of the book [23] of Henrot *et al.*), or in image segmentation (see, e.g., [25]). Numerous parameterizations of the boundary have

The first author is granted by Labex SIGMA-LIM.

been considered in the literature, such as polygons, Fourier series, etc. Each of these parameterizations has its own advantages and drawbacks, that depend on the nature of the problem studied.

In this paper we are especially interested in the geometric approximation of multiple-component shapes, in particular when the number of components is *a priori* unknown. Starting a parameterization method with a one-component initial shape in order to approximate a multiple-component target shape usually leads the deformation flow to make the boundary evolve until it surrounds all the components of the target shape (see Figure 1.1 for illustrations). This classical phenomenon tends to create double points on the boundary of the approximated shape.



(a) Two-dimensional case (b) Three-dimensional case

Figure 1.1. Geometric shape approximation of a two-component target shape starting from a one-component initial shape.

In order to improve the approximation of multiple-component shapes, our idea is to look for an appropriate parameterization that allows to achieve two numerical tasks. Firstly the parameterization has to be well-suited in order to prevent the potential formation of double points, *i.e.* to locate the parts of the boundary that are close to each other. Secondly it has to be adapted in order to easily change the topology of the approximated shape, precisely in order to divide a one-component shape into a two-component shape. Moreover, for practical uses, we look for a complete method that is easily implementable with a relatively low numerical cost.

We present in this paper a method based on a Bézier parameterization. The main idea is that this polynomial parameterization can be approximated by its control polygon. In particular one can easily prevent the potential formation of double points by looking for intersecting control polygons. We refer to Section 3.2 for details on the so-called *intersecting control polygons detection*. Once this first step is achieved, one can easily reorganize the control points of the Bézier parameterization in order to modify the topology of the shape, precisely in order to divide one component into two. We refer to Section 3.3 for details on the so-called *flip procedure*.[1] In this work we detail the above method in the two-dimensional case, using piecewise Bézier curves.[2]

In order to test the two procedures introduced in this paper, we perform numerical simulations on the classical inverse obstacle problem. Precisely we consider the inverse problem of detecting some unknown inclusions ω_{ex} in a larger bounded domain Ω from boundary measurements made on $\partial\Omega$. The aim is to reconstruct numerically an approximation of the target shape ω_{ex} using shape optimization tools (see Figure 1.2 for an illustration). In this paper we will study this inverse problem by minimizing a shape least-square functional.

[1]Actually a similar procedure can also be considered in order to merge two components into one (see Appendix B for some details).

[2]Let us give a brief discussion on the three-dimensional case. We refer for instance to [31] where deformation of piecewize Bézier surfaces is presented with an implementation. Note that the adaptation of the complete algorithmic setting of the flip procedure to the three-dimensional case would be nontrivial since it would increase the algorithmic and combinatoric complexities. Numerous considerations about this generalization could be addressed, however we postpone this interesting issue to a future work.

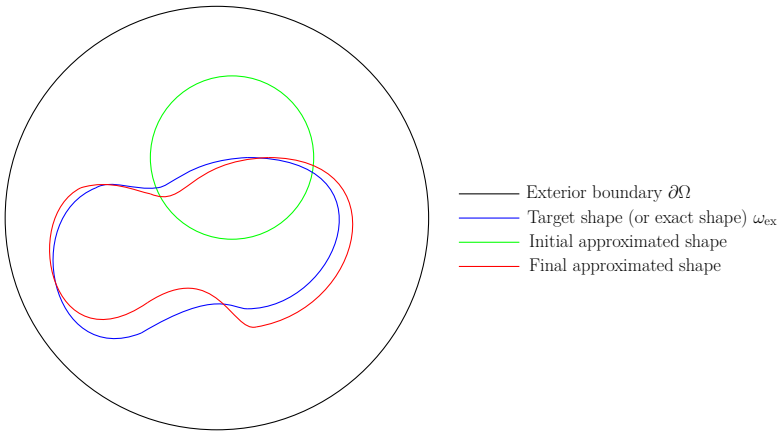


Figure 1.2. Illustration of reconstruction for the inverse obstacle problem.

We briefly recall now the major shape optimization techniques used in order to study this problem in the literature. Two main categories are *topological* and *geometric* shape optimization methods. The *topological* gradient approach was introduced by Schumacher in [33] and Sokolowski *et al.* in [36]. This method is based on asymptotic expansions and consequently is essentially adapted for relatively small inclusions. Moreover, even if the topological optimization is useful in order to find the number of inclusions, it may be not well-suited in order to find a satisfactory approximation of the shape of the inclusions (see, e.g., [12] and references therein). We refer to [4] and references therein for a comprehensive mathematical treatment with theoretical and numerical results about reconstruction of small inclusions from boundary measurements. We also refer to the work [21] for generalities on topological asymptotic expansion in the elastic context. In the *geometric* shape optimization category, two main techniques are addressed in the literature. They are both based on the computation of a shape gradient used as a flow making the shape evolve. These two methods use different representations of the shape and different techniques to deform it. The first approach is the so-called level set approach (see, e.g., the survey [9] of Burger *et al.* and references therein or [35]). It is originally based on an implicit representation of the approximated shape on a fixed mesh and, in the case of inverse problems, some regularization methods are usually needed (as curve shortening in, e.g., [32, Section 8]). In order to detect several inclusions, this method does not need any *a priori* knowledge on the number of inclusions. The second approach is based on boundary variations *via* mesh variations and, in the case of inverse problems, on an explicit representation of the approximated shape. This method is used e.g. in the work [1] of Afraites *et al.* (where a regularization by parameterization is used). Note that the standard algorithm based on shape derivatives moving the mesh does not provide the opportunity to change the topology of the shape and consequently the number of inclusions has to be known in advance. Recent works propose to mix several of the above different approaches. For instance we refer to the works of Allaire *et al.* in [2, 3] and Burger *et al.* in [8] (see also the thesis [30, Section 5]) that combine the classical geometric shape optimization through the level set method and the topological gradient, to the work of Pantz *et al.* in [29] which develops an algorithm using boundary variations, topological derivatives and homogenization methods and to the works of Caubet *et al.* in [11] and Christiansen *et al.* in [15] which couple topological and boundary variations approaches.

The method presented in this paper is based only on mesh variation techniques. The parameterization by piecewise Bézier curves and the flip procedure permit to dynamically change the topology of the shape in order to find the number of inclusions, and the shape derivatives approach allows to approximate the shape of the inclusions with an explicit representation. This new method seems to be well-suited in order to study the above inverse obstacle problem, in particular in the case where

the number of inclusions is *a priori* unknown, and can be seen as an alternative to the above mixed methods combining the level set approach and the topological gradient.

Organization of the paper. The paper is organized as follows. Section 2 recalls some basics and notations about piecewise Bézier curves. Section 3 is concerned with the two main features of this paper, that is, the intersecting control polygons detection and the flip procedure. Section 4 is dedicated to several numerical simulations in the context of the inverse obstacle problem.

2. Notations and basics on piecewise Bézier curves

In this section we fix our notations and recall some basics about Bézier curves (see, e.g., [19, 34] or [20, from p. 409] for more details). Let $d \in \mathbb{N}^*$ and a set of $d+1$ points $P_0, \dots, P_d$ of $\mathbb{R}^2$. The associated Bézier curve, denoted by $B([P_0, \dots, P_d])$, is defined by

$$\forall t \in [0,1], \quad B([P_0, \dots, P_d], t) := \sum_{j=0}^{d} P_j b_{j,d}(t),$$

where $b_{j,d}$ are the classical Bernstein polynomials given by

$$b_{j,d}(t) := \binom{d}{j} t^j (1-t)^{d-j}.$$

The integer d is the *degree* of the curve and the points $P_0, \dots, P_d$ are its *control points* (or its *control polygon*). Note that a Bézier curve does not go through its control points in general. However it starts at P_0 and finishes at P_d. If $P_0 = P_d$, the Bézier curve is said to be *closed*. Each point of a Bézier curve is a convex combination of its control points. As a consequence, a Bézier curve lies in the convex hull of its control polygon (see Figure 2.1).

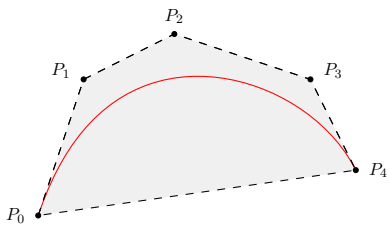


Figure 2.1. A non-closed Bézier curve of degree 4 lying in the convex hull of its control polygon.

Remark 2.1. As Bézier curves are widely used in Computer Aided Geometric Design (see [19, 34]), they are commonly defined as parametric curves lying in the euclidean space $\mathbb{R}^2$ (or $\mathbb{R}^3$). However this definition can be extended to $\mathbb{R}^n$ for any $n \in \mathbb{N}^*$. In this paper, we are only interested in the two-dimensional case $n = 2$.

In this paper we focus on the geometric approximation of boundaries of two-dimensional bounded shapes with the help of Bézier curves. In the sequel no distinction will be done between a two-dimensional bounded shape and its boundary.

Using a single closed Bézier curve in order to approximate a two-dimensional shape is not an efficient method for several reasons. Indeed, in order to approximate a shape with a lot of geometric features, one would need to increase the number of degrees of freedom, *i.e.* the number of control points. However, as is very well-known, increasing the degree of an approximating polynomial curve leads to a classical oscillation phenomenon and, in the particular case of a Bézier polynomial curve, it

leads to numerical instabilities (due to the ill-conditionness of the Bernstein-Vandermonde matrices, see, e.g., [27]). Moreover, since each control point has a global influence on the curve, one could not handle local complexities of a shape with a single Bézier curve. The classical idea is then to divide the curve in several Bézier curves of small degrees. This leads us to recall the following definition of piecewise Bézier curves.

Let $N \in \mathbb{N}^*$, $d \in \mathbb{N}^*$ and a set of $N(d+1)$ control points $P_{1,0}, \ldots, P_{1,d}, \ldots, P_{N,d}$ of $\mathbb{R}^2$ satisfying the *continuity relations* $P_{i,d} = P_{i+1,0}$ for every $i = 1, \ldots, N-1$.[3] The associated piecewise Bézier curve, denoted by $B([P_{1,0}, \ldots, P_{N,d}])$[4], is defined by

$$\forall t \in [0,1], \quad B([P_{1,0}, \ldots, P_{N,d}], t) := B([P_{i,0}, \ldots, P_{i,d}], Nt - i + 1), \qquad \text{if } t \in \left[\frac{i-1}{N}, \frac{i}{N}\right], \ i \text{ ranges from 1 to } N.$$

The global curve is then composed of N Bézier curves called *patches*. Note that a piecewise Bézier curve goes through $P_{i,0}$ and $P_{i,d}$ for all $i = 1, \ldots, N$. If $P_{1,0} = P_{N,d}$, the piecewise Bézier curve is said to be *closed*.

Remark 2.2. In practice we use cubic patches ($d = 3$) because they are sufficient in order to recover many geometrical situations, such as inflexion points (see Figure 2.2).

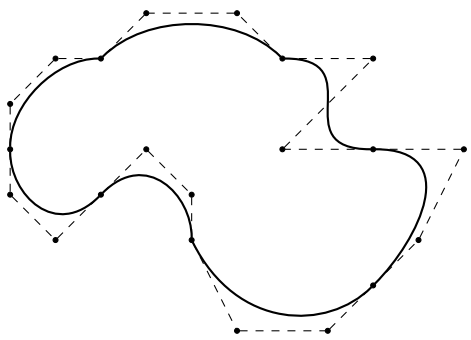


Figure 2.2. A closed piecewise Bézier curve composed of seven cubic patches.

Remark 2.3. In this paper, since each Bézier patch has the same degree d, the curve is said to be *uniform in degree*. Nevertheless one can easily build piecewise Bézier curves with patches of different degrees.

Adapting the proof of the classical Stone-Weierstrass theorem, one can prove the following result (which corresponds to a particular case of the classical Bishop theorem, see [6]).

Theorem 2.4. *Let $f \in \mathcal{C}([0,1], \mathbb{R}^2)$. For all $\varepsilon > 0$ and all $d \in \mathbb{N}^*$, there exist $N \in \mathbb{N}^*$ and a set of $N(d+1)$ control points $P_{1,0}, \ldots, P_{1,d}, \ldots, P_{N,d}$, satisfying the continuity relations, such that $\|f(t) - B([P_{1,0}, \ldots, P_{N,d}], t)\|_{\mathbb{R}^2} \leq \varepsilon$ for all $t \in [0,1]$.*

This result fully justifies the use of piecewise Bézier curves in order to approximate two-dimensional bounded shapes.

Remark 2.5. Recall that the use of polar coordinates, where the radius is expanded in a truncated Fourier series, is another common and efficient strategy in order to approximate two-dimensional

[3]The continuity relations guarantee the well-definedness and the continuity of the piecewise Bézier curve.

[4]One would note here a conflict in notations of a Bézier curve and of a piecewise Bézier curve. In the sequel no confusion is possible since we will only consider piecewise Bézier curves.

shapes (see, e.g., [1] in the context of inclusions detection). However it has two main drawbacks. Firstly it allows to represent only star-shaped domains and secondly, due to a classical oscillation phenomenon, it cannot represent rigorously straight lines (see, e.g., [14, Figure 5 p 140] in the context of inclusions detection). The use of piecewise Bézier curves is then an alternative in order to approximate non star-shaped domains and straight lines (see Section 4.3 for some numerical simulations in the context of inclusions detection). To conclude this remark, let us recall that the *flip procedure*, which is the main feature of this paper, is based on the detection of potential collisions between two parts of the boundary of the approximated shape (see Section 3 for more details). Thus, it is worth precising that a parameterization based on polar coordinates, where the radius is expanded in a truncated Fourier series, is not adapted to prevent such collisions, in contrary to a piecewise Bézier parameterization (see Section 3.2 for details).

3. Intersecting control polygons detection and flip procedure

In this paper we are interested in geometric two-dimensional shape approximation problems in which the target shape can have multiple connected components but the number of components is unknown. In such a case, starting a classical geometric approximation with a one-component initial shape may lead to the situation depicted in Figure 3.1, that is, the deformation flow makes the boundary evolve until it surrounds all the components of the target shape. This classical phenomenon tends to create a collision between two parts of the boundary of the approximated shape.

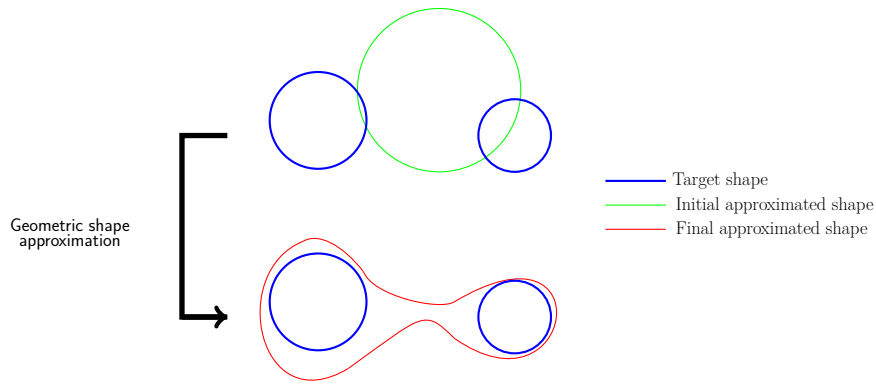


Figure 3.1. A geometric shape approximation of a two-component target shape starting from a one-component initial approximated shape. The final approximated shape surrounds the two components.

In this paper our major aim is to provide a simple and new concept (called *flip procedure*) that can be added to any shape approximation algorithm based on piecewise Bézier curves, and which allows to change the topology of the approximated shape. Precisely, the flip procedure allows to divide a one-component shape into a two-component shape.

Remark 3.1. In this paper, we focus on piecewise cubic Bézier curve ($d = 3$, see Remark 2.2). However, this method could be easily extended to any $d \geq 2$.

3.1. Overview

Let us consider a general geometric shape approximation algorithm in which the boundary of the approximated shape is parameterized by a piecewise cubic Bézier curve. It starts from a one-component

initial shape ω_0 and produces a sequence of one-component shapes $(\omega_k)_{k\geq 0}$ by deforming the boundary at each step. Our idea consists in two phases (that are summarized in Figure 3.2):

(1) check, at each step of the approximation algorithm, if the current shape ω_k is in the situation depicted in Figure 3.1, that is, if two parts of the boundary are very close to each other. The parameterization by piecewise Bézier curves allows us to prevent such a situation by looking for intersecting control polygons. This procedure will be called *intersecting control polygons detection* and will be detailed in Section 3.2;

(2) if some control polygons intersect each other, we apply the flip procedure in order to obtain a two-component shape by keeping unchanged all other control polygons. The flip procedure is detailed in Section 3.3.

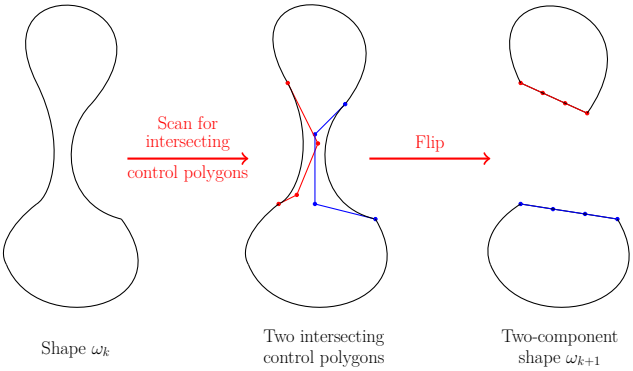
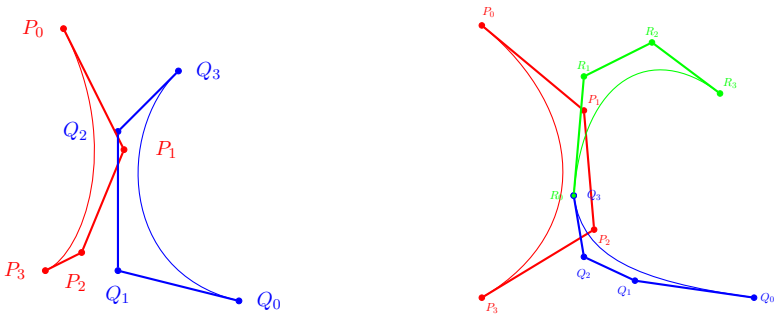


Figure 3.2. Overview of the complete procedure.

For the sake of simplicity of presentation, we will assume that, during the evolution of the shape ω_k, situations of intersection between control polygons have always the same pattern:

(A1) only two situations of intersection between control polygons are possible: either one control polygon intersects exactly another one, or one control polygon intersects exactly two consecutive ones (see Figure 3.3);



(a) Case with two control polygons

(b) Case with three control polygons

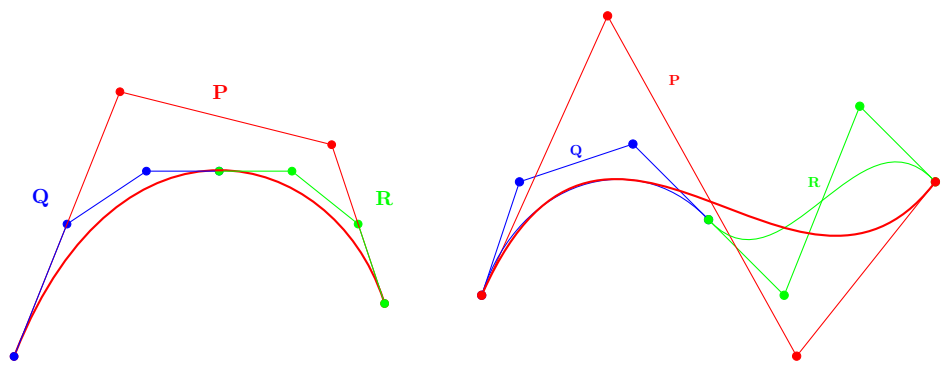
Figure 3.3. Assumption (A1).

(A2) futhermore, at each iteration, at most one situation of intersection occurs.

Assumptions (A1)-(A2) are ordinarily satisfied in practice, in particular in all numerical simulations we made (see Section 4.3). Removing these assumptions should not involve new deep ideas or new concept, however the complete algorithmic description and implementation would become considerably tricky and challenging. It is not our aim to deal with this issue in this paper.

Remark 3.2. In Figure 3.3, note that collisions between two patches are not excluded. In that case, we say that the shape ω_k is *self-intersecting*. However, from Assumption (A3) enunciated later, this situation does not jeopardize the integrity of Algorithm $\mathcal{A}$ presented in Section 4.3.1 (see in particular Step 3(b)ii of the algorithm).

Remark 3.3 (Controlling the size of the patches using split and merge functions)**.** In order to maintain numerical stability, one should control the size of the control polygons (that is, the diameter of their convex hull) in a range $[S_{\min}, S_{\max}]$ with $0 < S_{\min} < S_{\max}$. This avoids to deal with very large patches and/or very small ones. To this end, the diameter of the convex hull of each control polygon can be computed at each iteration. If a control polygon does not satisfy the size condition, it is either split into two control polygons or merged with a neighbor one. The split and merge functions (see [26] for more details) are inverse operations and both use interpolation in order to compute the new control polygons (see Figure 3.4). The split function divides a control polygon into two. Precisely, it interpolates the first half of the patch and, in a second time, interpolates the other half. Since each half of the patch is a Bézier curve, the shape is not modified after a split. The merge function is the reverse operation. From two consecutive control polygons $\mathbf{Q}$ and $\mathbf{R}$, it computes one patch that interpolates the four points $B(\mathbf{Q}, 0)$, $B(\mathbf{Q}, \frac{2}{3})$, $B(\mathbf{R}, \frac{1}{3})$ and $B(\mathbf{R}, 1)$. Then one starts from seven control points and ends with four. Note that merging polygons modifies slightly the boundary.



(a) $(Q, R) \longleftarrow \text{Split}(P)$ - The boundary is not modified.

(b) $P \longleftarrow \text{Merge}(Q, R)$ - The boundary is slightly modified.

Figure 3.4. Examples of the split and merge functions.

3.2. **Intersecting control polygons detection**

Checking if each control polygon intersects another one may be very expensive in terms of computations. Axis-Aligned Bounding Boxes (AABBs) are a very common tool in Computer Graphics and Computational Geometry in order to detect the collision of two objects (see, e.g., [18]), with a relatively low computational cost. AABB is defined as the smallest rectangle, whose sides are aligned with the axes, containing the control polygon (see Figure 3.5).

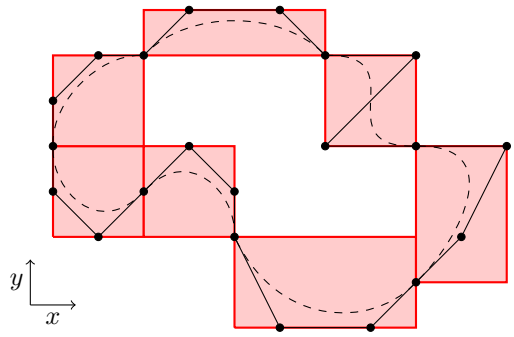


Figure 3.5. AABBs of control polygons.

A necessary condition for two intersecting control polygons is clearly the intersection of their respective AABBs. As a consequence, instead of looking directly for intersecting control polygons, we first look for intersecting AABBs. Thus, the intersecting control polygons detection consists in two steps:

(1) we first list all the pairs of intersecting AABBs;

(2) in a second time, we check these pairs in order to see if the associated control polygons intersect. To do so, we directly check the 9 segment-segment intersections of the polygons (see, e.g., [28, p. 28-30]).

Finally, each pair of intersecting control polygons will be given as input to the flip procedure detailed in the following section.

3.3. **The flip procedure**

From Assumption (A1), only two cases of intersecting control polygons are considered (see Figure 3.3). The flip procedure described in this section is a simple tool that can be easily implemented and that handles these two situations.

First case: two intersecting polygons. From $\mathbf{P} = \{P_0, P_1, P_2, P_3\}$ and $\mathbf{Q} = \{Q_0, Q_1, Q_2, Q_3\}$ being two intersecting polygons of a same connected component, the flip procedure builds two new polygons as follows, (see Figure 3.6):

$$\left\{P_0, P_0 + \frac{1}{3}\overrightarrow{P_0Q_3}, P_0 + \frac{2}{3}\overrightarrow{P_0Q_3}, Q_3\right\} \qquad \text{and} \qquad \left\{Q_0, Q_0 + \frac{1}{3}\overrightarrow{Q_0P_3}, Q_0 + \frac{2}{3}\overrightarrow{Q_0P_3}, P_3\right\}.$$

Second case: three intersecting polygons. The case with three control polygons is very similar. From $\mathbf{P} = \{P_0, P_1, P_2, P_3\}$ being a control polygon intersecting two consecutive ones $\mathbf{Q} = \{Q_0, Q_1, Q_2, Q_3\}$ and $\mathbf{R} = \{R_0, R_1, R_2, R_3\}$, the flip procedure builds two new polygons as follows, (see Figure 3.7):

$$\left\{P_0, P_0 + \frac{1}{3}\overrightarrow{P_0R_3}, P_0 + \frac{2}{3}\overrightarrow{P_0R_3}, R_3\right\} \qquad \text{and} \qquad \left\{Q_0, Q_0 + \frac{1}{3}\overrightarrow{Q_0P_3}, Q_0 + \frac{2}{3}\overrightarrow{Q_0P_3}, P_3\right\}.$$

For the sake of simplicity of presentation, we will assume that the following hypothesis is satisfied:

(A3) the flip procedure does not produce intersecting control polygons.

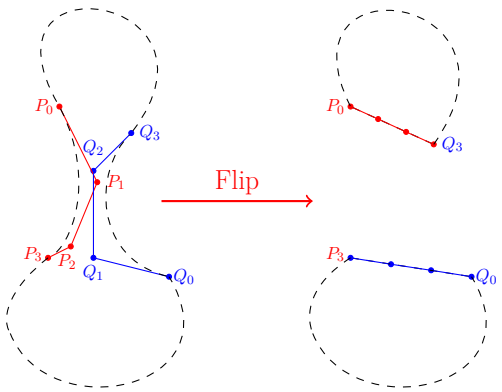


Figure 3.6. Flip procedure - Case of two control polygons.

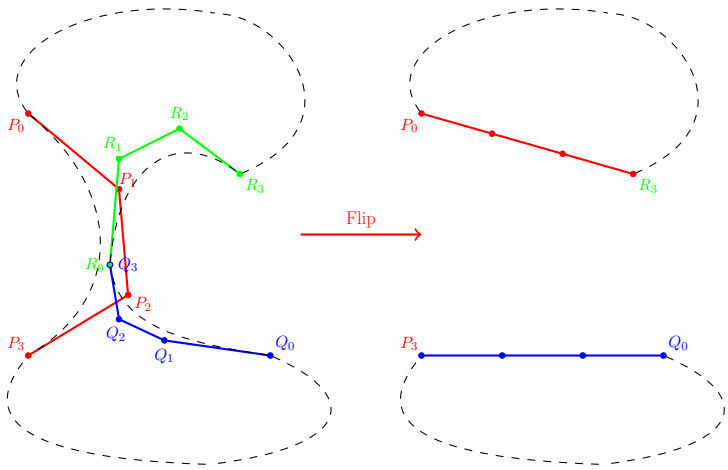


Figure 3.7. Flip procedure - Case of three control polygons.

In the same spirit of Assumptions (A1)-(A2), Assumption (A3) is ordinarily satisfied in practice, in particular in all numerical simulations we made (see Section 4.3).

4. **Application to multiple-inclusion detection**

This section focuses on the problem of reconstructing numerically an obstacle ω_{ex} living in a larger bounded domain Ω of $\mathbb{R}^2$ from boundary measurements. Our aim is in particular to test the flip procedure introduced in this paper in the case where ω_{ex} is a two-component obstacle (see Section 4.3.4).

In order to solve numerically the above inverse obstacle problem, we will actually consider a shape optimization problem, by minimizing a shape cost functional. In this paper we use the classical geometrical shape optimization approach, based on shape derivatives and on a shape gradient descent method. We refer to the classical books of Henrot *et al.* [23] and of Sokołowski *et al.* [37] for more details on the techniques of shape differentiability.

Let us fix some notations that will be used in this section. We denote by L^p, $\mathrm{W}^{m,p}$ and H^s the usual Lebesgue and Sobolev spaces. We note in bold the vectorial functions and spaces, such as $\mathbf{W}^{m,p}$. Let Ω

be a nonempty bounded and connected open set of $\mathbb{R}^2$ with a $C^{2,1}$ boundary and let $g \in \mathrm{H}^{5/2}(\partial\Omega)$ such that $g \neq 0$. We denote by $\mathbf{n}$ the external unit normal to $\partial\Omega$, and for a smooth enough function u, we denote by $\partial_{\mathbf{n}} u$ the normal derivative of u.

Let $d_0 > 0$ be fixed (small). In the sequel $\mathcal{O}_{d_0}$ stands for the set of all open subsets ω strictly included in Ω, with a $C^{2,1}$ boundary, such that the distance $\mathrm{d}(x, \partial\Omega)$ from x to the compact $\partial\Omega$ is strictly greater than d_0 for all $x \in \omega$, and such that $\Omega \backslash \overline{\omega}$ is connected. Finally we also introduce Ω_{d_0} an open set with a C^∞ boundary such that

$$\{x \in \Omega\,;\ \mathrm{d}(x, \partial\Omega) > d_0/2\} \subset \Omega_{d_0} \subset \{x \in \Omega\,;\ \mathrm{d}(x, \partial\Omega) > d_0/3\}\,.$$

4.1. **Problem setting**

We focus on the following inverse problem. Assume that an unknown obstacle $\omega_{\mathrm{ex}} \in \mathcal{O}_{d_0}$ is located inside Ω. We consider hereafter the Laplace equation in $\Omega \backslash \overline{\omega_{\mathrm{ex}}}$ with homogeneous Dirichlet boundary condition on $\partial\omega_{\mathrm{ex}}$ and non-homogeneous Dirichlet boundary condition on $\partial\Omega$. Precisely we denote by $u_{\mathrm{ex}} \in \mathrm{H}^1(\Omega \backslash \overline{\omega_{\mathrm{ex}}})$ the unique solution of the problem

$$\left\{ \begin{array}{rcll} -\Delta u_{\mathrm{ex}} & = & 0 & \text{in } \Omega \backslash \overline{\omega_{\mathrm{ex}}}, \\ u_{\mathrm{ex}} & = & g & \text{on } \partial\Omega, \\ u_{\mathrm{ex}} & = & 0 & \text{on } \partial\omega_{\mathrm{ex}}. \end{array} \right. \tag{4.1}$$

Since $g \in \mathrm{H}^{5/2}(\partial\Omega)$ and $\Omega \backslash \overline{\omega_{\mathrm{ex}}}$ has a $C^{2,1}$ boundary, note that u_{ex} belongs to $\mathrm{H}^3(\Omega \backslash \overline{\omega_{\mathrm{ex}}})$. Our main purpose is to reconstruct the unknown shape ω_{ex}, assuming that a measurement is done on the exterior boundary $\partial\Omega$. Precisely we assume in this paper that we know exactly the value of the measure $f_b := \partial_{\mathbf{n}} u_{\mathrm{ex}} \in \mathrm{H}^{3/2}(\partial\Omega)$ on $\partial\Omega$. Thus, for a given nontrivial Cauchy pair $(g, f_b) \in \mathrm{H}^{5/2}(\partial\Omega) \times \mathrm{H}^{3/2}(\partial\Omega)$, we are interested in the following geometric inverse problem:

find $\omega \in \mathcal{O}_{d_0}$ and $u \in \mathrm{H}^1(\Omega \backslash \overline{\omega}) \cap C^0(\Omega \backslash \overline{\omega})$ which satisfies the overdetermined system

$$\left\{ \begin{array}{rcll} -\Delta u & = & 0 & \text{in } \Omega \backslash \overline{\omega}, \\ u & = & g & \text{on } \partial\Omega, \\ \partial_{\mathbf{n}} u & = & f_b & \text{on } \partial\Omega, \\ u & = & 0 & \text{on } \partial\omega. \end{array} \right. \tag{4.2}$$

The existence of a solution is trivial since we assume that the measurement f_b is exact. From the classical Holmgren's theorem (see, e.g., [24]) one can obtain an identifiability result for this inverse problem which claims that the solution is unique. This fundamental question about uniqueness of a solution to the overdetermined problem (4.2) was deeply studied, see for example [7, Theorem 1.1], [16, Theorem 5.1] or also [17, Proposition 4.4 p. 87]. We recall the identifiability result for the reader's convenience.[5]

Theorem 4.1. *The domain ω and the function u that satisfy* (4.2) *are uniquely defined by the Cauchy data $(g, f_b) \neq (0, 0)$.*

Remark 4.2. Actually we could assume that the measurement f_b is done only on a nonempty subset O of $\partial\Omega$. All the presented result can be adapted to this case (see, e.g., [10]).

In order to solve the inverse problem (4.2) we will actually focus on the shape optimization problem

$$\omega^* \in \operatorname*{argmin}_{\omega \in \mathcal{O}_{d_0}} J(\omega), \tag{4.3}$$

[5]Note that Theorem 4.1 is true even with the following weaker assumptions: $(g, f_b) \in \mathrm{H}^{1/2}(\partial\Omega) \times \mathrm{H}^{-1/2}(\partial\Omega)$ and ω has only a continuous boundary (see [7, Theorem 1.1]).

where J is the nonnegative least-square functional defined by

$$J(\omega) := \int_{\partial\Omega} |\partial_{\mathbf{n}} u_\omega - f_b|^2 \,,$$

where $u_\omega \in \mathrm{H}^3(\Omega\backslash\overline{\omega})$ is the unique solution of the problem

$$\left\{ \begin{array}{rcll} -\Delta u_\omega & = & 0 & \text{in } \Omega\backslash\overline{\omega}, \\ u_\omega & = & g & \text{on } \partial\Omega, \\ u_\omega & = & 0 & \text{on } \partial\omega. \end{array} \right. \tag{4.4}$$

Indeed, the identifiability result ensures that $J(\omega) = 0$ if and only if $\omega = \omega_{\mathrm{ex}}$. Finally, in order to solve numerically the shape optimization problem (4.3), we will now compute the shape gradient of the cost functional J and apply a classical gradient descent method.

4.2. **Computation of the shape gradient.**

In order to define shape derivatives, we will use the *Hadamard's method*. We first introduce the space of admissible deformations given by

$$\boldsymbol{U} := \{\boldsymbol{V} \in \mathbf{W}^{3,\infty};\ \mathrm{Supp}\ \boldsymbol{V} \subset \overline{\Omega_{d_0}}\}. \tag{4.5}$$

In particular we are interested in the shape gradient of J defined by

$$\mathrm{D}J(\omega) \cdot \boldsymbol{V} := \lim_{t\to 0} \frac{J\big((\mathbf{I} + t\boldsymbol{V})(\omega)\big) - J(\omega)}{t},$$

for every $\omega \in \mathcal{O}_{d_0}$ and every $\boldsymbol{V} \in \boldsymbol{U}$. For sake of completeness, we recall the proof of the following result in Appendix A.

Proposition 4.3. *The least-square functional J is differentiable at $\omega \in \mathcal{O}_{d_0}$ in the direction $\boldsymbol{V} \in \boldsymbol{U}$ with*

$$\mathrm{D}J(\omega) \cdot \boldsymbol{V} = -\int_{\partial\omega} \partial_{\mathbf{n}} u_\omega \, \partial_{\mathbf{n}} w_\omega \, (\boldsymbol{V} \cdot \mathbf{n}) \,, \tag{4.6}$$

where $w_\omega \in \mathrm{H}^1(\Omega\backslash\overline{\omega})$ is the unique solution of the adjoint problem given by

$$\left\{ \begin{array}{rcll} -\Delta w_\omega & = & 0 & \text{in } \Omega\backslash\overline{\omega}, \\ w_\omega & = & 2\,(\partial_{\mathbf{n}} u_\omega - f_b) & \text{on } \partial\Omega, \\ w_\omega & = & 0 & \text{on } \partial\omega. \end{array} \right. \tag{4.7}$$

From the above explicit formulation of the shape gradient of J, we are now in a position to implement some numerical simulations based on a classical gradient descent method and we include the flip procedure introduced in this paper in order to detect in particular a multiple-component obstacle.

4.3. **Numerical simulations**

Before coming to numerical simulations, let us recall that many difficulties can be encountered in order to solve numerically Problem (4.3), as explained in [1, Theorem 1] (see also [5, Proposition 2.4]). Indeed, the gradient has not a uniform sensitivity with respect to the deformation directions. However, we use in this paper a parametric model of shape variations using piecewise Bézier curves which corresponds to a regularization method (as the truncated Fourier series used in [1]) allowing to overcome the ill-posedness of the inverse problem and then to solve it numerically.

Note that we use here piecewise Bézier curves that do not satisfy the $C^{2,1}$-regularity assumption made in the previous section.[6] This regularity hypothesis is sufficient in order to prove rigorously the

[6]However one could retrieve the $C^{2,1}$-regularity hypothesis by imposing some additional constraints on the control points of the piecewise Bézier curves.

previous theoretical results. In this section dedicated to numerical simulations, we make the choice to not deal with this regularity issue since we still observe relatively good numerical reconstructions of obstacles. Besides, let us mention that the issue of knowing if the computed shape gradient (computed in particular from approximations of shapes by mesh and of solutions of PDEs by a finite element method) is an actual approximation of the genuine one is a fully-fledged question and it is not our aim to address this issue in this paper.

4.3.1. *Framework for the numerical simulations*

The numerical simulations presented hereafter are performed in the two-dimensional case using the finite element library *FreeFem++* (see [22]). The exterior boundary $\partial\Omega$ is assumed to be the circle centered in the origin and of radius 10 and we consider the exterior Dirichlet boundary condition $g = 100$. In order to get a suitable measure f_b, we use a *synthetic data*, that is, we fix a shape ω_{ex} and solve Problem (4.1) using a finite element method (here P2 finite element discretization) and extract the measurement f_b by computing $\partial_{\mathbf{n}} u_{\text{ex}}$ on $\partial\Omega$.

Then we use a P1 finite element discretization to solve Problems (4.4) and (4.7) with 50 discretization points for both the exterior boundary and each cubic Bézier patch describing the shape ω. In order to numerically solve the optimization problem (4.3), we use the following classical gradient descent algorithm and we include the flip procedure at Step (3).

Algorithm $\mathcal{A}$

(1) Fix $k = 0$, fix an initial shape ω_0, fix a maximal number $M \in \mathbb{N}^*$ of iterations and fix $\lambda \geq 1$ a given tolerance coefficient for the flip procedure (see Step (3(b)ii), λ should be chosen close to 1).

(2) Control the size of the patches of ω_k (see Remark 3.3).

(3) Scan ω_k looking for intersecting control polygons (see Section 3.2):

  (a) in the case of no intersecting control polygons, go to Step (4);
  (b) in the case of intersecting control polygons:
    (i) apply the flip procedure and obtain a multiple-component shape $\omega_k^1 \cup \omega_k^2$; recall that $\omega_k^1 \cup \omega_k^2$ is not self-intersecting from Assumption (A3);
    (ii) compute $J(\omega_k^1 \cup \omega_k^2)$ and $J(\omega_k)$, and set $J(\omega_k) = +\infty$ if ω_k is self-intersecting:
      (A) if $J(\omega_k^1 \cup \omega_k^2) < \lambda J(\omega_k)$, do $\omega_k \leftarrow \omega_k^1 \cup \omega_k^2$;
      (B) else, go to Step (4).

(4) Solve Problems (4.4) and (4.7) with $\omega = \omega_k$.

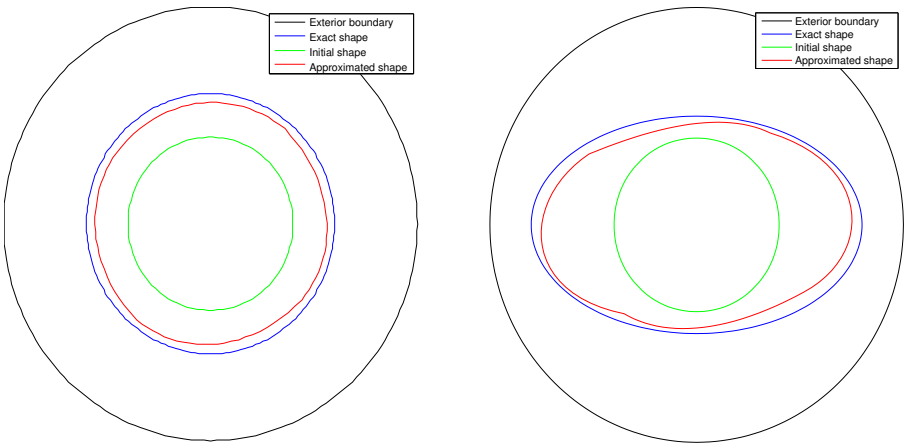
(5) Compute the shape gradient $\mathrm{D}J(\omega_k)$ from Formula (4.6).

(6) Move the control points of the shape, that is, do $\omega_{k+1} \leftarrow \omega_k - \alpha_k \mathrm{D}J(\omega_k)$, where α_k is a small positive coefficient chosen, e.g., by a classical line search.

(7) Do $k \leftarrow k + 1$ and get back to Step (2) while $k < M$.

4.3.2. *First simulations: detection of smooth and convex shapes*

We first test Algorithm $\mathcal{A}$ on the problem of detecting one smooth convex object. Precisely, we begin by detecting the circle centered at the origin and of radius 6 and the ellipse $\{(8\cos\theta,\ 5\sin\theta), \theta \in [0, 2\pi]\}$ using four cubic Bézier patches. Numerical simulations are performed and depicted in Figure 4.1.



(a) Detection of a circle (b) Detection of an ellipse

Figure 4.1. Detection of convex and smooth obstacles.

4.3.3. *Detection of a non-smooth shape and of a non-convex shape*

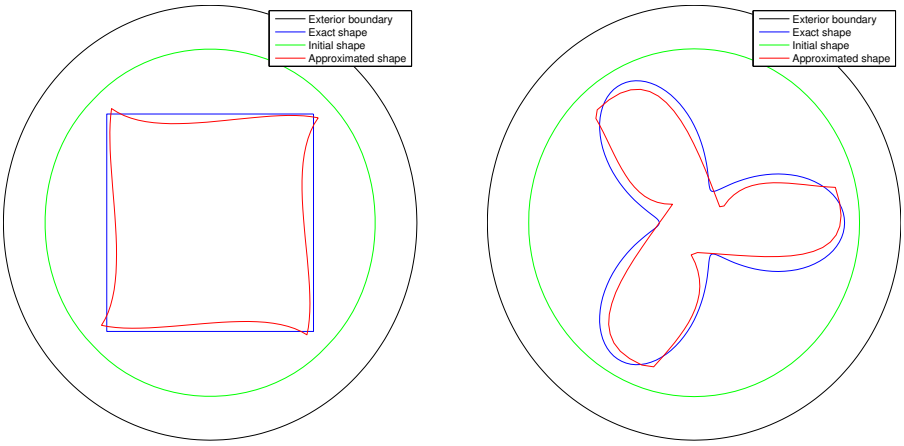
We test now Algorithm $\mathcal{A}$ on the problem of detecting a non-smooth shape and of detecting a non-convex shape (see Figure 4.2). Precisely we first consider the square of side 10 and centered at the origin and we use four cubic Bézier patches. As one can see in Figure 4.2(a), each Bézier patch detects a side of the square. Figure 4.3 shows the decrease of the objective function during the simulation. Secondly, in Figure 4.2(b), we consider the non-convex shape parameterized by $\{(2.8(1.6+\cos(3\theta))\cos(\theta), 2.8(1.6+\cos(3\theta))\cos(\theta)), \theta \in [0, 2\pi]\}$, using six cubic Bézier patches.[7]

4.3.4. *Detection of a two-component obstacle starting from a one-component shape*

In this section we test the flip procedure introduced in Section 3 in order to detect a two-component shape starting from a one-component initial shape. We consider two circles of radius 2 centered at $(-4, -4)$ and $(4, 4)$. We present different states of the algorithm in Figure 4.4. The initial Bézier shape consists in a single component with four cubic Bézier patches, located at the center (Figure 4.4(a)). The shape grows and surrounds the two objects until two control polygons intersect each other (Figure 4.4(b)). The flip procedure is performed and the shape is divided in two connected components (Figure 4.4(c)). At the end, the algorithm provides an approximation of the two obstacles (Figure 4.4(d)).

Figure 4.5 depicts the evolution of the objective function during this simulation. One can note a change of behavior after Iteration 133 which corresponds to the performance of the flip procedure. Precisely, the algorithm finds in a first place a local minimizer at Iteration 13, which corresponds

[7]This shape is also considered in [13, Figure 4] where authors obtained the convex hull of the shape. However, note that the authors used a different method where the descent direction is obtained by solving a boundary value problem involving the kernel of the shape gradient.



(a) Detection of a square (b) Detection of a non-convex shape.

Figure 4.2. Detection of a non-smooth obstacle and of a non-convex obstacle.

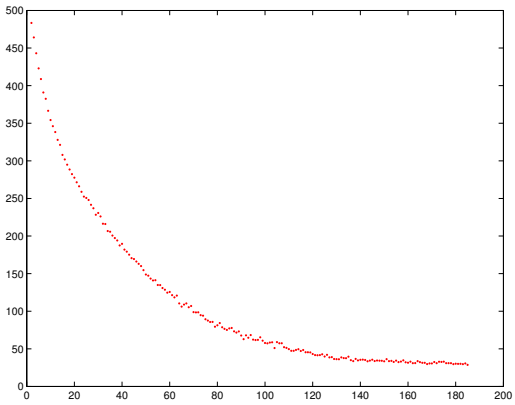
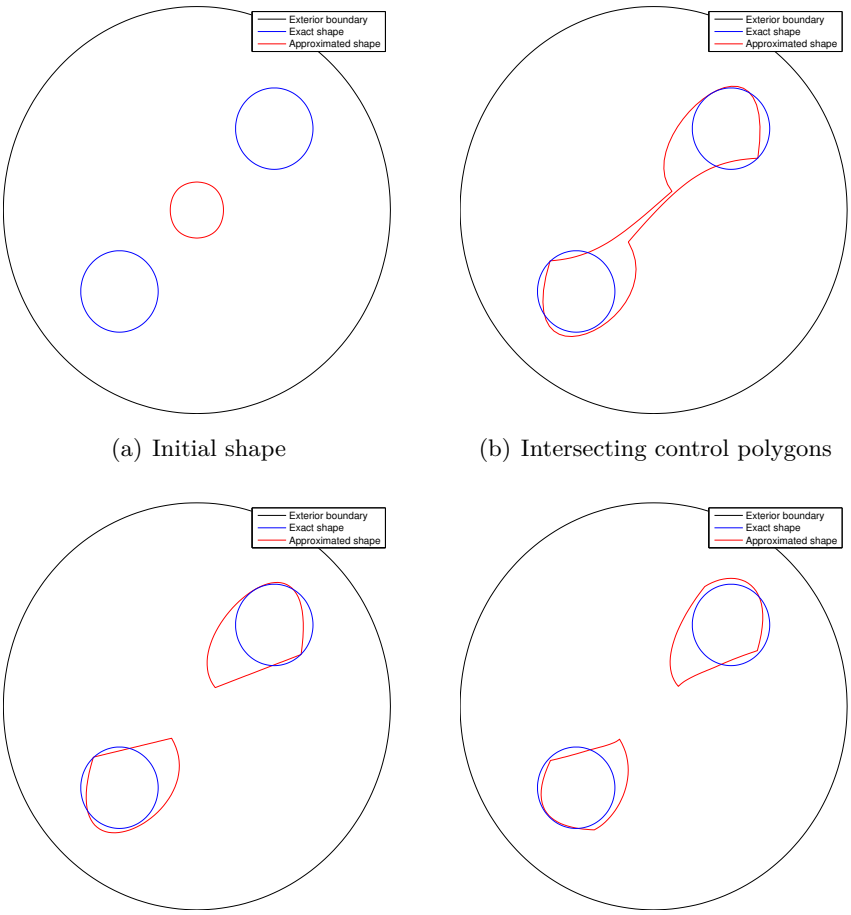


Figure 4.3. Evolution of the objective function for the detection of the square.

to a one-component minimizer. After oscillations around this local minimum, the flip procedure is performed and the functional decreases and stabilizes around a two-component minimizer.

4.3.5. *Checking the objective function value after a flip procedure*

In Algorithm $\mathcal{A}$, Step (3(b)ii) makes sure that, whenever a flip procedure is performed, the objective function value does not significantly increase. If $J(\omega_k^1 \cup \omega_k^2) \geq \lambda J(\omega_k)$ (for instance $\lambda = 1.1$), then we consider that adding another component to the shape is not a wise choice and we cancel the flip procedure. This situation typically occurs when the target shape has a single component with a very thin part (*i.e.* two parts of its boundary are very close to each other). In such a case Algorithm $\mathcal{A}$ probably leads to two control polygons intersecting each other and to a flip performance, while the target shape has a single component. We present an example of such a situation in Figure 4.6. The obstacle is composed of one component with a very thin part and the current shape ω_k of the algorithm has two control polygons intersecting each other. The objective function value before the flip procedure is $J(\omega_k) = 3211$ and after the flip procedure, it has increased to $J(\omega_k^1 \cup \omega_k^2) = 3579$. Since the ratio is greater than λ, the algorithm cancels the flip procedure and goes to Step (4).



(a) Initial shape (b) Intersecting control polygons

(c) Flip procedure (d) Final shape

Figure 4.4. Detection of two obstacles starting from a one-component shape

Remark 4.4. In Algorithm $\mathcal{A}$, if the previous situation ω_k is restored after the performance of a flip procedure (that is, in the case 3(b)iiB), then the obstacle is highly likely composed of one component with a very thin part. Next, the gradient flow makes evolve the approximated shape ω_k with a small deformation step to a better approximation ω_{k+1} of the target shape. This would lead to a new perfomance of a flip procedure on ω_{k+1}. Actually, in such a situation, a flip procedure is performed/cancelled at each iteration until the end.

5. Conclusion and perspectives

In this paper we studied the use of a piecewise Bézier parameterization for the representation of two-dimensional shapes in geometric approximation based on successive shape deformations. We proposed procedures in order to manipulate this parameterization and showed how to manage changes of topology and so multiple-component shape approximations. We applied this approach to a problem of multiple-inclusion detection and performed numerical simulations using *FreeFem++*. The computational efficiency of the method arises from the simplicity and the flexibility of the proposed parameterization.

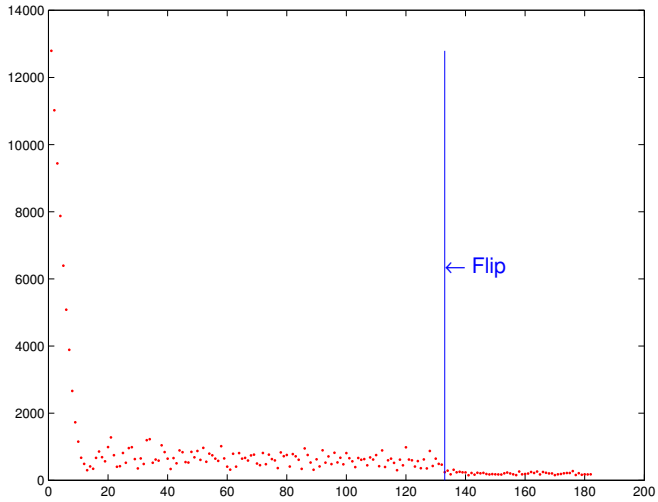
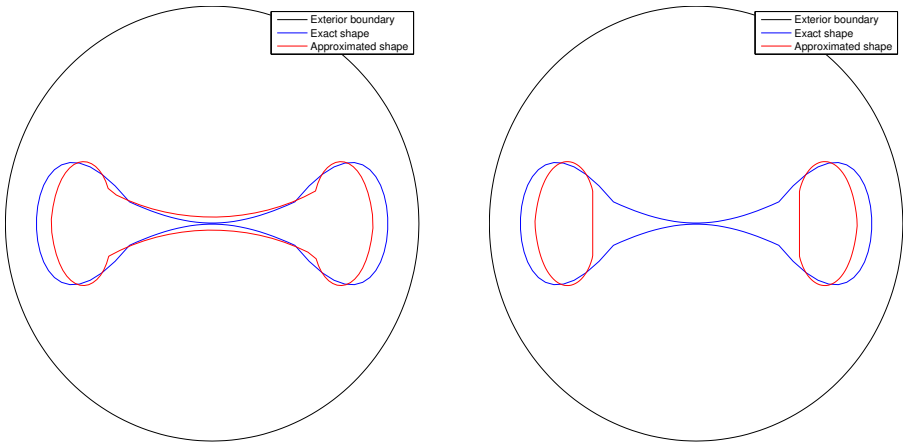


Figure 4.5. Evolution of the objective function for the detection of a two-component shape.



(a) Before the flip procedure, $J(\omega_k) = 3211$

(b) After the flip procedure, $J(\omega_k^1 \cup \omega_k^2) = 3579$

Figure 4.6. The objective function value significantly increases whenever the flip procedure is not needed.

We considered in this paper a two-dimensional problem, but the challenging extension to the three-dimensional case may be interesting and the algorithmic contents could be generalized. As a conclusion, the implementation described here was made for an experimental purpose and a complete and optimized implementation may be proposed.

Acknowledgment. The authors would like to thank the anonymous reviewers for their valuable comments and suggestions in order to improve the quality of the paper.

Appendix A. Proof of Proposition 4.3

We detail here the classical proof of Proposition 4.3 for the reader's convenience. For any $\boldsymbol{V} \in \boldsymbol{U}$ (where $\boldsymbol{U}$ is defined by (4.5)), we introduce the perturbed domain $\omega_t := (\mathbf{I}+t\boldsymbol{V})(\omega)$ and the functional j defined for all $t \in [0,T)$ by $j(t) := J(\omega_t)$ and we consider the unique solution $u_t \in \mathrm{H}^3(\Omega\backslash\overline{\omega_t})$ of the perturbed problem

$$\left\{\begin{array}{rcll} -\Delta u_t & = & 0 & \text{in } \Omega\backslash\overline{\omega_t}, \\ u_t & = & g & \text{on } \partial\Omega, \\ u_t & = & 0 & \text{on } \partial\omega_t. \end{array}\right.$$

Let us recall the definition of the shape derivative in our situation (see [23] for details). We introduce

$$\boldsymbol{\mathcal{U}} := \left\{\boldsymbol{\theta} \in \boldsymbol{U};\ \|\boldsymbol{\theta}\|_{3,\infty} < \min\left(\frac{d_0}{3}, 1\right)\right\}$$

and, for any $\boldsymbol{\theta} \in \boldsymbol{\mathcal{U}}$, we consider the unique solution $u_\theta \in \mathrm{H}^3(\Omega\backslash\overline{\omega_\theta})$ of the perturbed problem

$$\left\{\begin{array}{rcll} -\Delta u_\theta & = & 0 & \text{in } \Omega\backslash\overline{\omega_\theta}, \\ u_\theta & = & g & \text{on } \partial\Omega, \\ u_\theta & = & 0 & \text{on } \partial\omega_\theta, \end{array}\right.$$

where $\omega_\theta := (\mathbf{I}+\boldsymbol{\theta})(\omega)$. Then,

- if the mapping $\boldsymbol{\theta} \in \boldsymbol{\mathcal{U}} \mapsto u_\theta \circ (\mathbf{I}+\boldsymbol{\theta}) \in \mathrm{H}^1(\Omega\backslash\overline{\omega})$ is Fréchet differentiable at $\mathbf{0}$, we say that $\boldsymbol{\theta} \mapsto u_\theta$ possesses a *total first variation* (or derivative) at $\mathbf{0}$. In such a case, this total first derivative at $\mathbf{0}$ in the direction $\boldsymbol{\theta}$ is denoted by $\dot{u}_\theta$ and is called *material derivative* (or *Lagrangian derivative)*;

- if, for every $\mathscr{D} \subset\subset \Omega\backslash\overline{\omega}$, the mapping $\boldsymbol{\theta} \in \boldsymbol{\mathcal{U}} \mapsto u_{\theta\,|\mathscr{D}} \in \mathrm{H}^1(\mathscr{D})$ is Fréchet differentiable at $\mathbf{0}$, we say that $\boldsymbol{\theta} \mapsto u_\theta$ possesses a *local first variation* (or derivative) at $\mathbf{0}$. In such a case, this local first derivative at $\mathbf{0}$ in the direction $\boldsymbol{\theta}$ is denoted by u'_θ, is called *shape derivative* (or *Eulerian derivative*) and is well defined in the whole domain $\Omega\backslash\overline{\omega}$:

$$u'_\theta = \frac{d}{dt}(u_{t\boldsymbol{\theta}\,|\mathscr{D}})\,|_{t=0} \qquad \text{in each } \mathscr{D} \subset\subset \Omega\backslash\overline{\omega}.$$

In the sequel, let $\boldsymbol{V} \in \boldsymbol{\mathcal{U}}$ and let u' be the local first variation u'_V which is referred as the shape derivative of the state.

The differentiability of the cost functional J is directly obtained from the existence of the shape derivative of the state u given for example in [23, Theorem 5.3.1]. Notice that in [23, Theorem 5.3.1], the result claims the differentiability of $t \in [0,T] \mapsto \tilde{u}_t \in \mathrm{L}^2(\Omega)$, where $\tilde{u}_t$ is an extension of u_t in Ω. Since we want to obtain the differentiability of $t \in [0,T] \mapsto \tilde{u}_t \in \mathrm{H}^2(\Omega)$ (in order to differentiate properly the functional J), we have here to work with the mentioned spaces, that is with domains with a $C^{2,1}$ boundary (and not only Lipschitz) and perturbations $\boldsymbol{V}$ which belong to $\mathbf{W}^{3,\infty}(\mathbb{R}^2)$ (and not only to $\mathbf{W}^{1,\infty}(\mathbb{R}^2)$).

Moreover we can easily characterize the shape derivative $u' \in \mathrm{H}^1(\Omega\backslash\overline{\omega})$ as the solution of the following problem (see again for example [23, Theorem 5.3.1]):

$$\left\{\begin{array}{rcll} -\Delta u' & = & 0 & \text{in } \Omega\backslash\overline{\omega}, \\ u' & = & 0 & \text{on } \partial\Omega, \\ u' & = & -\partial_{\mathbf{n}} u\,(\boldsymbol{V}\cdot\mathbf{n}) & \text{on } \partial\omega. \end{array}\right. \tag{A.1}$$

Then by differentiation under the sum sign, we obtain

$$j'(0) = 2\int_{\partial\Omega} \partial_{\mathbf{n}} u'(\partial_{\mathbf{n}} u - f_b).$$

Using the weak formulation of Problem (A.1) solved by u' with w as a test function, we obtain

$$\int_{\Omega\setminus\overline{\omega}} \nabla u' \cdot \nabla w - \int_{\partial(\Omega\setminus\overline{\omega})} w\, \partial_{\mathbf{n}} u' = 0$$

and using the weak formulation of the adjoint Problem (4.7) solved by w with u' as a test function, we obtain

$$\int_{\Omega\setminus\overline{\omega}} \nabla w \cdot \nabla u' - \int_{\partial(\Omega\setminus\overline{\omega})} u'\, \partial_{\mathbf{n}} w = 0.$$

Finally, using the boundary conditions, the proof is complete.

Appendix B. Detection of one obstacle starting from a two-component shape

In this paper we have introduced the flip procedure as a method that enables to divide a one-component shape into a two-component shape. Actually the flip procedure can be easily adapted in order to perform the reverse operation, that is, to merge a two-component shape into a one-component shape (see Figure B.1).

We focus now on the detection of the one-component shape $\{(4\cos\theta, 6 + 2.5\sin\theta), \theta \in [0, 2\pi]\}$ and we start Algorithm $\mathcal{A}$ with a two-component shape. We present different states of the algorithm in Figure B.2. At the end, the algorithm provides an approximation of the one-component obstacle.

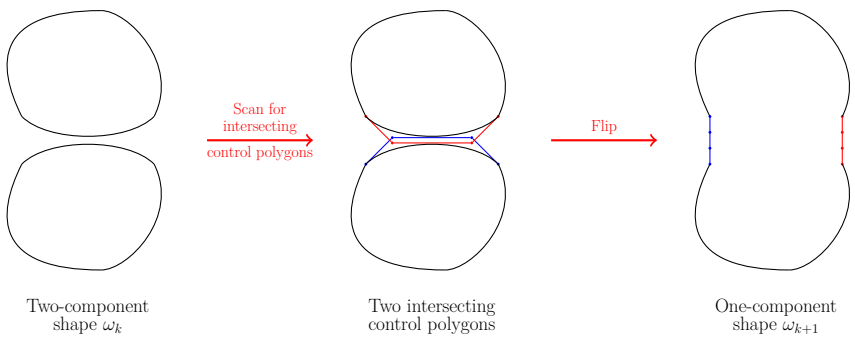
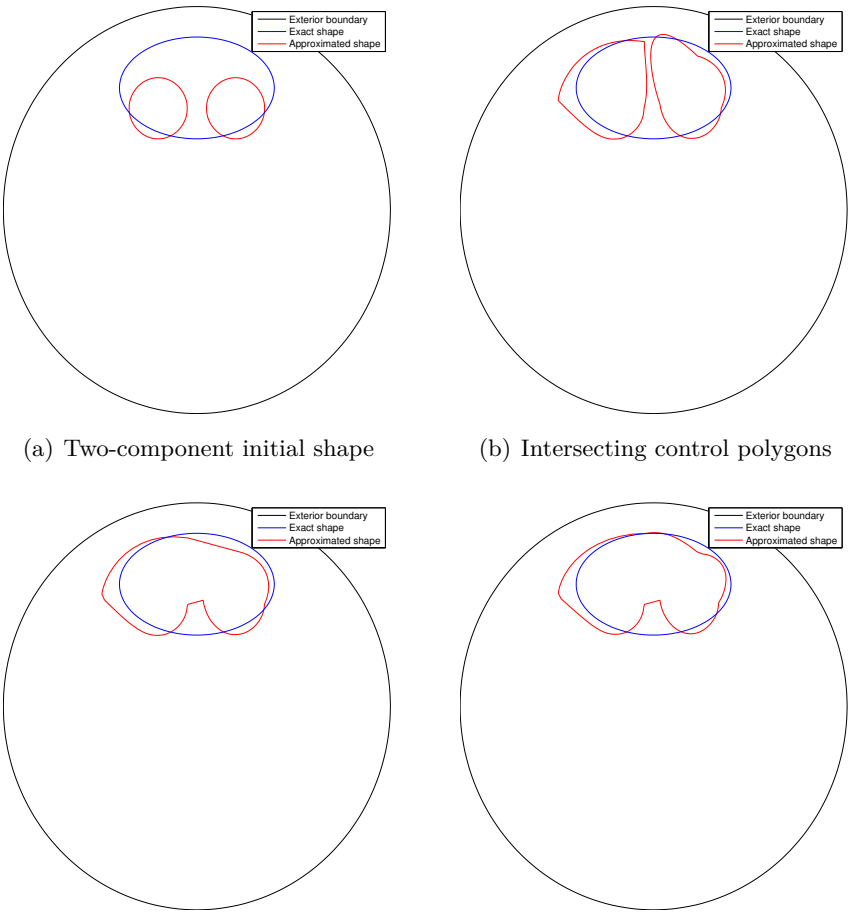


Figure B.1. The flip procedure can merge a two-component shape into a one-component shape.



(a) Two-component initial shape

(b) Intersecting control polygons

(c) Flip procedure - The two components have merged

(d) Final shape

Figure B.2. Detection of one obstacle starting from a two-component shape

References

[1] L. Afraites, M. Dambrine, K. Eppler, and D. Kateb. Detecting perfectly insulated obstacles by shape optimization techniques of order two. *Discrete Contin. Dyn. Syst. Ser. B*, 8(2):389–416 (electronic), 2007.

[2] G. Allaire, C. Dapogny, and P. Frey. Shape optimization with a level set based mesh evolution method. *Comput. Methods Appl. Mech. Engrg.*, 282:22–53, 2014.

[3] G. Allaire, F. de Gournay, F. Jouve, and A.-M. Toader. Structural optimization using topological and shape sensitivity via a level set method. *Control Cybernet.*, 34(1):59–80, 2005.

[4] H. Ammari and H. Kang. *Reconstruction of small inhomogeneities from boundary measurements*, volume 1846 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 2004.

[5] M. Badra, F. Caubet, and M. Dambrine. Detecting an obstacle immersed in a fluid by shape optimization methods. *Math. Models Methods Appl. Sci.*, 21(10):2069–2101, 2011.

[6] E. Bishop. A generalization of the Stone-Weierstrass theorem. *Pacific J. Math.*, 11(3):777–783, 1961.

[7] L. Bourgeois and J. Dardé. A quasi-reversibility approach to solve the inverse obstacle problem. *Inverse Probl. Imaging*, 4(3):351–377, 2010.

[8] M. Burger, B. Hackl, and W. Ring. Incorporating topological derivatives into level set methods. *J. Comput. Phys.*, 194(1):344–362, 2004.

[9] M. Burger and S. J. Osher. A survey on level set methods for inverse problems and optimal design. *European J. Appl. Math.*, 16(2):263–301, 2005.

[10] F. Caubet. Instability of an inverse problem for the stationary Navier-Stokes equations. *SIAM J. Control Optim.*, 51(4):2949–2975, 2013.

[11] F. Caubet, C. Conca, and M. Godoy. On the detection of several obstacles in 2d stokes flow: topological sensitivity and combination with shape derivatives. *Inverse Probl. Imaging*, 10(2):327–367, 2016.

[12] F. Caubet and M. Dambrine. Localization of small obstacles in stokes flow. *Inverse Problems*, 28(10):105007, 2012.

[13] F. Caubet, M. Dambrine, and D. Kateb. Shape optimization methods for the inverse obstacle problem with generalized impedance boundary conditions. *Inverse Problems*, 29(11):115011, 26, 2013.

[14] F. Caubet, M. Dambrine, D. Kateb, and C. Z. Timimoun. A Kohn-Vogelius formulation to detect an obstacle immersed in a fluid. *Inverse Probl. Imaging*, 7(1):123–157, 2013.

[15] A. N. Christiansen, M. Nobel-Jørgensen, N. Aage, O. Sigmund, and J. A. Bærentzen. Topology optimization using an explicit interface representation. *Structural and Multidisciplinary Optimization*, 49(3):387–399, 2014.

[16] D. Colton and R. Kress. *Inverse acoustic and electromagnetic scattering theory*, volume 93 of *Applied Mathematical Sciences*. Springer-Verlag, Berlin, second edition, 1998.

[17] J. Dardé. *Quasi-reversibility and level set methods applied to elliptic inverse problems*. Theses, Université Paris-Diderot - Paris VII, December 2010.

[18] C. Ericson. *Real-Time Collision Detection (The Morgan Kaufmann Series in Interactive 3-D Technology) (The Morgan Kaufmann Series in Interactive 3D Technology)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004.

[19] G. Farin. *Curves and Surfaces for CAGD: A Practical Guide*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 5th edition, 2002.

[20] P.J. Frey and P.L. George. *Mesh Generation: Application to Finite Elements, Second Edition*. Wiley-ISTE, 2008.

[21] S. Garreau, P. Guillaume, and M. Masmoudi. The topological asymptotic for PDE systems: the elasticity case. *SIAM J. Control Optim.*, 39(6):1756–1778 (electronic), 2001.

[22] F. Hecht. Finite Element Library Freefem++. `http://www.freefem.org/ff++/`.

[23] A. Henrot and M. Pierre. *Variation et optimisation de formes : Une analyse géométrique.*, volume 48 of *Mathématiques & Applications (Berlin) [Mathematics & Applications]*. Springer, Berlin, 2005.

[24] V. Isakov. *Inverse problems for partial differential equations*, volume 127. Springer Science & Business Media, 2006.

[25] S. Kichenassamy, A. Kumar, P. Olver, A. Tannenbaum, and A. Yezzi Jr. Gradient flows and geometric active contour models, 1994.

[26] O. Labbani-I., P. Merveilleux-O., and O. Ruatta. Free form based active contours for image segmentation and free space perception. preprint.

[27] A. Marco and J.-J. Martínez. A fast and accurate algorithm for solving bernstein–vandermonde linear systems. *Linear Algebra and its Applications*, 422(2):616 – 628, 2007.

[28] J. O'Rourke. *Computational Geometry in C*. Cambridge University Press, New York, NY, USA, 2nd edition, 1998.

[29] O. Pantz and K. Trabelsi. Simultaneous shape, topology, and homogenized properties optimization. *Struct. Multidiscip. Optim.*, 34(4):361–365, 2007.

[30] P.-O. Persson. *Mesh Generation for Implicit Geometries.* PhD thesis, Massachusetts Institute of Technology (USA), Cambridge, MA, USA, 2005. AAI0807802.

[31] T.D.M. Pjan. 3d free form method and applications to robotics. Master's thesis, University of Limoges, 2014. `http://www.unilim.fr/pages_perso/olivier.ruatta/pub/stage.pdf`.

[32] F. Santosa. A level-set approach for inverse problems involving obstacles. *ESAIM, Control Optim. Calc. Var.*, 1:17–33, 1996.

[33] A. Schumacher. *Topologieoptimisierung von Bauteilstrukturen unter Verwendung von Lopchpositionierungkrieterien.* PhD thesis, Universität-Gesamthochschule-Siegen, 1995.

[34] T. W. Sederberg. Computer aided geometric design, October 2014.

[35] J. A. Sethian. *Level set methods and fast marching methods. Evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science.* Cambridge: Cambridge University Press, 1999.

[36] J. Sokołowski and A. Żochowski. On the topological derivative in shape optimization. *SIAM J. Control Optim.*, 37(4):1251–1272 (electronic), 1999.

[37] J. Sokołowski and J.-P. Zolésio. *Introduction to shape optimization*, volume 16 of *Springer Series in Computational Mathematics.* Springer-Verlag, Berlin, 1992. Shape sensitivity analysis.