Shape Sensitivity of Constructive Representations

Jiaqin Chen* Michael Freytag† Vadim Shapiro‡

Spatial Automation Laboratory
University of Wisconsin at Madison
1513 University Avenue, Madison, WI 53706

Abstract

Most solid models are archived using boundary representations, but they are created, edited, and optimized using high level constructive methods that rely on parameterized Boolean set operations and feature-based techniques. Downstream applications often require optimization of integral-valued performance measures over such models that include volume, mass, and energy properties, as well as more general distributed fields (stress, temperature, etc.). A key computational utility in all such applications is the computation of the sensitivity of the performance measure with respect to the parameters in the solid's construction history.

We show that for a class of performance measures defined as domain integrals, the sensitivity with respect to a parameter requires integration over a subset of the solid's boundaries that is affected by that parameter. In contrast to earlier methods, the proposed approach for computing sensitivities does not require solid's boundary to remain homeomorphic, and may be used with most types of constructive representations, including CSG and feature-based representations, where the defining Boolean expression may not be known. Simplicity and effectiveness of the proposed technique are illustrated on several common shape optimization problems.

Keywords: shape sensitivity, constructive representation, implicit representation, *R*-functions

1 Introduction

1.1 Motivation

Computer analysis and simulation of digital models is now performed routinely in an effort to predict and *improve* the performance of engineering artifacts before the designs are finalized and built. The procedure for changing the form of the solid model based on the results of the analysis is known as shape optimization: finding an optimal shape to improve a certain performance measure under some constraints. The shape optimization process itself is driven by shape (design) *sensitivity* that is broadly conceived as the derivative of a *performance measure* with respect to *geometric parameters* that define the solid shape. Formally, a performance measure is quantified as a (typically) real-valued *objective function* or *cost function*, and the computation of its sensitivities provides the rigorous basis for most deterministic optimization algorithms.

In this paper we consider the shape sensitivity computation of a general class of objective functions which can be expressed as a domain integral $F(\Omega) = \int_\Omega f d\Omega$, where F is the objective function, f is the integrand function which we assume to be differentiable through the paper, and Ω is the geometric (solid) domain. Many common performance measures are represented in this form. For example, volume is the integration of a unity function over the domain, mass is the integration of the density function, moment of inertia is defined as $I = \int_\Omega \rho r^2 d\Omega$ where ρ is the density and r is the distance to the rotational axis. Other physical quantities can be written as elementary functions of integral-valued functions; for example, the center of gravity can be written as $\frac{\int_\Omega \vec{x}\rho(\vec{x})d\Omega}{\int_\Omega \rho(\vec{x})d\Omega}$, where $\vec{x}$ is the coordinate and $\rho(\vec{x})$ is the density function. An important class of performance measures involves integrals of some field functions. For example, the compliance is defined as the total strain energy of the domain, the average temperature is the integral of a temperature field divided by the volume, etc. Many engineering design problems and physically based approaches to geometric modeling/graphic tasks lead to a shape optimization problem with the objective function and/or constraint functions in this form. For example, a structural design problem may seek the optimal shape with the highest stiffness, which can be written as a compliance minimization problem subject to a weight constraint [Bendsøe and Sigmund 2003]. Volume preserving deformation problems [Rappoport et al. 1995; Hirota et al. 1999] can be formulated as a deformation energy minimization problem subject to the volume constraint.

Computing mass properties (including volume, inertia, areas, etc.) over constructive representations of solids and their boundaries was one of the first applications of solid modeling [Lee and Requicha 1982b; Lee and Requicha 1982a; Sarraga 1982]. Similar techniques may be used to compute integrals of more general functions [Freytag and Shapiro 2004]. Except in special situations, the integral functions are computed by discretizing the solid's interior and/or boundary. However, the geometric parameters of interest in shape optimization appear explicitly only as parameters in geometric *primitives* $\{\Omega_i, i = 1, \ldots, N\}$ of some *constructive representation* of the solid Ω. Such constructive representations include Constructive Solid Geometry, feature-based representations, and other procedural constructions of solids. Typical geometric parameters in constructive representations include sizes, radii, dimensions, angles, distances, control points, and so on. A constructive representation completely defines the solid Ω implicitly, and is sufficient to construct the boundary representation, $\partial\Omega$, via a suitable boundary evaluation procedure. But it should also be clear that the dependence of any objective func-

*e-mail: jiaqinchen@wisc.edu
†e-mail: freytag@cae.wisc.edu
‡e-mail: vshapiro@engr.wisc.edu

tion $F(\Omega) = \int_\Omega f d\Omega$ on geometric parameters is also implicit. Thus, if b is a geometric parameter of some primitive Ω_i, it is not clear how the shape sensitivity $\frac{dF}{db}$ may be computed.

1.2 Contributions

In this paper, we solve the shape sensitivity problem for constructive representations with a set of primitives $\{\Omega_i\}$. We will assume that the representation of a primitive Ω_i is sufficient to determine its boundary $\partial\Omega_i$ and to compute its dependence with respect to any parameter b that affects its shape. In particular, if a primitive Ω_i is represented *implicitly* by a function Φ_i, then this dependence is reflected by $\frac{\partial\Phi_i}{\partial b}$. For convenience, we will focus mostly on such implicitly defined primitives, but extensions to free-form and mesh-based primitives is relatively straightforward. Specific contributions of the paper are as follows:

- We provide a rigorous formulation of shape sensitivity that applies to any constructive representation satisfying the above assumptions. It does not assume or require knowledge of the Boolean expression or other details of the construction procedure. Consistency with the classical formulation of shape sensitivity is also shown.
- The formulation does not place artificial restrictions on the topology of the represented solid, Ω, or structure of its boundary representation, allowing to take full advantage of the versatility of constructive representations during shape optimization.
- The proposed approach to computing sensitivity leads to integration over the primitive boundaries. This suggests an efficient implementation of the method via localized algorithms, without requiring full boundary evaluation.
- Advantages and limitations of the method are explained and demonstrated by several implemented examples of shape optimization.

1.3 Previous work and state of the art

A classical result from shape (design) sensitivity analysis (SSA) shows that, for an objective function expressed as domain integration, the shape sensitivity can be expressed as a summation of two terms with one of them a domain integration and the other a boundary integration [Haug et al. 1986][Sokolowski and Zolesio 1992][Choi and Kim 2005a]. We will give an example of SSA in Section 2. The formulation relies on a differentiable mapping between the original domain and the deformed domain and is constructed in terms of the shape (design) velocity, which determines the boundary deformation as parameters change. Direct application of the SSA method to constructive representations is difficult, because it requires full boundary evaluation. No rigorous procedures for constructing the required velocity field are known. A typical assumption that the domain remains homeomorphic under the shape changes is difficult to implement computationally, and is usually translated into the requirement that the boundary representation remains isomorphic. These restrictions undermine the *raison d'être* of constructive representations.

Much of the earlier work on shape sensitivity focused on shape optimization in the context of structural or thermal analysis. In this case, a constructive representation is converted not into a boundary representation but into a finite element mesh. Application of SSA becomes even more problematic, because it requires relating the finite element nodal coordinates to the geometric parameters. For example, the mesh parametrization method [Yang et al. 1992; Chang and Choi 1992; Olhoff et al. 1992] generates the mesh from a set of master-points which are used as design variables and constructs a mapping between master-points and geometric entities. This method is difficult to use as the master-points have to be defined by users and the mesh generated may be excessively distorted. Alternately, the natural design variable method [Belegundu and Rajan 1988; Tortorelli 1993] uses fictitious analysis on the finite element model to ensure the geometric variations satisfy the shape constraints. This method requires defining the fictitious boundary conditions and the shapes generated are not compatible with the solid modeling data structures. In the context of solid modeling, most shape optimization approaches rely on an automatic mesh generator and then directly relate the geometric dimensions with finite element nodes based on variational geometry analysis [Kodiyalam et al. 1992; Botkin 1992; Chen and Tortorelli 1997]. Notably, in [Chen and Tortorelli 1997] the authors proposed an approach to computing design velocity using an iso-parametric mapping for geometric entities, assuming that the relationship among variational parameters and boundary variables can be determined.

All of the above methods are limited by their ability to associate geometric parameters with the finite element and boundary representations, and suffer from the same artificial limitations on topological form during the shape optimization process. In practice, most CAD systems still use the finite difference method to approximate the shape sensitivity. As simple as it is, the finite difference method has three intrinsic drawbacks: first, the accuracy is controlled by the step size which is difficult to choose a priori; second, it is computationally expensive; third, the finite difference method may fail and return incorrect values at points where the function is non-differentiable, without any warning to the user [Keane and Nair 2005]. Another brute-force approach to sensitivity computation is to use automatic differentiation [Griewank and Corliss 1991] of the programs for computing the performance measures. Despite of its generality, the approach is clearly impractical, because it requires differentiating all sources code (including boundary evaluation, integration, and field simulation) with respect to every (potentially hundreds or thousands) geometric parameter of interest.

1.4 Outline

The rest of the paper is organized as follows. We use a simple example in Section 2 to explain the SSA method for calculating the shape sensitivity and its limitations. In Section 3, we formulate the shape sensitivity using implicit representa-

tions as a theoretical tool and use it to derive shape sensitivity for constructive representations. We also discuss how the technical assumptions in our analysis are translated into verifiable conditions on primitives in constructive representations. In Section 4, several simple examples are used to illustrate the basic ideas and the computations implied by the proposed method, as well as its applicability to shapes with changing topology. Section 5 shows fully implemented numerical examples that demonstrate some of the capabilities of the proposed method. In conclusion (Section 6), we show that the proposed approach is consistent with the classical SSA method, and explain its application to constructive representation with free-form and variationally-defined primitives.

2 Shape sensitivity analysis (SSA) and computation

In this section, we first give a brief introduction to SSA (the details can be found in the classic books [Sokolowski and Zolesio 1992; Haug et al. 1986]). Then we use a simple example to illustrate and discuss how the sensitivity is calculated from the principle of SSA.

2.1 Shape sensitivity analysis

The basic idea in SSA is to characterize the shape deformation as a domain mapping between the original domain and the deformed domain. Under appropriate regularity assumptions, a design velocity field can be constructed from this mapping and the sensitivity analysis is carried through this velocity field.

Given a domain $\Omega \subseteq \mathbb{E}^d$ with a geometric parameter b, Ω can be regarded as a mapping from $\mathbb{R}$ to $\mathbb{E}^d$, $\Omega : b \mapsto \Omega(b)$. Denote δb a small variation of b, and $\Omega(b+\delta b)$ the deformed domain. Assume that there is a domain mapping $T : \mathbf{x}(b) \mapsto \mathbf{x}(b+\delta b)$ between $\Omega(b)$ and $\Omega(b+\delta b)$, $\Omega(b+\delta b) = T(\Omega(b); \delta b)$. With sufficient regularity assumptions and using Taylor series expansion, we have

$$\begin{aligned}\mathbf{x}(b+\delta b) &= T(\mathbf{x}(b);\delta b)\\ &= T(\mathbf{x}(b);0) + \delta b \left.\frac{\partial T}{\partial(\delta b)}\right|_{\delta b=0} + o(\delta b)\\ &= \mathbf{x}(b) + \delta b \frac{d\mathbf{x}(b)}{db} + o(\delta b).\end{aligned}$$

Define the design velocity as $\vec{v} = \frac{d\mathbf{x}}{db} = \left.\frac{\partial T}{\partial(\delta b)}\right|_{\delta b=0}$ and ignore the high-order terms in the Taylor expansion, we have $\mathbf{x}(b+\delta b) = \mathbf{x}(b) + \vec{v}\delta b$. The Jacobian, J, of this transformation is $J = \nabla T = I + \delta b \nabla(\vec{v})$. For the objective function $F(b) = \int_{\Omega(b)} f(b, \mathbf{x}(b)) d\Omega$, f is a differentiable function,

$$\begin{aligned}F(b+\delta b) &= \int_{\Omega(b+\delta b)} f(b+\delta b, \mathbf{x}(b+\delta b)) d\Omega\\ &= \int_{\Omega(b)} f(b+\delta b, \mathbf{x}(b)+\vec{v}\delta b)|J| d\Omega.\end{aligned}$$

So

$$\begin{aligned}\frac{dF}{db} &= \lim_{\delta b\to 0}\frac{F(b+\delta b)-F(b)}{\delta b} = \int_{\Omega(b)}\left[\frac{df}{db} + f\frac{d|J|}{db}\right] d\Omega\\ &= \int_{\Omega(b)}\left[\frac{\partial f}{\partial b} + \nabla f^T\cdot\vec{v} + f div(\vec{v})\right] d\Omega\\ &= \int_{\Omega(b)}\left[\frac{\partial f}{\partial b} + div(f\vec{v})\right] d\Omega \qquad (1)\end{aligned}$$

Applying the divergence theorem to Expression (1), we have

$$\frac{dF}{db} = \int_{\Omega(b)}\frac{\partial f}{\partial b} d\Omega + \int_{\partial\Omega(b)} f v_n d\Gamma, \qquad (2)$$

where v_n is the normal component of $\vec{v}$.

2.2 SSA for a triangle example

We now use a simple example to illustrate how the SSA can be used to compute shape sensitivities. We consider the volume (area) function $V = \int_\Omega d\Omega$ defined on a right triangle Ω with two legs a and b as shown in Figure 1. Assume b is the parameter subject to change, the goal is to compute the sensitivity $\frac{dV}{db}$. In this example, it is easy to see that $V = \frac{1}{2}ab$ and $\frac{dV}{db} = \frac{1}{2}a$ from basic knowledge of elementary geometry. It should be obvious that for most geometries, the analytic expression of the objective function is not known or requires tedious calculation (even for the simple volume function), and the derivative cannot be directly obtained.

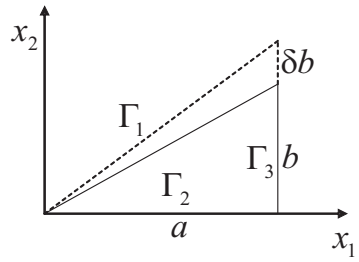


Figure 1: The triangle with one leg b as the design variable.

Substituting $f \equiv 1$ into Expression (1) and (2), we have

$$\frac{dV}{db} = \int_\Omega div\left(\frac{d\mathbf{x}}{db}\right) d\Omega = \int_{\partial\Omega}\frac{d\mathbf{x}}{db}\cdot\vec{n} d\Gamma, \qquad (3)$$

where $\vec{n}$ is the normal vector at boundary $\partial\Omega$.

We have two choices in computing the derivative: one is to use domain integration, the other is to use boundary integration:

(1) If we can construct the domain mapping T, then we can use the domain integration in Expression (1). In this simple example we can construct T by inspection as $T(x_1, x_2; \delta b) = (x_1, x_2) + \delta_b(0, \frac{x_2}{b})$. Then

$$\frac{d\mathbf{x}}{db} = \frac{\partial T}{\partial(\delta b)} = \left(0, \frac{x_2}{b}\right).$$

A direct calculation gives

$$\frac{dV}{db} = \int_\Omega div\left(\frac{d\mathbf{x}}{db}\right) d\Omega = \int_\Omega \frac{1}{b} d\Omega = \frac{1}{2}a.$$

(2) From Expression (3) we see that knowing the design velocity and the normal vector of the boundary is enough to compute the shape sensitivity. For example, we may construct the velocity field $\vec{v}$ at the boundary as:

$$\vec{v} = \begin{cases} \left(0, \frac{x_1}{a}\right), & \text{if} \quad x \in \Gamma_1 \\ (0,0), & \text{if} \quad x \in \Gamma_2 \\ \left(0, \frac{x_2}{b}\right), & \text{if} \quad x \in \Gamma_3 \end{cases}.$$

The corresponding normal vector is

$$\vec{n} = \begin{cases} \left(-\frac{b}{\sqrt{a^2+b^2}}, \frac{a}{\sqrt{a^2+b^2}}\right), & \text{if} \quad x \in \Gamma_1 \\ (0,-1), & \text{if} \quad x \in \Gamma_2 \\ (1,0), & \text{if} \quad x \in \Gamma_3 \end{cases}.$$

So, the derivative is

$$\begin{aligned} \frac{dV}{db} &= \int_{\partial\Omega} \frac{d\mathbf{x}}{db} \cdot \vec{n} d\Gamma = \int_{\Gamma_1+\Gamma_2+\Gamma_3} \vec{v} \cdot \vec{n} d\Gamma \\ &= \int_{\Gamma_1} \frac{x_1}{\sqrt{a^2+b^2}} d\Gamma = \frac{1}{2}a. \end{aligned}$$

As we see, the divergence theorem tells us that only the normal component of the design velocity at the boundary contributes to the sensitivity. Thus, we only need to know the normal velocity v_n at the boundary. A simplified calculation for the sensitivity may be obtained by directly constructing a normal velocity at the boundary if possible. In this example, if we construct v_n as

$$v_n = \begin{cases} \frac{x_1}{\sqrt{a^2+b^2}}, & \text{if} \quad x \in \Gamma_1 \\ 0, & \text{if} \quad x \in \Gamma_2 \\ 0, & \text{if} \quad x \in \Gamma_3 \end{cases},$$

then we have

$$\frac{dV}{db} = \int_{\partial\Omega} v_n d\Gamma = \int_{\Gamma_1} \frac{x_1}{\sqrt{a^2+b^2}} d\Gamma = \frac{1}{2}a.$$

Through this simple example, we see that using the SSA theory at least requires a boundary velocity field (or its normal component) that describes the boundary variations with parameter changes. If the boundary has explicit parametrization, for example parametric curves or surfaces, the velocity field may be obtainable. If we just have a set of primitives as in constructive representations, then it is generally not obvious how to compute this velocity. Besides, when the topology of the geometry changes, the boundary becomes self-intersecting, which imposes more difficulty in computing the velocity field. Further, when SSA is applied in solid modeling systems, even with the same topology it is difficult to construct the velocity when the boundary representation changes. This is why the homeomorphism is required in SSA and in addition, the isomorphic mapping for boundary representation is required for SSA to be used in solid modeling systems.

3 Proposed approach

The difficulty in SSA lies in the construction of the velocity field, especially when the topology changes. We now show that a much simpler sensitivity computing method can be used for constructive representations without tracking boundary movements. Our method can easily handle topological changes and it does not need to know or maintain the boundary representation.

We first formulate the shape sensitivity for any domain through the tool of implicit representations and discuss the validity of the sensitivity formulation, including the differentiability of the objective function and the implicit function. Then we discuss a sufficient condition under which the sensitivity analysis is valid for constructive representations, and show that the sensitivity can be computed efficiently as an integration over active boundaries (boundaries affected by the parameter).

We choose the implicit representation of the geometry to derive the sensitivity analysis because of its representational advantages in avoiding tracking each individual boundary piece. However, we emphasize that the implicit representation is only a convenient theoretical tool we use for formulating the sensitivity analysis. We do not need to know or construct such an implicit representation and the resulting sensitivity should not by any means depend on any particular representation.

3.1 Sensitivity formulation through implicit representation

Given a geometric domain Ω with boundary $\partial\Omega$, suppose there exists a differentiable implicit function Φ with non-zero gradient such that $\Omega = \{x|\Phi(x) \geq 0\}$ and $\partial\Omega = \{x|\Phi(x) = 0\}$. Assume that b is a parameter of Ω, then Φ is also a function of b. Let D be a fixed domain that contains Ω and its variations, if we use the following characteristic function

$$H(\Phi(x)) = \begin{cases} 1, & \text{if} \quad \Phi(x) \geq 0 \\ 0, & \text{if} \quad \Phi(x) < 0 \end{cases}$$

as an indicator of whether a given point x belongs to Ω or not, we have

$$\Omega = \{x \,|\, x \in D, \Phi(x) \geq 0\} = \{x \,|\, x \in D, H(\Phi) = 1\}.$$

Employing the characteristic function, the objective function F defined on Ω can be transformed to a function defined on the reference domain:

$$F = \int_\Omega f d\Omega = \int_D f H(\Phi) d\Omega. \tag{4}$$

Using the chain rule and the fact that $\frac{dH(\Phi)}{d\Phi} = \frac{\delta(\Phi)}{|\nabla\Phi|}$, where $\delta(\cdot)$ is the Dirac Delta function, the derivative of F with re-

spect to the parameter b is

$$\begin{aligned}\frac{dF}{db} &= \int_D \left[\frac{df}{db}H(\Phi) + f\frac{dH(\Phi)}{db}\right] d\Omega \\ &= \int_D \left[\frac{\partial f}{\partial b}H(\Phi) + f\frac{\delta(\Phi)}{|\nabla\Phi|}\frac{d\Phi}{db}\right] d\Omega \\ &= \int_\Omega \frac{\partial f}{\partial b} d\Omega + \int_{\partial\Omega} \frac{f}{|\nabla\Phi|}\frac{\partial\Phi}{\partial b} d\Gamma. \qquad (5)\end{aligned}$$

Expression (5) gives the derivative of F with respect to b *if* it exists. We had assumed the following two conditions for Expression (5) to be valid:

(1) The differentiability of F

As we know, for a function to be differentiable at a point the derivative of the function is continuous at that point. While it is generally difficult to characterize the differentiability of an objective function in geometric design problems, we have the following sufficient condition for determining the differentiability of F:

Definition 3.1. *A point set $X : c \mapsto X(c)$ is $\mathscr{G}$-continuous ($\mathscr{G}$ stands for geometric) at c_0 if $\int_{X\Delta X_0} dX \to 0$ as $c \to c_0$, where $X_0 = X(c_0)$ and $X\Delta X_0 \equiv (X - X_0) \cup (X_0 - X)$ is the symmetric difference of X and X_0.*

Theorem 3.2. *F defined in Expression (4) is differentiable at b_0 if Ω and $\partial\Omega$ are $\mathscr{G}$-continuous at b_0.*

Proof: With the assumption that f and Φ are differentiable at b_0, and the gradient of Φ is non-zero, the integrands in Expression (5) are continuous. Then we can conclude from Lemma 3.3 that F is differentiable.

Lemma 3.3. *Given a point set $X : c \mapsto X(c)$, if X is $\mathscr{G}$-continuous at c_0, then the function $G = \int_X g dX$ is continuous at c_0 for any continuous function g (may also be a function of c) defined on X.*

Proof: This can be easily verified by using the triangle inequality

$$\begin{aligned}|G - G_0| &= \left|\int_X g dX - \int_{X_0} g_0 dX\right| \\ \leq\ & \left|\int_X g dX - \int_{X_0} g dX\right| + \left|\int_{X_0} g dX - \int_{X_0} g_0 dX\right| \\ =\ & \left|\int_{X-X_0} g dX - \int_{X_0-X} g dX\right| + \left|\int_{X_0} (g - g_0) dX\right| \\ \leq\ & \int_{X-X_0} |g|\, dX + \int_{X_0-X} |g|\, dX + \int_{X_0} |g - g_0|\, dX \\ =\ & \int_{X\Delta X_0} |g|\, dX + \int_{X_0} |g - g_0|\, dX \\ \leq\ & \max|g| \int_{X\Delta X_0} dX + \int_{X_0} |g - g_0|\, dX.\end{aligned}$$

□

While the $\mathscr{G}$-continuity of Ω is satisfied for most geometric deformation problems, we often see the boundary $\partial\Omega$ not being $\mathscr{G}$-continuous in many situations. A small perturbation of a parameter may cause sudden changes of the boundary (as in the example shown in Figure 4). In these situations, the differentiability of F is indeterminable. Nevertheless, even if F is not differentiable, Expression (5) can still be used to compute the one-side derivative of F if one wishes.

(2) The differentiability of Φ

As we assumed the existence of a differentiable function Φ with non-zero gradient for Ω in deriving Expression (5), it may be difficult or impossible to construct the analytic expression of this function. More importantly, we *do not* want to construct such functions even if they exist. Notice that the properties of Φ do not matter at the interior points of Ω after we use the characteristic function $H(\Phi)$, so the existence of a function Φ that is differentiable with non-zero gradient *at the boundary* is enough to validate the sensitivity formulation. In fact, the second term in Expression (5) is a boundary integration and we know that the set of measure zero at the boundary does not affect the boundary integration, this property of Φ can be relaxed to the differentiability of Φ with non-zero gradient *almost everywhere at the boundary*. Therefore, to validate the sensitivity analysis in Expression (5), the differentiability of Φ everywhere with non-zero gradient in Ω can be relaxed to the differentiability of Φ with non-zero gradient *almost everywhere at the boundary* $\partial\Omega$.

3.2 Sensitivity for constructive representations

Let $\{\Phi_i, i = 1, \ldots, N\}$ be the implicit functions defining primitives $\{\Omega_i, i = 1, \ldots, N\}$ respectively, Ω_i is the point set which satisfies $\Omega_i = \{x | \Phi_i(x) \geq 0\}$. For the geometry Ω constructed from $\{\Omega_i\}$ by Boolean set operations, a single implicit function Φ for Ω can be constructed by many methods, for example, logical operations on implicit functions $\{\Phi_i\}$.

Nevertheless, the constructed implicit function Φ has to satisfy the differentiability requirement. One way to construct such a single analytic differentiable function $\Phi = \Phi(\Phi_1, \ldots \Phi_N)$ is to use the theory of R-functions. As is well known, the differentiability of implicit functions constructed using R-functions depends on the particular Boolean set expression used to combine the primitives. In many situations, we may not know or even care about this set expression. As in computing the shape sensitivity, we are only interested in the overall geometry variations corresponding to the parameters. Assume that there exists a Boolean set expression in which each primitive only appears once, then the theory of R-functions tells us that the constructed Φ for this set expression can be guaranteed to be m-differentiable and preserve the normal of each primitive at the boundary except at those points where more than one primitive is zero. The construction using R-functions which is sufficiently smooth and preserves normalization on the boundary has been well studied in the literature [Rvachev 1967; Rvachev 1982; Shapiro 1988; Shapiro and Tsukanov 1999].

If we assume that every point on $\partial\Omega$ belongs to the boundary of exactly one primitive except at a set of measure zero,

then the existence of a differentiable implicit function almost everywhere at the boundary is guaranteed by the theory of R-functions. (Hence we exclude set complement operations because a primitive and its complement have the same boundary.) Although Expression (5) is the sensitivity formulation in terms of Φ, we can further show that the sensitivity can be decoupled to sensitivities of each primitive for constructive representations. If a point x_0 on $\partial\Omega$ only belongs to the boundary of one primitive Ω_k, a basic result from the theory of R-functions states that the derivative of the function Φ at x_0 is determined by the derivative of that primitive function, i.e. $\frac{\partial \Phi}{\partial b}\Big|_{x_0} = \frac{\partial \Phi_k}{\partial b}\Big|_{x_0}$. Therefore, denoting $\partial\Omega_k$ as the portion of boundary $\partial\Omega$ corresponding to primitive Ω_k and assuming that the construction of Φ preserves the gradient of each primitive at the boundary, i.e. $\nabla\Phi|_{\partial\Omega_k} = \nabla\Phi_k|_{\partial\Omega_k}$, the second term in Expression (5) can be written as

$$\int_{\partial\Omega} \frac{f}{|\nabla\Phi|}\frac{\partial\Phi}{\partial b}d\Gamma = \sum_{k=1}^{N}\int_{\partial\Omega_k} \frac{f}{|\nabla\Phi_k|}\frac{\partial\Phi_k}{\partial b}d\Gamma. \tag{6}$$

We define the *active primitive* for parameter b to be the primitive depending on b, i.e. Φ_k is a function of b. Each parameter may have more that one active primitive. Denote $\mathscr{A}(b) \subseteq \{1,\ldots,N\}$ the index set of active primitives for parameter b, Expression (6) can be further simplified as

$$\int_{\partial\Omega} \frac{f}{|\nabla\Phi|}\frac{\partial\Phi}{\partial b}d\Gamma = \sum_{k\in\mathscr{A}(b)}\int_{\partial\Omega_k} \frac{f}{|\nabla\Phi_k|}\frac{\partial\Phi_k}{\partial b}d\Gamma. \tag{7}$$

Expression (7) tells us that for each parameter only its active primitives contribute to the boundary integral. Therefore we can ignore the rest of the primitives. Substituting this result into Expression (5), we have

$$\frac{dF}{db} = \int_{\Omega}\frac{\partial f}{\partial b}d\Omega + \sum_{k\in\mathscr{A}(b)}\int_{\partial\Omega_k} \frac{f}{|\nabla\Phi_k|}\frac{\partial\Phi_k}{\partial b}d\Gamma. \tag{8}$$

We refer to the *active boundary* for parameter b as the boundary point set which belong to some active primitive, i.e. $\bigcup_{k\in\mathscr{A}(b)} \partial\Omega_k$. From Expression (8), we see that for constructive representations, we can further localize the previous conditions and assumptions to those on the active primitives as follows:

- The differentiability of F: the $\mathscr{G}$-continuity condition for $\partial\Omega$ in Theorem 3.2 for F to be differentiable reduces to the $\mathscr{G}$-continuity of the active boundaries.
- The differentiability of Φ reduces to the differentiability on the portion of the boundary of active primitives except at a set of measure zero. Therefore, we only need the assumption that (1) there exists a Boolean expression where all active primitives appear once, and (2) every boundary point belonging to some active primitive's boundary does not belong to any other primitive's boundary except at a set of measure zero.

In summary, the formulation in Expression (8) only relies on the assumption that *any active boundary point only belongs to one primitive except a set of measure zero*. This assumption can be easily checked by computing the arrangement of primitives at the active boundaries. For example, a sufficient (but not necessary) condition is that primitives intersect transversely. As long as this assumption is satisfied during the shape deformation, the shape sensitivity can be computed by Expression (8). This formulation places *no* restriction on the topological properties of the domain and is general enough for handling geometric deformation problems with topological changes.

The computational advantage of Expression (7) should be obvious. We only need to know the implicit representation Φ_k and its corresponding portion of the boundary $\partial\Omega_k$ for each active primitive Ω_k of parameter b. We do not need to know or compute the implicit function Φ for the whole geometry Ω. We do not need to know the full boundary information. Computation of Expression (7) does not require the iso-morphism of the boundary representation and we certainly do not need to maintain it. For constructive representations, each parameter typically has only a few active primitives, which makes the computation very efficient. The integration in Expression (7) over boundary pieces can be implemented through any suitable numerical integration techniques (combined with a point membership test).

Finally consider the computation of the first term in Expression (8). If f is an explicit function of b, then the computation is a straightforward domain integration. In some cases, the first term may be difficult to calculate if $\frac{\partial f}{\partial b}$ is not directly obtainable. For example, function f may be a field function which depends on the solution of some boundary value problem defined on Ω, such as temperature (in a heat transfer problem), displacement (in an elasticity problem), etc. There are various techniques for computing such derivatives $\frac{\partial f}{\partial b}$, for example, adjoint method [Haug et al. 1986]. The discussion of such issues is beyond the scope of this paper. Here we only point out that for a wide class of problems, the first term can also be written as a boundary integral over active boundaries. For example, in [Chen et al. 2006] the sensitivity analysis for a minimum compliance optimization problem is performed and is shown to be a boundary integral.

4 Illustrative examples

Before introducing the examples, let us first restate our main results for constructive representations from the last section:

- A sufficient condition for the differentiability of the objective function F:

  If there exists a Φ which is differentiable with non-zero gradient almost everywhere at the active boundary, then F is differentiable if Ω and the active boundaries are $\mathscr{G}$-continuous.

- A sufficient condition for the existence of such a Φ:

  If there exists a Boolean expression where all active primitives appear only once, and the active boundary

points only belong to one primitive except a set of measure zero, then we can construct a Φ such that it is differentiable with non-zero gradient at the active boundary.

- Therefore, the sufficient conditions for validating the sensitivity formulation in Expression (8) include the following three components:

  Condition 1: Ω and the active boundaries are $\mathscr{G}$-continuous;

  Condition 2: There exists a Boolean expression where all active primitives appear only once;

  Condition 3: The active boundary points only belong to one primitive except a set of measure zero.

As we discussed in the last section, these conditions are reasonable for most practical applications and can be easily verified computationally, we now use several examples to illustrate the computational procedure for the proposed approach. The examples are chosen to illustrate the assumptions we impose on the geometry and to demonstrate the advantages of the proposed method. As the $\mathscr{G}$-continuity for Ω and the existence of a Boolean expression are generally satisfied for most engineering shapes (and are easy to check in the following examples), we focus on discussing the $\mathscr{G}$-continuity of the active boundaries and the measure zero condition. For simplicity, in all examples we use the volume function $V = \int_\Omega d\Omega$ as the objective function.

4.1 Examples with valid sensitivity analysis

As we said earlier, there are no artificial restrictions on the topological changes of the geometry or its boundary in our formulation. Once the sufficient conditions are satisfied, the proposed approach can be applied to compute the sensitivity analysis.

Example 1: Shape deformation with no topological changes (Figure1)

We use the triangle example in Section 2 to illustrate how the sensitivity can be calculated using our approach. The triangle can be constructed from the three half-spaces which are defined as:

$$\begin{aligned} \Phi_1 &= bx_1 - ax_2, \\ \Phi_2 &= x_2, \\ \Phi_3 &= a - x_1. \end{aligned}$$

The active primitive for parameter b is Φ_1 and its corresponding boundary is Γ_1. From Expression (8), for the volume function, $V = \int_\Omega d\Omega$, we have,

$$\frac{dV}{db} = \int_{\Gamma_1} \frac{1}{|\nabla\Phi_1|} \frac{d\Phi_1}{db} d\Gamma.$$

Since

$$\begin{aligned} \frac{d\Phi_1}{db} &= x_1, \\ |\nabla\Phi_1| &= \sqrt{a^2+b^2}, \end{aligned}$$

we have

$$\frac{dV}{db} = \int_{\Gamma_1} \frac{x_1}{\sqrt{a^2+b^2}} d\Gamma = \frac{1}{2}a.$$

Compared with the SSA method, the simplicity of our approach is obvious since there is no need to know the full boundary information once we know the active primitives.

Example 2: Shape deformation with topological changes (Figure 2)

When topological changes happen, the deformation of primitive shapes generates self-intersected boundaries. As in our earlier statement, if the active boundary is $\mathscr{G}$-continuous and the active boundary points which belong to more than one primitive is of measure zero, the sensitivity derived in Section 3.1 is still valid.

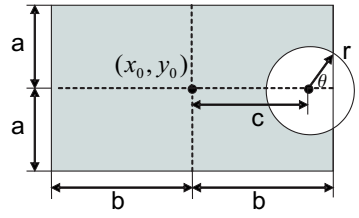


Figure 2: A rectangle with a circular hole.

Let us consider a simple geometry composed of a rectangle and a circular hole shown in Figure 2. The x-position c of the hole is chosen as the design parameter. When the hole moves in/out of the rectangle, obviously the topology of the geometry changes. But regardless of the topology, the intersection of the hole's boundary and the rectangle consists of zero, one or two points, which satisfies our measure zero condition.

A direct calculation gives

$$V = 4ab - \pi r^2 + r^2 \arccos\frac{b-c}{r} - (b-c)\sqrt{r^2-(b-c)^2}$$

and

$$\frac{dV}{dc} = 2\sqrt{r^2-(b-c)^2}.$$

We now try the proposed approach. This geometry can be defined from the following three primitives:

$$\begin{aligned} \Phi_a &= a^2 - (y-y_0)^2, \\ \Phi_b &= b^2 - (x-x_0)^2, \\ \Phi_r &= (x-x_0-c)^2 + (y-y_0)^2 - r^2. \end{aligned}$$

The active primitive of parameter c is the circular hole and its corresponding boundary is (part) of the circle. Since

$$\begin{aligned} \frac{d\Phi_r}{dc} &= -2(x-x_0-c), \\ |\nabla\Phi_r| &= 2\sqrt{(x-x_0-c)^2+(y-y_0)^2}, \end{aligned}$$

we have

$$\begin{aligned} \frac{dV}{dc} &= \int_{\partial\Omega_r} \frac{1}{|\nabla\Phi_r|}\frac{d\Phi_r}{dc}d\Gamma \\ &= \int_{\partial\Omega_r} -\frac{x-x_0-c}{\sqrt{(x-x_0-c)^2+(y-y_0)^2}}d\Gamma \\ &= \int_{\theta}^{-\theta} -r\cos\alpha d\alpha = 2r\sin\theta = 2\sqrt{r^2-(b-c)^2} \end{aligned} \quad (9)$$

At the critical points ($c = b \pm r$) where the topology changes, we have $\theta = 0$, thus the integration in Expression (9) gives zero. Figure 3 plots the relationship between the volume function and the parameter c. We can see that at the critical points, even though the topological changes do affect the behavior of the objective function, the objective function is still differentiable at those points. This is why the sensitivity analysis is still valid at critical points. With the proposed approach for sensitivity computation we do not have to identify topological events or know the behavior of the objective function. Nor do we need to maintain the boundary representation. As long as the sufficient condition is satisfied, the sensitivity computation can be carried out.

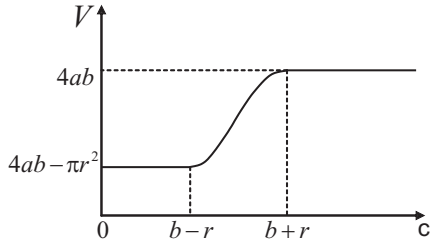


Figure 3: The plot of the volume function with respect to variable c in Figure 2.

4.2 When sensitivity analysis is invalid

Here we show examples for which the proposed approach does not apply and we explain which condition they do not satisfy.

Example 3: Condition 1 fails (Figure 4)

In Figure 4, the geometry consists of two rectangle primitives. As the length b of the left rectangle increases, it eventually hits the right rectangle ($b = b_0$). At this point, though the measure zero condition is satisfied, the $\mathscr{G}$-continuity is not satisfied. If we take a look at the function behavior at this transition point (Figure 5), it shows that the volume function is not differentiable at this point. In this case, the proposed approach can be used to compute a one-side derivative at b_0 (in this example, it is the the right derivative $\frac{dV}{db}\Big|_{b_0^+}$).

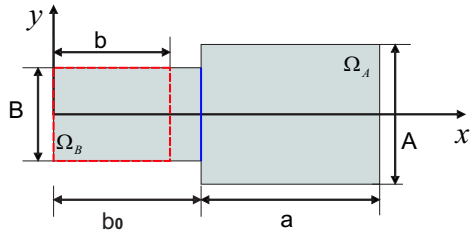


Figure 4: The union of two rectangles.

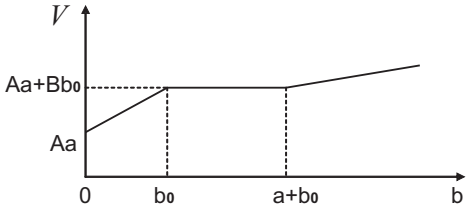


Figure 5: The volume function corresponding to the parameter b in Figure 4.

Example 4: Condition 3 fails (Figure 6)

Figure 6 shows another example with the union of two rectangles Ω_A and Ω_B. As the position b of Ω_B increases, its right edge will overlap with part of the right edge of Ω_A ($b = b_0$). At this point, though the $\mathscr{G}$-continuity is satisfied for the boundary, the measure zero condition is not satisfied since every point on the right edge of Ω_B also belongs to the boundary of Ω_A. One can easily plot the volume function to see that it is not differentiable at this point.

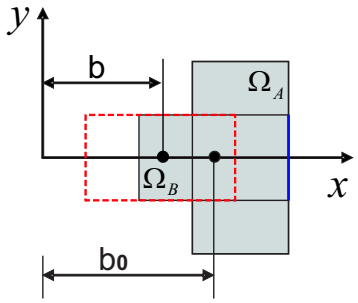


Figure 6: The union of two rectangles.

Example 5: Condition 3 fails but F is differentiable (Figure 7)

The sufficient condition we stated is not a necessary condition for valid sensitivity analysis. It is easy to construct an example that does not satisfy the sufficient condition but the objective function is still differentiable, as in Figure 7. The domain Ω is the union of two rectangles Ω_A and Ω_B. It is easy to see that the volume function is constant and, therefore, differentiable as long as b is within a range that ensures the left edge of Ω_B is contained in the left edge of Ω_A. But as in Example 4, the measure zero condition is not satisfied.

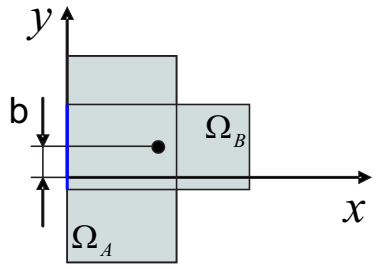


Figure 7: The union of two rectangles.

5 Numerical Examples

5.1 Optimization of a hard disk arm

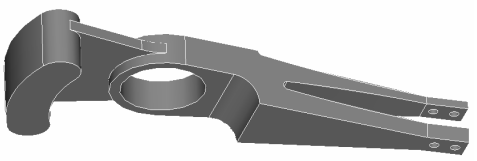
Figure 8 shows the parametrization of a hard disk arm that we would like to optimize by minimizing the volume while maintaining a fixed value for the mass moment of inertia, I_z. To do this, we intersect the arm with the complement of a cylinder aligned parallel to the z-axis and optimize the x-position a and radius r of the cylinder, see Figure 8(b). The active primitive for a and r is the cylinder and active boundaries are the part of the side face of the cylinder belonging to the arm (a and r have the same active primitive and active boundaries in this example). It is easy to check that this problem satisfies our sufficient condition for valid sensitivity analysis ($\mathscr{G}$-continuity of active boundaries and measure zero condition) and we can apply the proposed method to compute the sensitivities. As gradient-based optimization algorithms require the derivatives of both objective function and constraint functions, we compute the sensitivities of the objective function V and the constraint function I_z with respect to the design parameters a and r.

For the cylinder primitive, we have the following implicit representation:

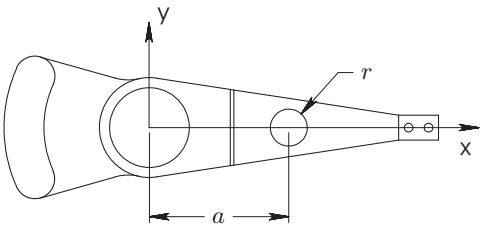
$$\Phi_R = (x-a)^2+y^2-r^2,$$

whose derivatives, with respect to the parameters a and r are:

$$\frac{d\Phi_R}{da} = -2(x-a), \frac{d\Phi_R}{dr} = -2r,$$



(a) The hard disk arm.



(b) The hole parameters.

Figure 8: The 3D view, coordinate system, and hole parameters of the arm.

and its gradient is

$$|\nabla\Phi_R| = 2\sqrt{(x-a)^2+y^2}.$$

The volume and mass moment of inertia are found through the following integrals:

$$V = \iiint_\Omega 1 d\Omega,$$
$$I_z = \iiint_\Omega (x^2+y^2) d\Omega,$$

which are computed directly by the geometric engine.

The sensitivities of V and I_z with respect to a and r are:

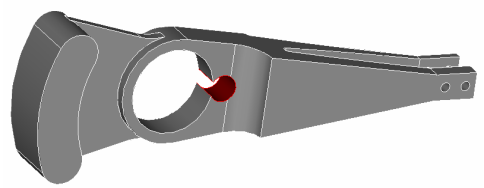
$$\frac{dV}{da} = -\iint_{\partial\Omega_R} \frac{x-a}{\sqrt{(x-a)^2+y^2}} dS,$$
$$\frac{dV}{dr} = -\iint_{\partial\Omega_R} \frac{r}{\sqrt{(x-a)^2+y^2}} dS,$$
$$\frac{dI_z}{da} = -\iint_{\partial\Omega_R} \frac{x-a}{\sqrt{(x-a)^2+y^2}} (x^2+y^2) dS,$$
$$\frac{dI_z}{dr} = -\iint_{\partial\Omega_R} \frac{r}{\sqrt{(x-a)^2+y^2}} (x^2+y^2) dS,$$

$\partial\Omega_R$ is the active boundary for a and r, i.e. the boundary point set belonging to the side face of the cylinder, which is shown in red in Figure 9.

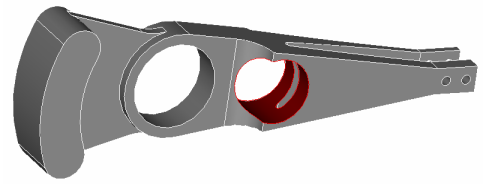
For experimental purposes, primitives are represented using boundary representations in the Parasolid modeling kernel. The optimization program operates on two primitives: the arm primitive, Figure 8(a); and our active primitive: the parameterized cylinder. Intersecting the complement of the cylinder with the arm creates one or more new faces corresponding to the active boundaries of the cylinder primitive. We integrate

over these Parasolid faces using a standard quadrature rule. Though the implementation described here computes the resulting boundary representation, in principle, boundary evaluation is not necessary. The integration may be carried out over portions of the active boundaries that classify *on* with respect to a constructive representation of the arm, provided such membership classification is supported [Sarraga 1982].

The optimization is carried out within Matlab using an interface to communicate with the Parasolid engine. Besides the moment of inertia constraint, we impose additional linear constraints on a and r such that the hole is fully contained in the arm to prevent disconnected designs. Matlab's non-linear constrained optimization function `fmincon()` uses the sensitivity information from the above integrals to converge to a final solution shown in Figure 9 within seconds in 7 iterations. At this position, the optimum amount of material is removed, while maintaining a constant mass moment of inertia about the z-axis. We can see that the optimal geometry has a very different topology with the initial geometry.



(a) The initial geometry.



(b) The optimal geometry.

Figure 9: The initial and optimal geometry of a hard disk arm.

5.2 Minimum compliance design problems

We consider a more complicated example where the objective function depends on a field function. Figure 10 shows a short cantilever beam design problem. The objective is to minimize the compliance, which is defined as the domain integral of the strain energy. The strain energy depends on the solution of a linear elasticity problem. The design parameters are the position of the circular hole and the rectangular slot. The problem formulation and sensitivity analysis can be found in [Chen et al. 2006]. It has been shown that the shape sensitivity of the compliance can be written as an integral over active boundaries, which in this case are the partial circle and partial rectangle. Figure 11 shows the initial design and the optimal shape. The topological changes, such as intersection of hole and the slot, are handled easily without any additional effort. A meshfree approach with distance fields is used to solve the structural analysis problem, whose solution is used to compute the shape sensitivity. The main optimization algorithm and numerical implementation details can be found in [Chen et al. 2006].

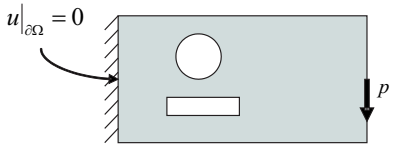
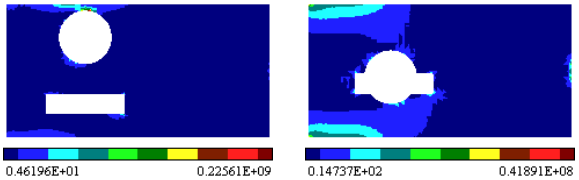


Figure 10: A minimum compliance optimization problem.

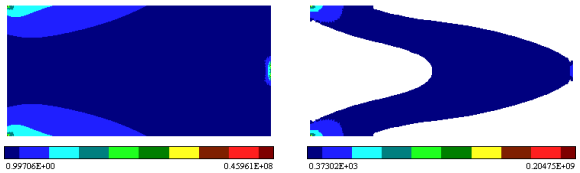


(a) The initial design. (b) The optimal shape.

Figure 11: Strain energy distribution of the initial and optimal shapes of a rectangle with a circular and rectangular slot. The design variables are the positions of the hole and the slot.

Figure 12 shows another short cantilever beam design problem. The difference with the previous example is in the parametrization of the geometry. In the previous example, the design variables are the geometric dimensions. In this example the geometry is represented by an implicit function approximated by a linear combination of B-spline basis functions: $\Phi = \sum c_i \chi_i(x)$, where $\{c_i\}$ are the coefficients and $\{\chi_i(x)\}$ are linear B-splines. Therefore, we have a freeform shape design problem. As $\{c_i\}$ are the design variables, the shape sensitivity with respect to each c_i can be shown to be a boundary integral over its active boundaries: the boundary of the shape contained within the support of the corresponding B-spline χ_i. The details of the sensitivity analysis and numerical implementation can be found in [Chen et al. 2006]. This free-form surface representation for the geometry is similar to the concept used in the level set method developed over the past several years [Sethian and Wiegmann 2000; Allaire et al. 2004; Wang et al. 2003]. However, the level-set method relies on the boundary variation for shape sensitivity analysis, and therefore does not allow topological changes. When the topology changes, the shape sensitivity is undefined and in practice one often chooses to ignore this fact and accept shape changes. The proposed shape sensitivity analysis gives a formal justification of this action in terms of the rigorous conditions listed in Section 4. Figure 13 shows the initial and optimal shapes. The material properties and the boundary conditions are the same as in Figure 10. This example confirms our earlier statement that the sensitivity formulation in our method is very general and can be applied to any geometries represented implicitly by some function Φ.

Figure 12: A minimum compliance optimization problem.



(a) The initial design (b) The optimal shape.

Figure 13: Strain energy distribution of the initial and optimal shapes. The design variables are the B-spline coefficients.

6 Conclusions

We proposed a new approach for shape sensitivity computation for a general class of functions defined as domain integrals on constructive representations of solids. Using existence of a sufficiently smooth implicit representation as a theoretical tool, we showed that the shape sensitivity can be formulated as a sum of a domain integral and a boundary integral. We further showed that the boundary integration can be computed efficiently over the boundaries of the active primitives used within the constructive representation of a solid. In contrast to other methods, our method does not place artificial topological constraints on the boundary representation of the solid and in many cases does not require full boundary evaluation. The advantages of the approach are illustrated by the simple examples in section 4 and are demonstrated by several numerical experiments in shape optimization for different objective functions. Our implementation is sufficient for demonstration purposes, but is relatively unsophisticated. A more general and efficient code would rely on automatic differentiation of arbitrary primitives [Shapiro and Tsukanov 1999; Tsukanov and Hall 2003] and would avoid full boundary evaluation whenever possible.

The proposed shape sensitivity formulation is both simpler and more powerful than earlier approaches. Yet, it is fully consistent with the classical SSA theory in the restricted situations implemented by others. Compare Expression (5) with Expression (2). In Expression (5), since $\Phi(x)|_{\partial\Omega} = 0$, we have $\frac{d\Phi}{db}\Big|_{\partial\Omega} = \frac{\partial\Phi}{\partial b}\Big|_{\partial\Omega} + \nabla\Phi \cdot \frac{dx}{db}\Big|_{\partial\Omega} = 0$, implying that $\frac{\partial\Phi}{\partial b} = -\nabla\Phi \cdot \frac{dx}{db}$ for every $x \in \partial\Omega$. By definition, $\frac{dx}{db} = \vec{v}$ and $\vec{n} = -\frac{\nabla\Phi}{|\nabla\Phi|}$. Therefore, we conclude that $\frac{\partial\Phi}{\partial b} = v_n|\nabla\Phi|$, and that the expression (5) is equivalent to (2).

The above analysis also indicates wide applicability of the proposed approach to constructive representations with most types of primitives. We only relied on *existence* of implicit representations, but in fact it does not matter whether the primitive is represented implicitly, parametrically, variationally, or procedurally. The proposed approach applies as long as we are able to compute the shape (design) velocity v_n in the direction normal to the primitive's boundary. For instance, when the primitive is a free-form (B-spline, Bezier, etc.) surface patch that is trimmed by the constructive representation, the normal velocity may be easily constructed as in [Choi and Kim 2005b] under the assumption that the topology of the patch remains invariant. Another important class of primitives can be broadly classified as extrusions of variationally constrained sketches [Hoffmann and Juan 1993]. The design velocities for such primitives are intimately connected to constraint solving techniques (for example, see [Hoffmann and Joan-Arinyo 2005; Brüderlin and Roller 1998] and references therein). If one does not have access to the relevant source code, the primitive design velocity may be estimated using finite difference techniques [Chen and Tortorelli 1997]. Once the primitive velocities are computed based on the properties and representations of the individual primitives, they can all be used simultaneously within the framework described in this paper.

Acknowledgements

This works was supported in part by the National Science Foundation grants CMMI-0323514, CMMI-0500380, CMMI-0621116, OCI-0537370, and Wisconsin Industrial & Economic Development Research Program (I&EDR).

References

ALLAIRE, G., JOUVE, F., AND TOADER, A. M. 2004. Structural optimization using sensitivity analysis and a level-set method. *Journal of Computational Physics 194*, 363–393.

BELEGUNDU, A. D., AND RAJAN, S. D. 1988. A shape optimization approach based on natural design variables and shape functions. *Computer Methods in Applied Mechanics 66*, 1, 87–106.

BENDSØE, M. P., AND SIGMUND, O. 2003. *Topology Optimization: Theory, Methods and Applications*. Springer Verlag, Berlin Heidelberg.

BOTKIN, M. E. 1992. Three-dimensional shape optimization using fully automatic mesh generation. *AIAA Journal 30*, 7, 1932–1934.

BRÜDERLIN, B., AND ROLLER, D. 1998. *Geometric Constraint Solving and Applications*. Springer.

CHANG, K. H., AND CHOI, K. K. 1992. A geometry-based parameterization method for shape design of elastic solids. *Mechanics of Structures and Machines 20*, 22, 215–252.

CHEN, S., AND TORTORELLI, D. A. 1997. Three-dimensional shape optimization with variational geometry. *Structural Optimization 13*, 81–94.

CHEN, J., SHAPIRO, V., SURESH, K., AND TSUKANOV, I. 2006. Shape optimization with topological changes and parametric control. *International Journal of Numerical Methods in Engineering*. in press.

CHOI, K. K., AND KIM, N. H. 2005. *Structural Sensitivity Analysis and Optimization I: Linear Systems*. Springer.

CHOI, K. K., AND KIM, N. H. 2005. *Structural Sensitivity Analysis and Optimization II: Nonlinear Systems and Applications*. Springer.

FREYTAG, M., AND SHAPIRO, V. 2004. B-rep SE: simplicially enhanced boundary representation. In *ACM Symposium on Solid Modeling and Applications*, G. Elber, N. Patrikalakis, and P. Brunet, Eds., 157–168.

GRIEWANK, A., AND CORLISS, G. 1991. *Automatic Differentiation of Algorithms: Theory, Implementation, and Application*. Society for Industrial and Applied Mathematics.

HAUG, E. J., CHOI, K. K., AND KOMKOV, V. 1986. *Design Sensitivity Analysis of Structural Systems*. Academic Press, New York, NY.

HIROTA, G., MAHESHWARI, R., AND LIN, M. C. 1999. Fast volume-preserving free form deformation using multi-level optimization. In *Proceedings of the Fifth ACM Symposium on Solid and Physical Modeling*, ACM Press, 234–245.

HOFFMANN, C., AND JOAN-ARINYO, R. 2005. A brief on constraint solving. *Computer-Aided Design and Applications 2*, 5, 655–663.

HOFFMANN, C. M., AND JUAN, R. 1993. Erep, an editable, high-level representation for geometric design and analysis. In *Geometric and Product Modeling*, P. Wilson, M. Wozny, and M. Pratt, Eds. North Holland.

KEANE, A., AND NAIR, P. 2005. *Computational Approaches for Aerospace Design: The Pursuit of Excellence*. John-Wiley and Sons.

KODIYALAM, S., KUMAR, V., AND FINNIGAN, P. 1992. Constructive solid geometry approach to three-dimensional structural shape optimization. *AIAA Journal 30*, 5, 1408–1415.

LEE, Y. T., AND REQUICHA, A. A. G. 1982. Algorithms for computing the volume and other integral properties of solids. i. known methods and open issues. *Commun. ACM 25*, 9, 635–641.

LEE, Y. T., AND REQUICHA, A. A. G. 1982. Algorithms for computing the volume and other integral properties of solids. ii. a family of algorithms based on representation conversion and cellular approximation. *Commun. ACM 25*, 9, 642–650.

OLHOFF, N., LUND, E., AND RASMUSSEN, J. 1992. Concurrent engineering design optimization in a cad environment. *Special report 16, Institute of Mechanical Engineering, Aalborg University*.

RAPPOPORT, A., SHEFFER, A., AND BERCOVIER, M. 1995. Volume-preserving free-form solid. In *Proceedings of the Third ACM Symposium on Solid and Physical Modeling*, ACM Press, 361–372.

RVACHEV, V. L. 1967. *Geometric Applications of Logic Algebra*. Naukova Dumka. In Russian.

RVACHEV, V. L. 1982. *Theory of R-functions and Some Applications*. Naukova Dumka. In Russian.

SARRAGA, R. 1982. Computation of surface areas in gmsolid. *IEEE Computer Graphics and Applications 2*, 7, 65 – 70.

SETHIAN, J. A., AND WIEGMANN, A. 2000. Structural boundary design via level set and immersed interface methods. *Journal of Computational Physics 163*, 2, 489–528.

SHAPIRO, V., AND TSUKANOV, I. 1999. Implicit functions with guaranteed differential properties. In *Proceedings of Fifth ACM Symposium on Solid Modeling and Applications*, 258–269.

SHAPIRO, V. 1988. Theory of R-functions and applications: A primer. *Technical Report, Cornell University* (November).

SOKOLOWSKI, J., AND ZOLESIO, J. P. 1992. *Introduction to Shape Optimization: Shape Sensitivity Analysis*. Springer-Verlag, New York.

TORTORELLI, D. A. 1993. A geometric representation scheme suitable for shape optimization. *Mechanics of Structures and Machines 21*, 95–121.

TSUKANOV, I., AND HALL, M. 2003. Data structure and algorithms for fast automatic differentiation. *International Journal for Numerical Methods in Engineering 56*, 13 (April), 1949–1972.

WANG, M. Y., WANG, X. M., AND GUO, D. M. 2003. A level set method for structural topology optimization. *Computer Methods in Applied Mechanics and Engineering 192*, 1-2, 227–246.

YANG, R. J., DEWHIRST, D. L., ALLISON, J. E., AND LEE, A. 1992. Shape optimization of connecting rod pin and using a generic model. *Finite Elements in Analysis and Design 11*, 257–264.