A Survey of Numerical Methods for Shape from Shading

Jean-Denis Durou, Maurizio Falcone, Manuela Sagona

IRIT
Université Paul Sabatier
118 route de Narbonne
31062 TOULOUSE Cedex 4

A Survey of Numerical Methods for Shape from Shading

Jean-Denis Durou[1]
Maurizio Falcone[2]
Manuela Sagona[2]

[1] Institut de Recherche en Informatique, Université Paul Sabatier, 118 route de Narbonne, 31062 Toulouse Cedex, France. E-mail:durou@irit.fr

[2] Dipartimento di Matematica, Università di Roma "La Sapienza", Piazzale Aldo Moro, 2, 00185 Roma, Italy. E-mail:{falcone,sagona}@mat.uniroma1.it

Rapport de recherche IRIT N° 2004–2–R

janvier 2004

Résumé

Plusieurs algorithmes ont été proposés pour résoudre le problème du *shape from shading*, et la publication du livre de Horn et Brooks [27] commence à dater. Dans ce rapport, nous faisons un tour d'horizon le plus à jour possible, en détaillant plusieurs algorithmes qui nous semblent particulièrement représentatifs de trois classes de méthodes : (i) les méthodes relevant de la résolution des équations aux dérivées partielles, (ii) les méthodes de minimisation et (iii) les méthodes faisant une approximation de l'équation de luminance. Un de nos buts est d'établir un protocole de comparaison rigoureux entre ces méthodes. À cette fin, nous décrivons brièvement chacune d'entre elles, en rappelant les hypothèses sur lesquelles elles sont basées, ainsi que leurs propriétés mathématiques. Nous avons recours à quelques tests sur images de synthèse aussi bien que sur images réelles, et nous comparons ces méthodes vis-à-vis de leurs temps de calcul et de leurs précisions dans la reconstruction des surfaces, à l'aide d'un certain nombre d'estimateurs. Nous testons également leurs comportements lorsque la direction de la source lumineuse est modifiée.

Mots-clés : *shape from shading*, équation de l'eikonale, méthodes numériques, comparaison d'algorithmes.

Abstract

Several algorithms have been suggested for the shape from shading problem, and some years have passed since the publication of Horn and Brooks' book [27]. In this survey paper, we try to update the approach in presenting some algorithms which seem to be particularly representative of three classes of methods: (i) methods based on partial differential equations, (ii) minimization methods, and (iii) methods approximating the image irradiance equation. One of the goals of this paper is to set the comparison of these methods on a firm basis. To this end, we provide a brief description of each method highlighting their basic assumptions and mathematical properties. We examine some numerical examples comparing the methods in terms of their efficiency and accuracy in the reconstruction of surfaces corresponding to synthetic as well as to real images. We also discuss their robustness faced with the presence of perturbations concerning the direction of the light source, and compare their accuracy in terms of a number of error indicators.

Keywords : shape from shading, eikonal equation, numerical methods, algorithms comparison.

Contents

1 Introduction

The shape from shading (SFS) problem has recently attracted several researchers, and a number of papers have appeared following the classical study by Horn and Brooks [27]. The reason for this renewed interest is probably due to the fact that, despite the simplicity of its formulation, the SFS problem deserves analysis and new approaches since a global method for its resolution under realistic assumptions is still lacking. Many technical questions (*e.g.*, the uniqueness of solutions without continuity assumptions) remain open. Moreover, new mathematical tools and numerical techniques have appeared in the last five years, so it seems to us appropriate to update the approach. Some of the new methods involve non-smooth solutions and several types of boundary conditions, guarantee convergence to an approximate solution under rather large assumptions, are reasonably fast and in some cases can be extended to deal with dark shadows (*i.e.*, black spots) in the image.

In this paper we will review a number of algorithms for the SFS problem which seem to be the most recent and/or representative in every class. Our classification of the algorithms takes into account the mathematical formulation behind the problem, the tools that are used to compute the solution, their features and the assumptions needed by each of them to compute a solution. Moreover, we will evaluate the performances on some test problems (on both synthetic and real images) and try to compare the accuracy of the methods with respect to several types of error. To the best of our knowledge, this is one of the first attempts to compare the algorithms although, following Horn and Brooks' book [27], other survey papers have appeared, *e.g.* [60].

We start by giving a brief outline of the SFS problem and introducing the basic assumptions. Consider the image of a surface given as a graph $z = u(x)$, $x = (x_1, x_2) \in \mathbb{R}^2$ and assume that there is a unique light source at infinity whose direction is indicated by the unit vector $\omega = (\omega_1, \omega_2, \omega_3) \in \mathbb{R}^3$. Also assume for simplicity that ω is given (an analysis of the consequences of an uncertainty on ω will be attempted in Section 4). As is well known (see [27]), the partial differential equation related to the SFS model can be derived by the "image irradiance equation":

$$R(n(x)) = I(x), \tag{1}$$

where I is the brightness function measured at all points x in the image, R is the reflectance function giving the value of the light reflection on the surface as a function of its orientation (*i.e.*, of its normal) and $n(x)$ is the unit normal to the surface at point $(x, u(x))$:

$$n(x) = \frac{1}{\sqrt{1 + p(x)^2 + q(x)^2}} \, (-p(x), -q(x), 1), \tag{2}$$

where $p = \partial u/\partial x_1$ and $q = \partial u/\partial x_2$, so that $\nabla u(x) = (p(x), q(x))$. Brightness function I is the datum in the model since it is measured on each pixel of the image, for example in terms of a greylevel (from 0 to 255). To construct a continuous model we will assume that I takes real values in the interval $[0, 1]$. Let us assume that u has to be reconstructed on a compact domain Ω, called the "reconstruction domain". Recalling that, for a Lambertian surface of uniform albedo equal to 1, $R(n(x)) = \omega \cdot n(x)$, Eq. (1) can be written, using (2):

$$I(x)\sqrt{1 + |\nabla u(x)|^2} + (\omega_1, \omega_2) \cdot \nabla u(x) - \omega_3 = 0, \qquad \text{for } x \in \Omega, \tag{3}$$

which is a first order non-linear partial differential equation of the Hamilton-Jacobi type. Most of the time, the equation in Ω must be complemented with a boundary condition in boundary $\partial\Omega$. For an image containing an "occluding boundary" (also called a "silhouette"), it is usual to consider it as boundary $\partial\Omega$. For example, in Fig. 1, if the part of the image representing the object in greylevels is Ω, then $\partial\Omega$ coincides with the occluding boundary.

A natural choice is to consider Dirichlet type boundary conditions in order to take into account (at least) two different possibilities. The first corresponds to the assumption that the surface is standing

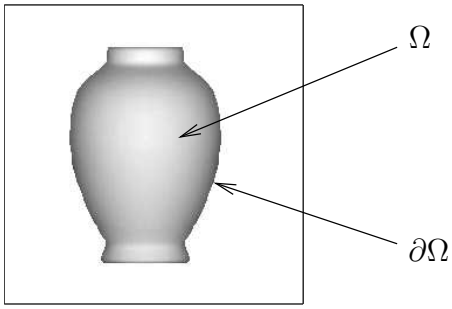


Figure 1: Image with occluding boundary, which might be used as boundary $\partial\Omega$.

on a flat background *i.e.*, we set:

$$u(x) = 0, \qquad \text{for } x \in \partial\Omega. \tag{4}$$

The second possibility occurs when the height of the surface on the occluding boundary is known, *e.g.*, this happens when we know (or assume) that the object is a surface of revolution around a given axis. This situation leads to the more general condition:

$$u(x) = g(x), \qquad \text{for } x \in \partial\Omega. \tag{5}$$

The solution of the above Dirichlet problem (3), (4) or (3), (5) will give the surface corresponding to brightness $I(x)$ measured in Ω. Points x where $I(x)$ is maximal correspond to the particular situation where ω and $n(x)$ point in the same direction: these points are usually called "singular points".

Finally, it should be mentioned that if Eq. (3) is not the most general equation of SFS, an even less general equation is used in many papers. If the light source is in the direction $\omega = (0, 0, 1)$, then (3) becomes the "eikonal equation":

$$|\nabla u(x)| = f(x), \qquad \text{for } x \in \Omega, \tag{6}$$

where:

$$f(x) = \sqrt{\frac{1}{I(x)^2} - 1}. \tag{7}$$

The present paper is organized as follows. Section 2 is devoted to a short description of the main classes of methods and of the most representative algorithms in every class. In Section 3 we describe our methodology to compare the methods, and give the mathematical definitions of the errors as well as some hints at reconstructing the physical quantities of the model (normals and light intensity) starting from the shapes computed by every algorithm. Section 4 contains the tests including error tables, figures and comments. Finally, in Section 5 we summarize the results of the study giving some directions for future research.

2 A Review of Shape from Shading Algorithms

A large number of SFS methods which use a great variety of mathematical tools are available. We decided to classify the methods into three categories: (i) methods of resolution of partial differential equations (PDEs), (ii) methods using optimization, and (iii) methods approximating the image irradiance equation (a very similar categorization is made in [60]). We will select one method from each category and subsequently compare them. We will also point out the assumptions necessary to make the algorithms converge. This information will also be very useful in analyzing the results of Section 4.

2.1 Methods of Resolution of PDEs

The eikonal equation has attracted much attention in the research community in PDEs for its wide range of applications. In the framework of the SFS problem several methods of resolution have been tested: characteristic strips expansion, power series expansion, approximation of viscosity solutions and level set methods.

2.1.1 Characteristic Strips Expansion

The first mention of 3D reconstruction using photometric cues is due to the Dutch astronomer Van Diggelen [56]. The first resolution was suggested by Rindfleisch [48], who demonstrated that, if the photometric behaviour of a surface follows certain properties, then the shape can be expressed as an integral, along a set of convergent straight lines. He implemented this computation on images of the Moon, claiming that its surface verifies the necessary photometric properties reasonably well. Later, Horn suggested calling this problem "shape from shading", and showed that the resolution proposed by Rindfleisch in a particular case could be generalized, while still using the characteristic strips expansion [24], under the following two conditions: (i) the function u has to be of class C^2; (ii) the 5-uplet (x_1, x_2, u, p, q) has to be known at every point of a curve called the "initial curve", which means in fact that two boundary conditions are needed simultaneously, one on u (Dirichlet boundary condition) and the other on (p, q) (Neumann boundary condition). The "characteristic lines" (which are the lines along which the integration has to be performed) can be of any form in the image plane, and this differs from the case studied by Rindfleisch. Besides the inherent defect of error accumulation, which is typical of every method of resolution using integration, the determination of these characteristic lines is a new problem in itself, since they are also defined through integration. Therefore, the accuracy of boundary conditions is much more crucial than for other methods. It follows that a certain number of obstacles must be overcome [14], *e.g.* the crossing of characteristic lines, which should normally occur only at singular points, or the presence of holes in Ω, which must be filled using secondary lines [24]. In fact, this method has essentially been useful for the theoretical study of the number of solutions of class C^2 of the eikonal equation [4, 42], and we subsequently decided not to select it for our comparative study.

2.1.2 Power Series Expansion

The first resolution of the eikonal equation using a power series expansion at a singular point is due to Bruss [6]. Of course, the solution must be analytical in a neighbourhood of the singular point, and the same assumption must hold for the brightness function. These assumptions seem too limiting since real images are not analytical. This explains why we did not select this method for our comparative study. In fact, as for the previous method, the power series expansion is interesting only for the study of the number of analytical solutions to the SFS problem. Using this method, Durou and Piau could exhibit a "non-visible deformation", *i.e.*, a continuous family of analytical shapes giving the same image [18].

2.1.3 Approximation of Viscosity Solutions

Starting from the paper by Lions, Rouy and Tourin [40], the most recent approach to the resolution of SFS uses the notion of "viscosity solutions" to first order PDEs, see for example [39], [3]. To give an idea, these are "almost-everywhere" (AE) solutions which can be obtained by being the limit in a family of solutions for regularized second order problems (the so-called "vanishing viscosity" method). These solutions are typically Lipschitz continuous solutions (but discontinuous viscosity solutions have also been considered in the literature, *cf.* [3]). The development of the theory of viscosity solutions for Hamilton-Jacobi type equations provides the right framework for the analysis of the SFS problem (see [11] for an up-to-date presentation of that theory).

Moreover, several algorithms have been proposed to compute viscosity solutions (see for example [12]). Finite difference numerical methods have been used for SFS in [49], [40] and [47] but similar results have been obtained by Oliensis and Dupuis [43] with an algorithm based on the Markov Chain approximation. Unfortunately, the Dirichlet problem (3), (4) can have several "weak solutions" in the viscosity sense and also several classical solutions (the so-called "concave/convex ambiguity", see [24]). As an example, all the surfaces represented in Fig. 2 are viscosity solutions of the same eikonal equation. The solution represented in Fig. 2(a) is the maximal solution and is smooth. All the non-smooth AE solutions which can be obtained by a reflection with respect to a horizontal axis, are still admissible weak solutions (Fig. 2(b)). In this example, the lack of uniqueness of the viscosity solutions is due to the existence of a singular point, where the right hand side of (6) vanishes. An additional effort is then needed to define which one will be our solution since the lack of uniqueness is also a big drawback when trying to compute a numerical solution. In order to circumvent those difficulties, the problem is usually solved by adding some information such as the height at the singular points, or the complete knowledge of a level curve (see for example [40, 31]).

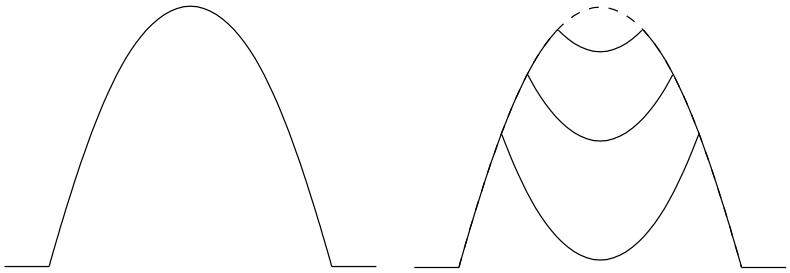


Figure 2: Illustration of the concave/convex ambiguity: (a) maximal solution and (b) AE solutions giving the same image.

More recently, an attempt has been made to eliminate the need for *a priori* additional information. In recent results in the theory of viscosity solutions the "maximal solution" without additional information apart from the equation was characterized, as was the construction of an algorithm which converges to that solution. A result by Ishii and Ramaswamy [30] guarantees that if I is continuous and the number of singular points is finite, then a unique maximal solution (in the viscosity sense) of (6), (4) exists. It should be noted that their result on the characterization of the maximal solution does not apply to the general situation when the set of singular points has a positive measure (this is the case, for example, of a flat roof). More general uniqueness results for maximal solutions of the eikonal equation have been recently obtained by Camilli and Siconolfi [7, 9, 10]. Several papers have followed this approach providing different algorithms to compute the maximal solution of (3), (4) which has been shown to be unique, see for example [20], [8], [50] and the references therein. One of these algorithms, the work of Falcone and Sagona [20], was selected for our comparative study. Some of its features are now discussed.

2.1.4 The Falcone and Sagona's Method

First we introduce the semi-Lagrangian approximation for (6), (4). Here and in the sequel we will assume for simplicity that $u \geq 0$ in Ω. This is not restrictive, since Eq. (6) depends only on ∇u and we can always add to u the minimum value of $u(x)$ on Ω to satisfy that requirement. In order to obtain an approximation scheme in the form of a fixed point problem it is useful to introduce the change of the unknown:

$$v(x) = 1 - \exp\left(-u(x)\right). \tag{8}$$

Note that by definition $0 \leq v \leq 1$. The SFS problem for the new unknown v is:

$$\left\{ \begin{array}{ll} v(x) + \max\limits_{a \in B_2(0,1)} \left\{ -\dfrac{a}{f(x)} \cdot \nabla v(x) - 1 \right\} = 0 & , \quad \text{for } x \in \Omega, \\ v(x) = 0 & , \quad \text{for } x \in \partial\Omega, \end{array} \right. \tag{9}$$

where f is given by (7) and assuming $u = 0$ in $\partial\Omega$ (the generalization to $u = g$ is straightforward). It is known (see for example [1]) that (9) has a unique continuous viscosity solution provided f is bounded and never vanishes in Ω.

Now we introduce the fully discrete scheme, discretizing $\mathbb{R}^2$ with a regular squared mesh of size δ. A node of this mesh, or pixel, is designated by $x_{i,j}$, and it is always possible to choose the axes and the origin in $\mathbb{R}^2$ so that $x_{i,j} = (i\,\delta, j\,\delta)$. Now, the set of indices (i,j) of the pixels belonging to Ω is designated by D, its cardinal by $N = card(D)$, and the following distinction is made: D_{in} contains the indices of D such that $x_{i,j} + \delta\, B_2(0,1)$ is included in Ω, and $D_{\text{bd}} = D \backslash D_{\text{in}}$ (we will impose the boundary conditions on the pixels $x_{i,j}$ having their indices in D_{bd}). We look for a solution v in the space of piecewise affine functions which are linear on the cells (P^1 finite element approximation):

$$\left\{ \begin{array}{lll} v(x_{i,j}) = \min\limits_{a \in B_2(0,1)} & \{ \exp(-h)\, v\,(x_{i,j}(a))\} & \\ & +1 - \exp(-h) & , \text{ for } (i,j) \in D_{\text{in}}, \\ v(x_{i,j}) = 0 & & , \text{ for } (i,j) \in D_{\text{bd}}, \end{array} \right. \tag{10}$$

where h is a small parameter, $x_{i,j}(a) = x_{i,j} + ha/f(x_{i,j})$ and $v\,(x_{i,j}(a))$ is computed by linear interpolation on the pixels of the grid, *i.e.*:

$$v\,(x_{i,j}(a)) = \sum_{(l,m) \in D} \lambda_{l,m}^{i,j}(a)\, v(x_{l,m}), \tag{11}$$

where coefficients $\lambda_{l,m}^{i,j}(a)$ are the local (barycentric) coordinates of point $x_{i,j}(a)$ with respect to the vertices of the cell containing that point, if these vertices belong to Ω, and otherwise $v\,(x_{i,j}(a)) = 0$. If V denotes the array containing the N unknown values $v(x_{i,j})$, then (10) can be reformulated as $V = H(V)$, $H : \mathbb{R}^N \to \mathbb{R}^N$. In [2, 19] it has been proved that the numerical solution to (10) exists and is unique by a fixed point argument on the iteration $V^{k+1} = H(V^k)$. Moreover, an *a priori* estimate for the convergence holds true provided function f is Lipschitz continuous (we refer the interested reader to [19] and [50] for a precise result). We note in passing that $0 \leq V \leq 1$ implies $0 \leq H(V) \leq 1$ and that $V_1 \leq V_2$ implies $H(V_1) \leq H(V_2)$. This monotonicity property implies that starting from a subsolution ($V^0 \leq H(V^0)$) the sequence will monotically converge to the fixed point. This property is crucial in speeding up convergence and it also helps to compute the maximal solution (see [20]). Other acceleration techniques for the eikonal equation can be found in [51] where the so-called "fast marching method" is described. It should be noted that the presence of singular points makes the vectorfield $a/f(x_{i,j})$ unbounded since f vanishes. This is an additional difficulty which is usually solved by truncating f below at the ε level. In this way the following perturbed (non-degenerate) eikonal equation is solved:

$$\left\{ \begin{array}{ll} |\nabla u(x)| = f_\varepsilon(x) & , \quad \text{for } x \in \Omega, \\ u(x) = 0 & , \quad \text{for } x \in \partial\Omega, \end{array} \right. \tag{12}$$

where $f_\varepsilon(x) = \max\{f(x), \varepsilon\}$. Naturally, the algorithm has to be analyzed with respect to ε. This was first done by Camilli and Siconolfi in [9], who showed that the solution of the perturbed problem actually converges to the solution of the original problem for ε going to 0. Following that result, Camilli and Grüne suggested a scheme which converges to the maximal solution of the eikonal equation in [8]. More recently, Sagona [50] has proved the convergence to the maximal solution of the above algorithm and has established an *a priori* error estimate in the L^∞ norm which takes into account all the

perturbation and discretization parameters. The crucial condition for the convergence to the maximal solution is:

$$h \left\| \frac{1}{f_\varepsilon} \right\|_\infty \leq \delta. \tag{13}$$

Using the definitions of the parameters given in this subsection, the algorithm corresponding to Falcone and Sagona's method [20], designated in the following by FS, is:

Fix h, ε and ζ.
Fix the starting vector V^0 for the fixed point iteration and set $k \leftarrow 0$.
Repeat:
  Compute $H(V^k)$ as in (10),
  $V^{k+1} \leftarrow H(V^k)$, $k \leftarrow k+1$,
until $|H(V^k) - V^k|_\infty < \zeta$.

The values of the parameters that were used for our tests are $h = \delta / \left\| \frac{1}{f_\varepsilon} \right\|_\infty$, which is the biggest admissible value of h for which Eq. (13) holds, $\varepsilon = 0.1425$, meaning that greylevel I is truncated at maximal value 0.99, and $\zeta = 10^{-6}$.

2.1.5 Level Set Methods

Another approach which produces a global solution to SFS is the one proposed by Kimmel and Bruckstein [32]. Their algorithm couples a level set method for the computation of several surfaces (weighted distance functions) with a merging procedure where those surfaces are coupled according to certain rules based on differential geometry to obtain an AE solution. The method assumes that the surface is regular, more precisely, it is a Morse surface in differential geometry terminology (see [22] for definitions and properties). The method consists of two major steps: the computation of (weighted) distance functions from all the singular points, and the merging of the above surfaces according to the topological properties of Morse surfaces. The algorithm can compute a global solution (which is an AE solution) of the eikonal equation in the reconstruction domain, only combining the local solutions obtained during the first step.

Let us sketch the two steps. For the first it suffices to solve the eikonal equation setting to 0 the value of the solution on a neighbourhood Ω_s of each singular point P_s, $s = 1, \ldots, M$ (no matter if P_s is a local maximum, a local minimum or a saddle point for the surface). Thus the algorithm starts solving M eikonal type equations coupled with Dirichlet boundary conditions:

$$\begin{cases} |\nabla u(x)| = f(x) & , \quad \text{for } x \in \Omega \setminus \Omega_s, \\ u(x) = 0 & , \quad \text{for } x \in \partial\Omega_s, \end{cases} \tag{14}$$

for $s = 1, \ldots, M$. To solve one of these problems the level set method [44] can be used: compute the solution of the following evolutive eikonal equation:

$$\begin{cases} \dfrac{\partial u}{\partial t}(x,t) + |\nabla u(x,t)| = f(x) & , \text{ for } (x,t) \in \Omega \setminus \Omega_s \times [0,T], \\ u(x,0) = 0 & , \text{ for } x \in \partial\Omega_s, \end{cases} \tag{15}$$

and pass to the limit for t tending to $+\infty$. Another possibility is to use a direct method for the stationary problem as a fast marching method (*cf.* [51] and [33]), or a semi-Lagrangian scheme introduced in the previous section with an acceleration technique. The first step gives the (approximately) correct solution in the case of a unique singular point which is a maximum or a minimum, provided the height of the surface is known in a small neighbourhood of that point. Note that, in the first step,

the above solutions are computed separately so that they do not interact. Since they are correct only in a neighbourhood of local maxima and minima they are called "local solutions" although they are defined everywhere in Ω. Denote by $u_s(x)$ the local solution corresponding to singular point P_s. In order to compute a global solution, in the second step all the local solutions $u_s(x)$, $s = 1, \ldots, M$, have to be merged. To this end the authors exploit smoothness conditions and topological considerations which essentially require a Morse surface.

This algorithm can also produce other AE solutions if the merging process is allowed to continue after the first AE solution is computed. This further means that it is quite difficult to define a stopping rule. Moreover, the fact that the merging process is separated (offline) and uses a dynamic list of local solutions makes it quite difficult to compare this algorithm to the others in terms of CPU time. This is why it was not selected for our comparative study.

2.2 Optimization Methods

Another category of algorithms which have been suggested are optimization methods based on the variational approach. The interested reader can find in [26] and Horn and in Brooks' book [27] several results and references relating to the variational approach. In this class of methods two basic ingredients must be chosen: the functional which has to be optimized (in fact, minimized) and the minimization method. It is surprising that a certain number of papers falling within the domain of optimization, do not clearly show the implications of these two choices, so that the choice of a functional is sometimes only guided by considerations on convergence. Even in [26], which is a major reference in the field, the discussion of several functionals is sometimes based on the possibility of finding an algorithm that converges towards a minimum. Indeed, it is *a priori* possible to freely combine any functional and any minimization algorithm. For reasons of space, we cannot here give a commentary on all the existing methods of SFS which use optimization. This would require a different article, since the number of papers dealing with this approach is considerable, starting with Strat [52], and followed by, amongst others, [29, 26, 21, 41, 25, 36, 53, 38, 58, 16]. The interested reader will find a comprehensive exposition of several optimization algorithms for SFS in Daniel's thesis [14].

In all the above papers, some regularity of the surface is assumed, and the approximate solutions computed are typically local minima of the functional. To obtain a global minimum, a global optimization algorithm has to be used, typically a stochastic algorithm like simulated annealing (see the recent paper by Crouzil, Descombes and Durou [13]). Naturally, the price to pay is a longer CPU time for the computation.

2.2.1 Choice of a Functional

The first difficulty encountered in the SFS problem is the choice of unknowns. The natural unknown is of course height u. Daniel and Durou showed [16] that it is almost impossible to deal with unknown u, because this induces an extreme slowness of the associated algorithms (several hours of CPU time for a very simple image of size 256×256!). But u appears, in the image irradiance equation, only through its first derivatives p and q, which are two non-independent functions since, if u is of class C^2:

$$\partial p/\partial x_2 = \partial q/\partial x_1. \tag{16}$$

The only problem with these unknowns is that p or q become infinite at each point x belonging to an occluding boundary. This problem is not a cause for concern if no point x in the reconstruction domain Ω is such that $I(x) = 0$. As Eq. (16) is a hard constraint on p and q, and recalling that $R(n(x)) = 1/(1 + p(x)^2 + q(x)^2)$ for a Lambertian surface of uniform albedo equal to 1, lit with

$\omega = (0, 0, 1)$, the most natural functional associated with Eqs. (6), (16) is:

$$\begin{aligned} \mathcal{F}_1(p, q, \mu) = \int_{x \in \Omega} & \left[1/(1 + p(x)^2 + q(x)^2) - I(x)\right]^2 \, dx \\ & + \int_{x \in \Omega} \mu(x) \left[\partial p / \partial x_2(x) - \partial q / \partial x_1(x)\right] \, dx, \end{aligned} \tag{17}$$

where μ is a Lagrange multiplier. Horn and Brooks [26] have proved that the three Euler equations associated with $\mathcal{F}_1$ can be reduced, for u being of class C^2, to the Euler equation associated with the following functional:

$$\mathcal{F}_2(u) = \int_{x \in \Omega} \left[1/(1 + \partial u / \partial x_1(x)^2 + \partial u / \partial x_2(x)^2) - I(x)\right]^2 \, dx, \tag{18}$$

but the algorithms dealing directly with u as unknown are very slow, as has already been stated. Horn and Brooks proposed [26] an alternative functional, called $\mathcal{F}_3$, where the constraint term becomes a penalty term. This means that, firstly, the constraint in (17) becomes squared and, secondly, the Lagrange multiplier μ is replaced by a positive constant λ_{int} called "integrability factor" (the choice of λ_{int} is arbitrary). This new functional is:

$$\begin{aligned} \mathcal{F}_3(p, q) = \int_{x \in \Omega} & \left[1/(1 + p(x)^2 + q(x)^2) - I(x)\right]^2 \, dx \\ & + \lambda_{\text{int}} \int_{x \in \Omega} \left[\partial p / \partial x_2(x) - \partial q / \partial x_1(x)\right]^2 \, dx. \end{aligned} \tag{19}$$

It leads to quite satisfactory results [53], even if Daniel and Durou [16] noted that, on noisy images, the noise will be wrongly interpreted as a consequence of a texture on the surface. Another problem with $\mathcal{F}_3$ is that it is not convex, because of its first term. A way to (partly) solve these two defects is to add another penalty term to $\mathcal{F}_3$, as suggested in [26]:

$$\mathcal{F}(p, q) = \mathcal{F}_3(p, q) + \lambda_{\text{smo}} \int_{x \in \Omega} \left[|\nabla p(x)|^2 + |\nabla q(x)|^2 \right] \, dx, \tag{20}$$

where λ_{smo} is a positive constant given the name of "smoothing factor" (the choice of λ_{smo} is arbitrary as well). The presence of two penalty terms seems to be redundant. However, it has been shown in [16] that these two terms have quite different effects, and that the smoothing term is usually more efficient than the integrability term. Moreover, an advantage in simultaneously using these two penalty terms is that it renders the minimization problem well-posed, even without any boundary condition on (p, q). This would not be true with only one penalty term (see [13]).

Let it also be mentioned that the three unknowns (u, p, q) have been dealt with simultaneously [25], and that other unknowns have been used: among them, the stereographic coordinates of the normal [29, 21, 37] which present the great interest of being bounded on the occluding boundaries, contrary to (p, q).

2.2.2 Choice of an Energy

The discretization of Eq. (16) at a pixel $x_{i,j}$ gives, using forward finite differences:

$$\frac{p_{i,j+1} - p_{i,j}}{\delta} \approx \frac{q_{i+1,j} - q_{i,j}}{\delta}, \tag{21}$$

and an estimate of $|\nabla p|^2$ at $x_{i,j}$ is:

$$\left(|\nabla p|^2\right)_{i,j} \approx \left(\frac{p_{i+1,j} - p_{i,j}}{\delta}\right)^2 + \left(\frac{p_{i,j+1} - p_{i,j}}{\delta}\right)^2. \tag{22}$$

Doing the same for q, we obtain the "energy" corresponding to $\mathcal{F}$:

$$\begin{aligned} E(G) &= \delta^2 \sum_{(i,j)\in D} \left[1/(1+{p_{i,j}}^2+{q_{i,j}}^2) - I_{i,j}\right]^2 \\ &\quad +\lambda_{\text{int}} \sum_{(i,j)\in \widetilde{D}} \left[(p_{i,j+1}-p_{i,j})-(q_{i+1,j}-q_{i,j})\right]^2 \\ &\quad +\lambda_{\text{smo}} \sum_{(i,j)\in \widetilde{D}} \left[(p_{i+1,j}-p_{i,j})^2+(p_{i,j+1}-p_{i,j})^2\right. \\ &\qquad\qquad \left. +(q_{i+1,j}-q_{i,j})^2+(q_{i,j+1}-q_{i,j})^2\right], \end{aligned} \tag{23}$$

where $\widetilde{D}$ is the subset of D containing the indices (i,j) such that $(i+1,j)$ and $(i,j+1)$ are in D, and G is the array containing the $2N$ values $(p_{i,j}, q_{i,j})$, for $(i,j) \in D$, which are the unknowns of the discrete problem. It has been proved in [26] that this energy is quasi-independent of δ.

2.2.3 Choice of a Minimization Method

As recalled by Szeliski in [53], two main strategies to find the minimum of an energy E exist: either minimizing E directly, or solving the new system σ of equations produced by $\nabla E = 0$. Surprisingly, most of the methods found in the literature use the second way of processing, which is subject to the two following drawbacks: many configurations different from the minimizers of E are solutions of σ; the resolution of a big system of non-linear equations is a difficult problem. For instance, the convergence of a Jacobi iteration is hard to prove (nevertheless, see [37]). It has even been proved that two Jacobi iterations found in the literature are definitely divergent [17]. For these reasons, we prefer the first way of processing *i.e.*, direct minimization. It is interesting to summarize the experience on this matter. Szeliski has used the conjugate gradient descent [53], a method of direct minimization which is particularly well adapted to quadratic energies, but not to SFS, since no proof of convergence exists. To avoid divergence, the iteration is stopped as soon as E increases. Even if the results presented in [53] are convincing (mostly in terms of CPU time), we preferred to select Daniel and Durou's method, for which convergence is guaranteed.

2.2.4 Daniel and Durou's Method

The use of a gradient descent based on line search (see [5]) was suggested in [16]: at each iteration k, a positive value d^k which is a minimizer (at least, a local minimizer) of the function $\phi^k(d) = E\left(G^k - d\ \nabla E(G^k)\right)$ must be found. The iteration is then defined by $G^{k+1} = G^k - d^k\ \nabla E(G^k)$. The iteration is stopped when $|\nabla E(G^k)|$ or d^k are less than a threshold. This is the minimization method selected because convergence is guaranteed. Finally, as already mentioned, another minimization method which has been successfully tested is the simulated annealing one [13], but the CPU time was so considerable that it would not have been reasonable to select this method in a comparative study.

Using the definitions given in the previous subsections, the algorithm corresponding to Daniel and Durou's method [16], designated in the following by DD, is:

Fix λ_{int}, λ_{smo}, β and γ.

Fix the starting vector G^0 for the iteration and set $k \leftarrow 0$.

Repeat:

  Compute $\nabla E(G^k)$,

  Find a local minimizer d^k of $\phi^k(d) = E\left(G^k - d\ \nabla E(G^k)\right)$,

  $G^{k+1} \leftarrow G^k - d^k\ \nabla E(G^k)$, $k \leftarrow k+1$,

until $|\nabla E(G^k)| < \beta\sqrt{2N}$ or $d^k < \gamma\sqrt{2N}$.

Vector G^0 contains values $(p_{i,j}, q_{i,j})_{(i,j)\in D}$ of a "starting shape". For the tests in Section 4, the starting shape used is the Gaussian curve $u(x) = 2\,e^{-|x|^2}$.

For DD, as well as for FS, the optimal configuration will be obtained in the limit. Of course, a stopping criterion has to be chosen. The thresholds on $|\nabla E(G^k)|$ and on d^k are proportional to $\sqrt{2N}$ because G is a vector in $\mathbb{R}^{2N}$. The values of λ_{int}, λ_{smo}, β and γ used are, respectively, 10.0, 50.0, 1.0 and 10^{-7}. Obviously, a bigger value as threshold reduces CPU time of the tests, but produces less accurate surfaces. Let it also be said that line search for d^k is performed assuming that $\phi^k(d)$ can be approximated by a parabola (quadratic approximation).

The algorithm described above provides as a result, an evaluation of G, but subsequently, a residual problem, called "integration", consists of computing the values $(u_{i,j})_{(i,j)\in D_{\text{in}}}$ from G and $(u_{i,j})_{(i,j)\in D_{\text{bd}}}$, which is the Dirichlet boundary condition. Some tests [16] have shown that Wu and Li's method [59] combined with that put forward by Horn and Brooks in [26] give excellent results. Wu and Li's method computes the height along diagonals and is very rapid. The result is then used as an initial shape for Horn and Brooks' method which is iterative. Of course, this computation is taken into account in the CPU time of DD.

2.3 Methods Approximating the Image Irradiance Equation

Finally, there exists a third class of SFS methods, whose name designates their common feature, recognizing that all of them make an approximation of the image irradiance equation. In [60], they have been classified into two sub-categories: "local methods" and "linear methods". These methods are generally easy to implement but produce rather disappointing results, except one method due to Tsai and Shah [54], that will be discussed later.

2.3.1 Local Methods

Local methods make the computation of the normal at each point in the image independently of the same computation for the other points, but they need a strong assumption on the observed surface, which is generally difficult to justify. Without that assumption, these methods would be unfeasible. The usual assumption on the surface is that it is locally spherical [45, 23] (except in [57] where it is assumed to be locally cylindrical, but the latter paper is concerned with radarclinometry, which is quite different from SFS). The consequence of making this very strong assumption is that the obtained normals are usually very far from being integrable, so that the computed shapes are often very bad, except if the observed shape exactly satisfies the local sphericity assumption, *i.e.*, if the shape is a part of a sphere. For these reasons, we decided to select none of these methods, even though it would have been far easier to implement them.

2.3.2 Linear Methods

While local methods deny the first difficulty of SFS, that is to say, that at each point in the image, there is only one equation for two unknowns p and q, linear methods deny the second difficulty of SFS, which is the non-linearity of this equation. The non-linearity of the image irradiance equation comes from the non-linearity of the reflectance function. Subsequently, is it reasonable to approximate the reflectance function by a linear function?

For a Lambertian surface of uniform albedo equal to 1, the reflectance function is equal to the following function of p and q:

$$r_L(p,q) = \frac{-\omega_1 p - \omega_2 q + \omega_3}{\sqrt{1+p^2+q^2}}. \tag{24}$$

When $\omega = (0,0,1)$, *i.e.*, in the case of frontal lighting, all the terms of odd orders in the development of r_L at $(p_0, q_0) = (0,0)$ vanish. Let us look for the terms of orders 1 and 2 in the development of r_L

at $(p_1, q_1) = (1, 1)$, setting $(p, q) = (p_1, q_1) + (\chi, \psi)$:

$$\begin{aligned} r_L(p,q) &= \frac{1}{\sqrt{1+(1+\chi)^2+(1+\psi)^2}} \\ &= \frac{1}{\sqrt{3}}\left(1 - \frac{\chi+\psi}{3} + \frac{\chi\psi}{3} + \cdots\right). \end{aligned} \tag{25}$$

When $\psi = \chi$, the ratio of the term of order 2 to the term of order 1 is equal to $-\chi/2$, so there is no way to make r_L equivalent to its development of order 1, elsewhere than in a neighbourhood of (p_1, q_1). We would come to the same conclusion with oblique lighting, even if Pentland proved in [46] that, the greater the angle between the observer's direction and the lighting direction, the better the approximation of r_L by its linear development at (p_0, q_0). So, linear SFS cannot be used efficiently for a Lambertian surface (unless (p, q) is quasi-invariant on the whole surface). Consequently, it appears that this reduces the interest of the linear approach, even if the resolution techniques which follow from it, using either finite differences [35, 55] or Fourier transform [46], are convincing.

Among the linear methods, the only method that can be applied to a variety of situations is that of Tsai and Shah, since the method uses local linearization. In [60], Tsai and Shah's method is one of the six which have been compared. Even if it quantitatively obtains the worst rank (total error equal to 59.3, compared to 41.3 for the best method, which shows that the scores are very close), it is qualitatively the best method on synthetic images, while the greatest part of the scores is obtained on real images, for which all the reconstructions are very bad. For this reason, and because it is easy to implement[1], we decided to choose Tsai and Shah's method in our study as representative of the third class of SFS methods. We now describe this method.

2.3.3 Tsai and Shah's Method

Tsai and Shah use a development of the reflectance function to the first order centered at each point in the image [54], contrary to other linear methods. The basic idea of the method is to approximate $p_{i,j}$ and $q_{i,j}$ directly by the following finite differences:

$$\begin{cases} p_{i,j} \approx \dfrac{u_{i,j} - u_{i-1,j}}{\delta}, \\ q_{i,j} \approx \dfrac{u_{i,j} - u_{i,j-1}}{\delta}. \end{cases} \tag{26}$$

By these approximations, Tsai and Shah obtain a system of equations of the following type:

$$r\left(\frac{u_{i,j} - u_{i-1,j}}{\delta}, \frac{u_{i,j} - u_{i,j-1}}{\delta}\right) = I_{i,j}, \tag{27}$$

where r is the function such that $R(n) = r(p, q)$. Provided that the height is known on the boundary (Dirichlet condition), this system contains as many equations as unknowns, thus overcoming the first difficulty of SFS: since Tsai and Shah's method consists of the resolution of a well-determined system, the problem becomes well-posed in the sense of Hadamard. Nevertheless, the non-linearity of the equations remains. Trying to solve Eqs. (27), $(i, j) \in D$, by propagation starting from the boundary, would be destined to failure. In order to be sure, consider the case of a Lambertian surface with frontal lighting, for which (27) can be rewritten:

$$\left(\frac{u_{i,j} - u_{i-1,j}}{\delta}\right)^2 + \left(\frac{u_{i,j} - u_{i,j-1}}{\delta}\right)^2 = \frac{1}{{I_{i,j}}^2} - 1. \tag{28}$$

[1]Even so, the source code is available by anonymous ftp under the `pub/tech_paper/survey` directory, at `eustis.cs.ucf.edu` (132.170.108.42).

If $u_{i-1,j}$ and $u_{i,j-1}$ are known, then this equation is of degree 2 in $u_{i,j}$. When the data are non-noisy, it may be hoped that it admits two solutions, among which the choice is a problem (this ambiguity is connected to the concave/convex ambiguity). The problem is even more serious for noisy data, since it may be that Eq. (28) has no solution.

Tsai and Shah's method is a variant of the well-known Newton-Raphson method. The aim is to solve system (27), $(i,j) \in D$, by means of the following iteration:

$$u_{i,j}^{k+1} = u_{i,j}^{k} - \frac{e_{i,j}^{k}}{e_{i,j}^{k\,\prime}}, \tag{29}$$

where $e_{i,j}^{k}$ and $e_{i,j}^{k\,\prime}$ are defined by:

$$\left\{ \begin{array}{rl} e_{i,j}^{k} = & r\left(\dfrac{u_{i,j}-u_{i-1,j}}{\delta}, \dfrac{u_{i,j}-u_{i,j-1}}{\delta}\right) - I_{i,j}, \\ e_{i,j}^{k\,\prime} = & -\dfrac{1}{\delta}\left[r_p\left(\dfrac{u_{i,j}-u_{i-1,j}}{\delta}, \dfrac{u_{i,j}-u_{i,j-1}}{\delta}\right)\right. \\ & \left. +r_q\left(\dfrac{u_{i,j}-u_{i-1,j}}{\delta}, \dfrac{u_{i,j}-u_{i,j-1}}{\delta}\right)\right], \end{array} \right. \tag{30}$$

and where r_p and r_q denote the two partial derivatives of r. In fact, system (27), $(i,j) \in D$, can be rewritten as $F(U) = 0$, where $U = (u_{i,j})_{(i,j)\in D}$ is a vector of $\mathbb{R}^N$ and $F : \mathbb{R}^N \to \mathbb{R}^N$. For such a system, the Newton-Raphson method consists of the following iteration:

$$U^{k+1} = U^k - J^{-1}(U^k)\, F(U^k), \tag{31}$$

where $J(U^k)$ designates the Jacobian matrix of F at iteration k. Many terms in matrix $J(U^k)$ are null (three terms per row at most are non-null) but the computation of $J^{-1}(U^k)$ can be tedious. Iteration (29) suggested by Tsai and Shah is a much simpler variant of (31). There is no guarantee that $e_{i,j}^{k\,\prime}$ will not vanish. In order to avoid the latter defect, Tsai and Shah put forward the following modification of (29):

$$u_{i,j}^{k+1} = u_{i,j}^{k} - K_{i,j}^{k}\, e_{i,j}^{k}, \tag{32}$$

where $K_{i,j}^{k}$ has to respect the following two constraints: $K_{i,j}^{k}$ should be "close" to $1/e_{i,j}^{k\,\prime}$ if $e_{i,j}^{k\,\prime}$ is non-null, and equal to 0 otherwise; $K_{i,j}^{k}$ should tend to 0 when k tends to $+\infty$. This leads to the following algorithm [54], called TS in the course of this article:

Fix W and $k_{\max}$.

Fix the starting vector $U^0 \leftarrow 0$.

For $(i,j) \in D$, $S_{i,j}^{0} \leftarrow 1$.

For $k \leftarrow 0 \cdots k_{\max}$ and for $(i,j) \in D$:

Compute $e_{i,j}^{k}$ and $e_{i,j}^{k\,\prime}$ as in (30),

$$K_{i,j}^{k} \leftarrow \frac{S_{i,j}^{k}\, e_{i,j}^{k\,\prime}}{W + S_{i,j}^{k}\left[e_{i,j}^{k\,\prime}\right]^2},$$

Compute $u_{i,j}^{k+1}$ as in (32),

$$S_{i,j}^{k+1} \leftarrow \left[1 - K_{i,j}^{k}\, e_{i,j}^{k\,\prime}\right] S_{i,j}^{k}.$$

Following Tsai and Shah, we took $W = 0.01$. Whereas a proof of convergence exists for FS and for DD, the TS method is usually not convergent, according to our tests. The stopping criterion consists

of fixing the value of $k_{\max}$, in the knowledge that an iteration for which the configuration is optimal exists. Our tests have led us to consider that the 20^{th} iteration is globally optimal. Since in [54], $k_{\max} = 5$ is recommended as a good choice, we decided to test these two values for each image.

Three features of TS arose through the tests: even if the scene is lit from the observer's direction, it is necessary to model r by an oblique lighting (there is no obvious explanation of this statement); in (27), the value of δ must be equal to 1, although any value should *a priori* be admissible for δ ; finally, the greylevel of the pixels outside Ω must be put to 0 ("black background"). Otherwise, the results would not be the same as in [54].

3 Methodology

Obviously, a comparison of SFS methods belonging to the same class can be performed in a rather natural way, such as in Szeliski's paper [53], where an existing method due to Horn [25] is improved using several techniques. Conversely, the same comparison is not an easy task when the methods use different mathematical tools and algorithms, as in [60] and in the present work. In this section, we describe: (i) the data necessary for the selected methods, (ii) the panel of images selected for the tests, and (iii) the measures by which to numerically compare the performances of the methods.

3.1 Data Necessary for the Selected Methods

In order to compare different methods aiming at solving a common problem, we should use a common set of data for all the methods. What amount of information is needed by the three methods that have been selected? Of course, the fundamental data are the image and the knowledge of the reconstruction domain Ω. A great difference between FS, DD and TS is that FS and TS directly compute u, whereas DD first computes (p, q), and then u. Most of the methods similar to DD require knowledge of (p, q) in $\partial\Omega$, which is of no use for FS and TS, but DD does not need this knowledge, consequently this was a convincing argument to select DD as representative of the methods using optimization. In fact, our three selected methods need the same boundary conditions: knowledge of u in $\partial\Omega$ (a version of DD using no knowledge at all on the boundary has been suggested in [13], but its performances are not very satisfactory). It could seem that TS does not require any knowledge on the boundary but, as already mentioned, the greylevel is put to 0 outside Ω, and this implies that u will be uniformly equal to 0 there. For FS and DD, the values of the height in $\partial\Omega$ can be fixed in a more natural way. All the tests are done at least with $u = 0$ as a boundary condition. In addition, when the real height g in $\partial\Omega$ is known and is not uniformly equal to 0, a second test for FS and DD with $u = g$ as boundary condition will be done.

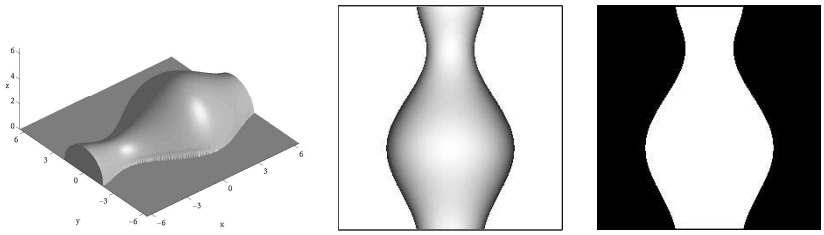
The choice of images for the tests is a serious difficulty. Of course, it would seem logical to choose images which conform to the basic assumptions of SFS. As noted in [60], this is an easy task for synthetic images but much more difficult for real images, even if a process by Daniel and Durou [15] creates real images verifying almost all the assumptions of SFS. It is interesting to compute synthetic images because it cannot be that there is no solution ("impossible images", see for example [34, 28]) and, moreover, by this means the reconstructed and real shapes can be compared. Of course, it could occur that a computed image corresponds to an infinite family of shapes, as already mentioned [18], but if u is fixed in $\partial\Omega$, then any ambiguity is avoided. For our tests, we decided to select three synthetic and two real images.

3.2 Panel of Images Selected for the Tests

The three synthetic images were computed from known shapes. We computed all the synthetic images on the same domain $[-6.4, 6.4]^2$ of $\mathbb{R}^2$ projected on a regular squared mesh of 256×256 pixels, meaning that the size of the mesh is $\delta = 12.8/256 = 0.05$ (for one of the shapes, we also tested images of sizes

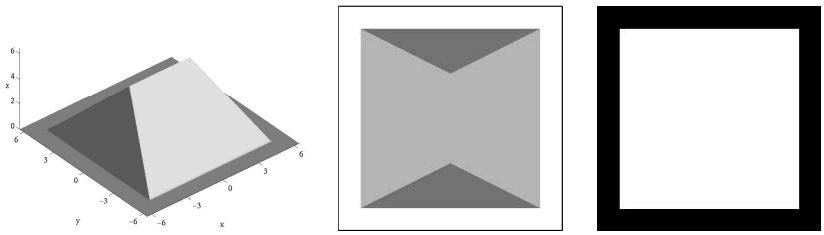
128×128, 64×64 and 32×32, implying that δ will be equal to 0.1, 0.2 and 0.4). Then, p and q are evaluated by differentiation and, finally, I is computed through image irradiance equation (3), depending on the value of ω (we first used $\omega = (0, 0, 1)$ and then some other values of ω for one of the shapes).

The first shape represents a vase lying on a flat background and called "synthetic vase" (SV) in the following. It is defined by $u_{\mathrm{SV}}(x) = \sqrt{P(\overline{x}_1)^2 - {x_2}^2}$ on $\Omega_{\mathrm{SV}} = \{x \in]-0.5, 0.5[\times\mathbb{R}, P(\overline{x}_1)^2 \geq {x_2}^2\}$ and $u_{\mathrm{SV}}(x) = 0$ elsewhere, with $P(\overline{x}_1) = -138.24\,{\overline{x}_1}^6 + 92.16\,{\overline{x}_1}^5 + 84.48\,{\overline{x}_1}^4 - 48.64\,{\overline{x}_1}^3 - 17.60\,{\overline{x}_1}^2 + 6.40\,\overline{x}_1 + 3.20$ and $\overline{x}_1 = x_1/12.8$. The second shape represents a "Canadian tent" (CT) lying on a flat background and is defined by $u_{\mathrm{CT}}(x) = \min(-2\,|x_1| + 10.24, -|x_2| + 5.12)$ on $\Omega_{\mathrm{CT}} = [-5.12, 5.12]^2$, and $u_{\mathrm{CT}}(x) = 0$ elsewhere. The third shape is a "digital elevation model" (DEM) defined by $u_{\mathrm{DEM}}(x) = 3\,(1 - \overline{x}_1)^2 \exp\left(-{\overline{x}_1}^2 - (\overline{x}_2 + 1)^2\right) - 10\,(\overline{x}_1/5 - {\overline{x}_1}^3 - {\overline{x}_2}^5) \exp\left(-{\overline{x}_1}^2 - {\overline{x}_2}^2\right) - 1/3 \exp\left(-(\overline{x}_1 + 1)^2 - {\overline{x}_2}^2\right)$, with $(\overline{x}_1, \overline{x}_2) = x/1.6$. For DEM, we use $[-6.4, 6.4]^2$ as reconstruction domain Ω_{DEM}, from which points x near the edges such that $I(x) \geq 254/255$, have been withdrawn (by analogy with SV and CT, these points constitute a flat background). These three shapes are represented in Figs. 3(a), 4(a) and 5(a). The three corresponding synthetic images of size 256×256 computed with $\omega = (0, 0, 1)$ are represented in Figs. 3(b), 4(b) and 5(b). Finally, the three reconstruction domains are represented in Figs. 3(c), 4(c) and 5(c).



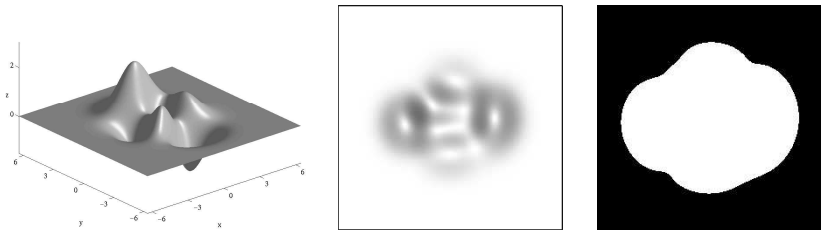
(a) (b) (c)

Figure 3: SV : (a) exact shape, (b) 256×256 image and (b) Ω_{SV}.



(a) (b) (c)

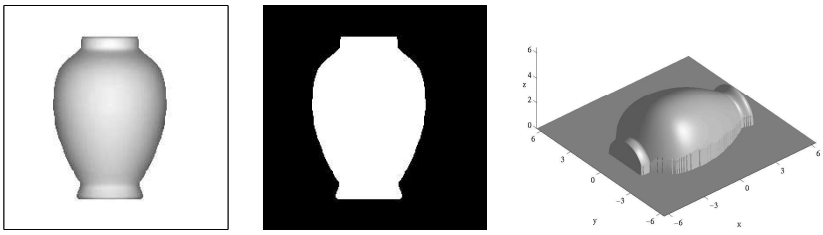
Figure 4: CT : (a) exact shape, (b) 256×256 image and (b) Ω_{CT}.



(a) (b) (c)

Figure 5: DEM : (a) exact shape, (b) 256×256 image and (b) Ω_{DEM}.

The two real images selected for the tests are of size 256×256. These two images represent a vase, called "real vase" (RV), and the moulding of an elk, which was obtained through Daniel and Durou's process already mentioned [15]. They are represented in Figs. 6(a) and 7(a). The reconstruction domains of these two images have been determined differently: Ω_{RV} is constituted by the pixels situated on the vase (Fig. 6(b)), while Ω_{elk} contains no pixel near the edges of the image such that $I \geq 254/255$, considering that such pixels correspond to a flat background (Fig. 7(b)). Note that the greylevels of the pixels outside the reconstruction domains have been put to 1, giving the illusion that the shapes are really lying on a flat background. The problem is then to obtain the corresponding shapes. For the vase, which is radially symmetrical, this can be done while assuming that the symmetry axis is parallel to the image plane. By this assumption, we can subsequently detect the occluding boundary and deduce the complete shape by rotation around the symmetry axis (see Fig. 6(c)). As the original image is of size 256×256, we interpret the corresponding scene as if δ were equal to 0.05, that is to say, as if the orthogonal projection of the scene on the image plane were of size 12.8×12.8. It is not that easy to get the real shape of the elk. Nevertheless, thanks to direct measures[2], we could obtain the set of 3D points represented in Fig. 7(c). Unfortunately, it can be noted that the points are not arranged in a regular way. Thus, we would have had to coregister and resample this set of points, in order to produce a dense map of the height, and this was not done. This explains why for the elk we will not compare the reconstructed shapes with the real shape.



(a) (b) (c)

Figure 6: RV : (a) original 256×256 image, (b) Ω_{RV} and (c) estimated shape.

(a) (b) (c)

Figure 7: Elk : (a) original 256×256 image, (b) Ω_{elk} and (c) measured shape.

The choice of these five images as test problems was motivated by the fact that they have different peculiarities: the synthetic vase is smooth and is a rather simple shape; the Canadian tent is an example of a synthetic non-smooth surface with $u = 0$ boundary condition; the DEM is smooth and has the simple boundary condition $u = 0$, but its shape is complicated; the real vase is equivalent to a noisy version of SV; the elk is equivalent to a noisy version of DEM.

[2]We would like to thank Michel Labarrère, metrologist at the *Département de Génie Mécanique*, École Nationale Supérieure d'Ingénieurs de Constructions Aéronautiques, Toulouse, France.

3.3 Evaluation of the Performances of the SFS Methods

The most natural criterion for the evaluation of a shape reconstruction is of course to measure the difference between reconstructed shape $\tilde{u}$ and real shape u. An interesting alternative is to compare input image I with approximate image $\tilde{I}$ computed from the reconstructed shape, particularly for the elk, the shape of which is not available. Finally, when (p, q) is known for the real shape, it would seem interesting to compare it with its estimate $(\tilde{p}, \tilde{q})$ or, even better, to compare normal n to its estimate $\tilde{n}$, because these two vectors are normed and their comparison is equivalent to the measure of an angle in $\mathbb{R}^3$. Once surface $\tilde{u}$ has been computed, we usually adopt an estimate of normal $\tilde{n}$ which is coherent with the computation of intensity $\tilde{I}$. As can be seen in Fig. 8, for each pixel $x_{i,j}$, we consider the four triangles $T^1_{i,j}$, $T^2_{i,j}$, $T^3_{i,j}$ and $T^4_{i,j}$ having $x_{i,j}$ in common. On each triangle the image intensity is well

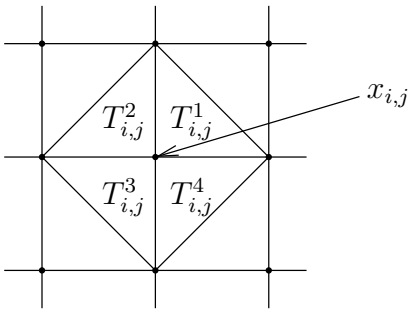


Figure 8: Triangles used in the image intensity approximation.

defined if the plane passing through the value of the numerical solution at its vertices is considered. Related approximated image intensities $\tilde{I}^1_{i,j}$, $\tilde{I}^2_{i,j}$, $\tilde{I}^3_{i,j}$ and $\tilde{I}^4_{i,j}$ are given by the scalar product between the normal to the corresponding plane and ω. The image intensity approximation on pixel $x_{i,j}$ is done in the following way:

$$\tilde{I}_{i,j} = \min\{\tilde{I}^1_{i,j}, \tilde{I}^2_{i,j}, \tilde{I}^3_{i,j}, \tilde{I}^4_{i,j}\}. \tag{33}$$

Subsequently, we chose as approximation $\tilde{n}_{i,j}$ the normal to the plane which corresponds to the lowest intensity.

We compared the methods in accordance with three error estimators. Weighted L^1 and L^2 errors and L^∞ error are defined as follows:

$$\begin{cases} |\Delta f|_1 = \dfrac{1}{N} \displaystyle\sum_{(i,j)\in D} \left|\tilde{f}_{i,j} - f_{i,j}\right|, \\ |\Delta f|_2 = \left[\dfrac{1}{N} \displaystyle\sum_{(i,j)\in D} \left|\tilde{f}_{i,j} - f_{i,j}\right|^2\right]^{1/2}, \\ |\Delta f|_\infty = \displaystyle\max_{(i,j)\in D} \left\{\left|\tilde{f}_{i,j} - f_{i,j}\right|\right\}, \end{cases} \tag{34}$$

for every known f and computable $\tilde{f}$ (recall that $N = \mathrm{card}(D)$). These estimators are commonly called, respectively: the Mean Absolute Error, the Root Mean Square Error and the Maximal Absolute Error. They will be applied to u, n and I, that is to say, we will measure the accuracy of the reconstructed shapes with nine numerical values, except for the elk, for which u and n are not available (note that the three errors on u are not bounded, while the three on n are bounded by 2.0 and the three on I are bounded by 1.0). In comparison, the survey made in [60] used only three numerical values: the Mean Absolute Error on u and on (p, q), and the Standard Deviation on u.

Finally, we also report the CPU time of each reconstruction, expressed in seconds. All the tests were done on a Sun Enterprise E420.

4 Tests

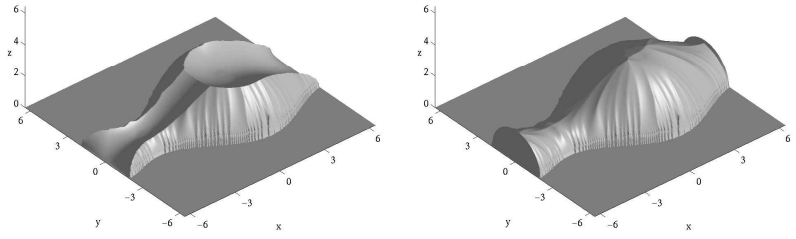
This section is devoted to the analysis of the tests. At the end of this section we will also discuss the robustness of the algorithms faced with a perturbation in the estimated direction of the light source.

4.1 Test 1: Synthetic Vase

Start from the synthetic image of a vase represented in Fig. 3(b). Two boundary conditions were considered. The first is the homogeneous Dirichlet boundary condition $u = 0$ in $\partial\Omega_{\mathrm{SV}}$ which is simple, but clearly false at the top and at the bottom of the vase. The second boundary condition corresponds to $u = g_{\mathrm{SV}}$ in $\partial\Omega_{\mathrm{SV}}$, where g_{SV} is a continuous function vanishing on the left and on the right hand side of the vase, and having the value of the real height at the top and at the bottom. All the errors for both choices are provided.

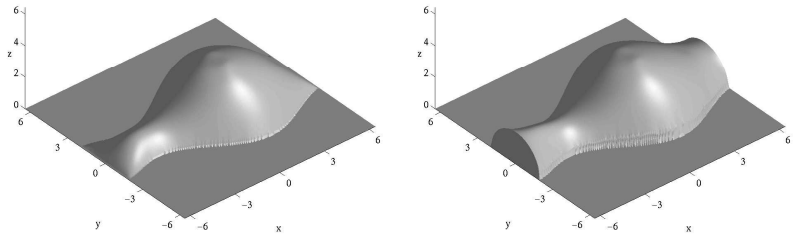
The qualitative behaviour of the surface is well represented in the solutions obtained by the three methods (Figs. 9, 10 and 11). It can be observed that the TS scheme also gives reasonable information at the boundary (at the top and at the bottom of the vase); this is surprising because that method imposes $u = 0$ as a boundary condition. The effect of boundary conditions is clearly visible in the results obtained by the FS and DD methods. The FS method does not introduce any smoothing, thus the small kinks on the boundary (which is approximated by square cells) produce several lines of discontinuity for $\tilde{n}$ and for $\tilde{I}$ inside Ω_{SV}. Moreover, the wrong boundary condition $u = 0$ produces a wrong shape due to the concave/convex ambiguity in the model (see Fig. 9(a)). On the other hand, the DD method produces a smooth solution due to the regularization term which is present in the functional and does not seem to be affected by the concave/convex ambiguity. The reconstruction of the image after the shape is based on the simple rule $I(x) = \omega \cdot n(x)$ and on the methodology described in Section 3. As can be seen in Figs. 12, 13 and 14, the approximation in terms of I is quite satisfactory for the FS and DD methods. Table 3 shows that all the errors relating to the DD and FS methods have the same order of magnitude in all cases. Note that the results of the TS method generally do not improve by passing from 5 to 20 iterations.

Conversely, Table 1 shows that the DD algorithm is more accurate in the reconstruction of u and is also faster with respect to the FS algorithm (by a factor of 10). Table 2 shows the error in the approximation of the normals. It can be seen that all the methods have the same order of magnitude even if the DD method (on both boundary conditions), and the TS method have lower error tests compared to the FS method.



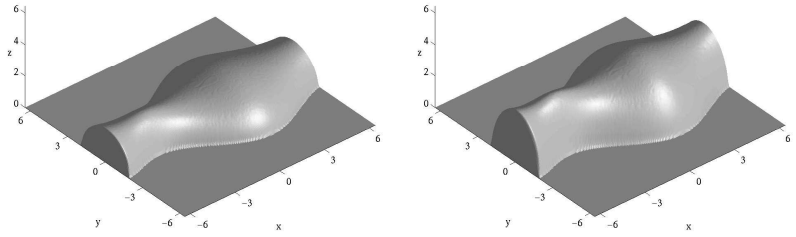
(a) (b)

Figure 9: FS on SV 256×256: computed shapes with (a) $u = 0$ and (b) $u = g_{\mathrm{SV}}$ on the boundary.



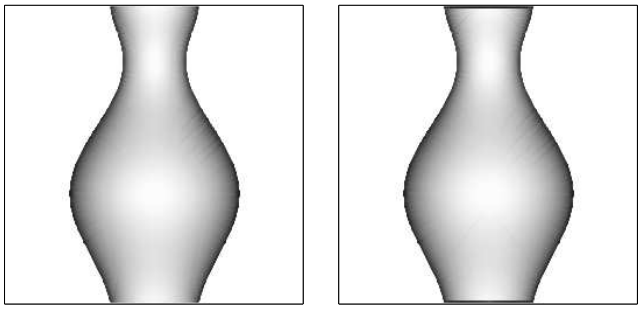
(a) (b)

Figure 10: DD on SV 256×256: computed shapes with (a) $u = 0$ and (b) $u = g_{\mathrm{SV}}$ on the boundary.



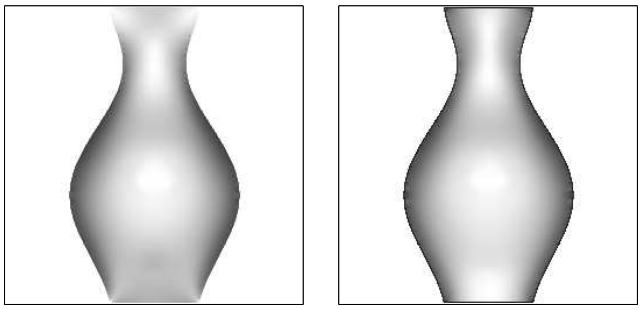
(a) (b)

Figure 11: TS on SV 256×256: computed shapes with (a) 5 and (b) 20 iterations.



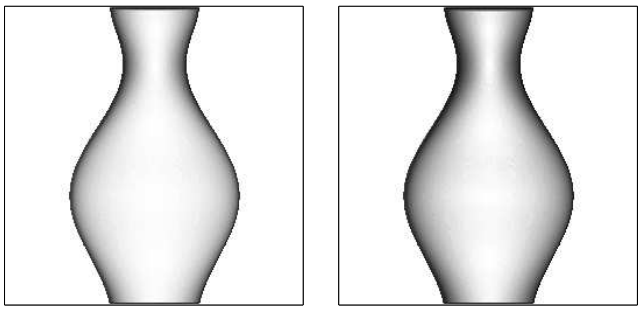
(a) (b)

Figure 12: FS on SV 256×256: computed images with (a) $u = 0$ and (b) $u = g_{\mathrm{SV}}$ on the boundary.



(a) (b)

Figure 13: DD on SV 256×256: computed images with (a) $u = 0$ and (b) $u = g_{\mathrm{SV}}$ on the boundary.



(a) (b)

Figure 14: TS on SV 256×256: computed images with (a) 5 and (b) 20 iterations.

Table 1: SV 256×256: errors on the computed shape and CPU time.

method	boundary condition	$\lvert\Delta u\rvert_1$	$\lvert\Delta u\rvert_2$	$\lvert\Delta u\rvert_\infty$	CPU
FS	$u = 0$	8.2427e-01	1.0171e+00	1.9494e+00	213.0
FS	$u = g_{\mathrm{SV}}$	3.0514e-01	3.1881e-01	5.7427e-01	151.4
DD	$u = 0$	6.1508e-01	6.9996e-01	1.9413e+00	15.93
DD	$u = g_{\mathrm{SV}}$	1.8528e-01	2.2771e-01	4.3051e-01	16.12
TS (5 it.)	$u = 0$	8.0773e-01	9.6358e-01	1.6910e+00	0.29
TS (20 it.)	$u = 0$	5.0433e-01	6.3173e-01	1.7352e+00	1.17

Table 2: SV 256×256: errors on the computed normals.

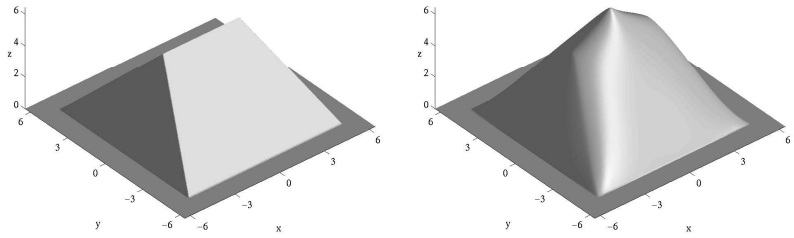
method	boundary condition	$\lvert\Delta n\rvert_1$	$\lvert\Delta n\rvert_2$	$\lvert\Delta n\rvert_\infty$
FS	$u = 0$	8.2598e-01	9.2353e-01	1.9802e+00
FS	$u = g_{\mathrm{SV}}$	8.6121e-01	9.2691e-01	1.9802e+00
DD	$u = 0$	1.9534e-01	2.8676e-01	1.3111e+00
DD	$u = g_{\mathrm{SV}}$	1.1287e-01	1.6782e-01	1.4219e+00
TS (5 it.)	$u = 0$	2.9061e-01	3.3502e-01	1.4190e+00
TS (20 it.)	$u = 0$	2.4689e-01	3.2353e-01	1.4268e+00

Table 3: SV 256×256: errors on the computed image.

method	boundary condition	$\|\Delta I\|_1$	$\|\Delta I\|_2$	$\|\Delta I\|_\infty$
FS	$u = 0$	1.0302e-02	2.9603e-02	7.0336e-01
FS	$u = g_{\mathrm{SV}}$	1.3442e-02	6.7775e-02	8.4251e-01
DD	$u = 0$	6.6356e-02	1.1086e-01	9.3725e-01
DD	$u = g_{\mathrm{SV}}$	3.8146e-02	7.5277e-02	9.7255e-01
TS (5 it.)	$u = 0$	1.1342e-01	1.4375e-01	9.6863e-01
TS (20 it.)	$u = 0$	7.9675e-02	1.2318e-01	9.8039e-01

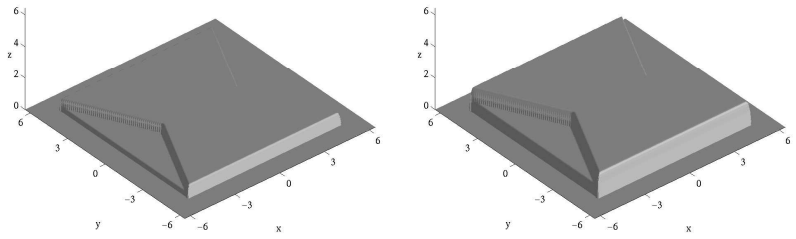
4.2 Test 2: Canadian Tent

The second test we discuss is related to a synthetic image representing a Canadian tent with two different slopes (see Fig. 4(b)). The important difference with the first example consists of the sharp edges of the surface and the corresponding discontinuities on the input image. In this case we have a natural boundary condition, the homogeneous Dirichlet boundary condition $u = 0$. As can be seen in Figs. 15 and 16, the best result is obtained by the FS method due to its capacity to follow the jumps in the gradient of u. The DD method provides a very smooth surface with no kinks and the TS method computes a quite flat surface. The qualitative behaviour of the three methods is confirmed numerically in Tables 4, 5 and 6. On the other hand, FS CPU time is longer than the others. We also tested the robustness of the algorithms by implementing them on three reduced images. We used the information given by three different images of the same object: a 128×128 pixels image, a 64×64 pixels image and a 32×32 pixels image. All the numerical results are shown in Tables 7-15, where it can be observed that the accuracy on the full-size image of the three methods is also confirmed on the low resolution images. In terms of reconstruction of the normals, Tables 8, 11 and 14 show that the FS method still produces the best results maintaining a reasonably good order of accuracy.



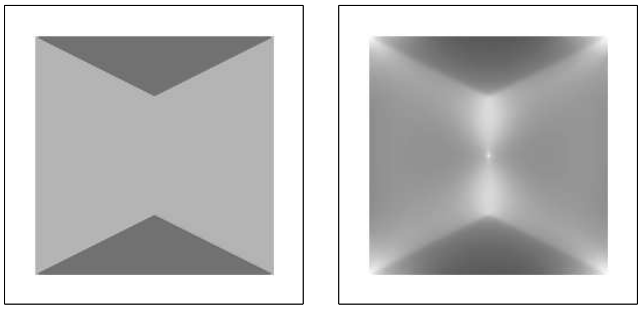
(a) (b)

Figure 15: CT 256×256: computed shapes with (a) FS and (b) DD.



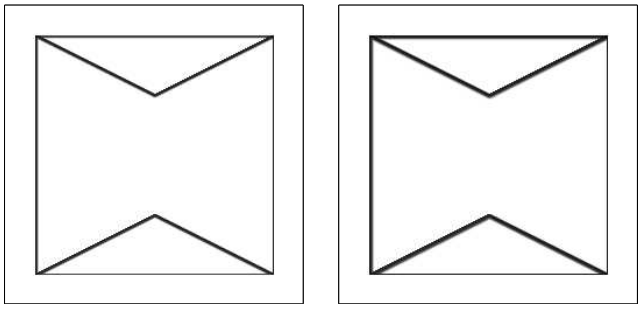
(a) (b)

Figure 16: TS on CT 256×256: computed shapes with (a) 5 and (b) 20 iterations.



(a) (b)

Figure 17: CT 256×256: computed images with (a) FS and (b) DD.



(a) (b)

Figure 18: TS on CT 256×256: computed images with (a) 5 and (b) 20 iterations.

Table 4: CT 256×256: errors on the computed shape and CPU time.

method	boundary condition	$\lvert\Delta u\rvert_1$	$\lvert\Delta u\rvert_2$	$\lvert\Delta u\rvert_\infty$	CPU
FS	$u = 0$	2.2916e-02	2.6338e-02	4.9804e-02	233.7
DD	$u = 0$	5.3548e-01	7.0539e-01	1.8559e+00	21.10
TS (5 it.)	$u = 0$	1.6368e+00	2.0723e+00	4.8254e+00	0.30
TS (20 it.)	$u = 0$	1.3216e+00	1.6715e+00	4.4172e+00	1.18

Table 5: CT 256×256: errors on the computed normals.

method	boundary condition	$\|\Delta n\|_1$	$\|\Delta n\|_2$	$\|\Delta n\|_\infty$
FS	$u = 0$	5.0607e-03	5.2369e-02	1.1694e+00
DD	$u = 0$	2.5981e-01	3.2013e-01	1.0148e+00
TS (5 it.)	$u = 0$	8.1733e-01	8.3676e-01	1.6977e+00
TS (20 it.)	$u = 0$	8.1256e-01	8.3403e-01	1.7148e+00

Table 6: CT 256×256: errors on the computed image.

method	boundary condition	$\|\Delta I\|_1$	$\|\Delta I\|_2$	$\|\Delta I\|_\infty$
FS	$u = 0$	8.6081e-04	1.0059e-02	2.5989e-01
DD	$u = 0$	8.4982e-02	9.8069e-02	5.4118e-01
TS (5 it.)	$u = 0$	3.5148e-01	3.7020e-01	6.5098e-01
TS (20 it.)	$u = 0$	3.5014e-01	3.6950e-01	6.7843e-01

Table 7: CT 128×128: errors on the computed shape and CPU time.

method	boundary condition	$\|\Delta u\|_1$	$\|\Delta u\|_2$	$\|\Delta u\|_\infty$	CPU
FS	$u = 0$	7.0956e-02	7.8727e-02	1.3952e-01	29.3
DD	$u = 0$	5.2053e-01	7.1120e-01	1.8999e+00	2.38
TS (5 it.)	$u = 0$	1.6223e+00	2.0610e+00	4.6998e+00	0.07
TS (20 it.)	$u = 0$	1.3093e+00	1.6609e+00	4.2395e+00	0.29

Table 8: CT 128×128: errors on the computed normals.

method	boundary condition	$\|\Delta n\|_1$	$\|\Delta n\|_2$	$\|\Delta n\|_\infty$
FS	$u = 0$	1.0247e-02	7.5503e-02	1.1694e+00
DD	$u = 0$	2.8919e-01	3.5042e-01	1.0496e+00
TS (5 it.)	$u = 0$	7.9391e-01	8.2343e-01	1.6615e+00
TS (20 it.)	$u = 0$	7.8723e-01	8.1784e-01	1.6970e+00

Table 9: CT 128×128: errors on the computed image.

method	boundary condition	$\lvert\Delta I\rvert_1$	$\lvert\Delta I\rvert_2$	$\lvert\Delta I\rvert_\infty$
FS	$u = 0$	1.8065e-03	1.5092e-02	2.5989e-01
DD	$u = 0$	8.4821e-02	1.0098e-01	5.3333e-01
TS (5 it.)	$u = 0$	3.3409e-01	3.5641e-01	5.9608e-01
TS (20 it.)	$u = 0$	3.3322e-01	3.5412e-01	6.5098e-01

Table 10: CT 64×64: errors on the computed shape and CPU time.

method	boundary condition	$\lvert\Delta u\rvert_1$	$\lvert\Delta u\rvert_2$	$\lvert\Delta u\rvert_\infty$	CPU
FS	$u = 0$	4.3838e-02	5.5006e-02	1.1555e-01	3.5
DD	$u = 0$	4.5714e-01	5.9182e-01	1.6306e+00	0.43
TS (5 it.)	$u = 0$	1.6463e+00	2.0819e+00	4.6998e+00	0.02
TS (20 it.)	$u = 0$	1.3158e+00	1.6692e+00	4.2174e+00	0.07

Table 11: CT 64×64: errors on the computed normals.

method	boundary condition	$\lvert\Delta n\rvert_1$	$\lvert\Delta n\rvert_2$	$\lvert\Delta n\rvert_\infty$
FS	$u = 0$	2.1362e-02	1.1165e-01	1.1694e+00
DD	$u = 0$	3.1316e-01	3.8043e-01	1.2770e+0
TS (5 it.)	$u = 0$	7.6224e-01	7.9692e-01	1.5857e+00
TS (20 it.)	$u = 0$	7.3764e-01	7.8453e-01	1.6599e+00

Table 12: CT 64×64: errors on the computed image.

method	boundary condition	$\lvert\Delta I\rvert_1$	$\lvert\Delta I\rvert_2$	$\lvert\Delta I\rvert_\infty$
FS	$u = 0$	4.0087e-03	2.3793e-02	2.5989e-01
DD	$u = 0$	8.2742e-02	1.0299e-01	5.2157e-01
TS (5 it.)	$u = 0$	3.1088e-01	3.3316e-01	5.5294e-01
TS (20 it.)	$u = 0$	2.9621e-01	3.2371e-01	5.9608e-01

Table 13: CT 32×32: errors on the computed shape and CPU time.

method	boundary condition	$\lvert\Delta u\rvert_1$	$\lvert\Delta u\rvert_2$	$\lvert\Delta u\rvert_\infty$	CPU
FS	$u = 0$	4.1594e-02	5.1906e-02	1.1354e-01	0.4
DD	$u = 0$	5.2870e-01	6.1879e-01	1.4067e+00	0.44
TS (5 it.)	$u = 0$	1.7052e+00	2.1300e+00	4.5383e+00	0.01
TS (20 it.)	$u = 0$	1.3421e+00	1.7021e+00	4.0174e+00	0.02

Table 14: CT 32×32: errors on the computed normals.

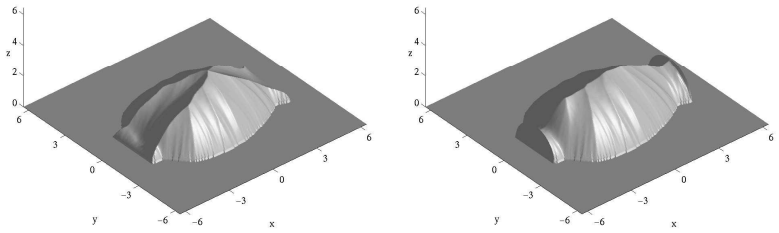
method	boundary condition	$\lvert\Delta n\rvert_1$	$\lvert\Delta n\rvert_2$	$\lvert\Delta n\rvert_\infty$
FS	$u = 0$	4.5875e-02	1.7030e-01	1.1694e+00
DD	$u = 0$	3.3752e-01	4.1798e-01	1.3431e+00
TS (5 it.)	$u = 0$	7.2485e-01	7.6012e-01	1.4383e+00
TS (20 it.)	$u = 0$	6.8353e-01	7.3464e-01	1.5826e+00

Table 15: CT 32×32: errors on the computed image.

method	boundary condition	$\lvert\Delta I\rvert_1$	$\lvert\Delta I\rvert_2$	$\lvert\Delta I\rvert_\infty$
FS	$u = 0$	9.4680e-03	3.9691e-02	2.5989e-01
DD	$u = 0$	8.2974e-02	1.0690e-01	5.0196e-01
TS (5 it.)	$u = 0$	2.9234e-01	3.1537e-01	5.4902e-01
TS (20 it.)	$u = 0$	2.6545e-01	2.8996e-01	5.4902e-01

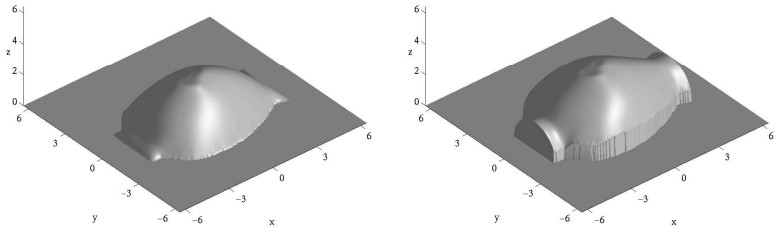
4.3 Test 3: Real Vase

Consider the real image of a vase represented in Fig. 6(a). As in Test 1 (SV) we considered two boundary conditions. It can be seen from Figs. 19, 20 and 21 and from Tables 16, 17 and 18 that the numerical results show behaviour in the three methods which is similar to Test 1. It can be noted that the DD surface seems to be more accurate in the reconstruction of the shape with respect to FS and TS. However, comparison of the approximate image indicates a better performance of the FS method. Table 17 confirms the DD method to be more accurate in the reconstruction of the normals. Also in this case CPU times show a big difference between the FS and the DD and TS methods, which are faster.



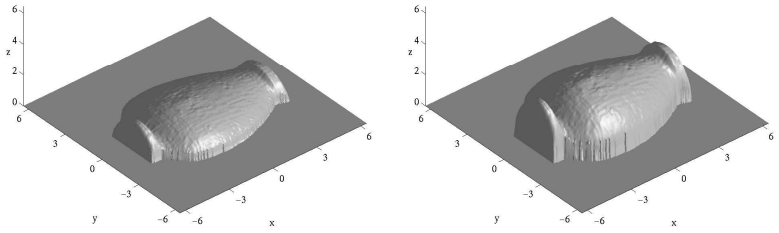
(a) (b)

Figure 19: FS on RV 256×256: computed shapes with (a) $u = 0$ and (b) $u = g_{\mathrm{RV}}$ on the boundary.



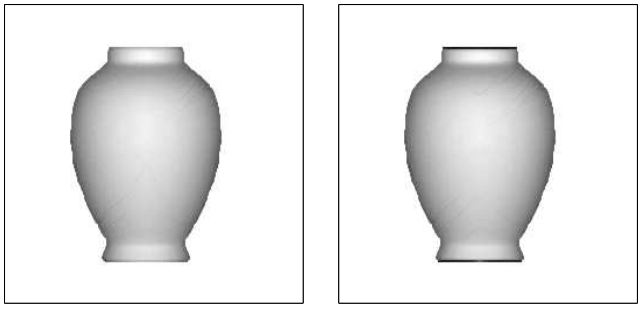
(a) (b)

Figure 20: DD on RV 256×256: computed shapes with (a) $u = 0$ and (b) $u = g_{\mathrm{RV}}$ on the boundary.



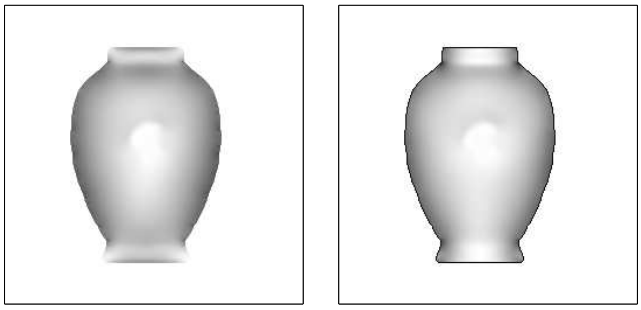
(a) (b)

Figure 21: TS on RV 256×256: computed shapes with (a) 5 and (b) 20 iterations.



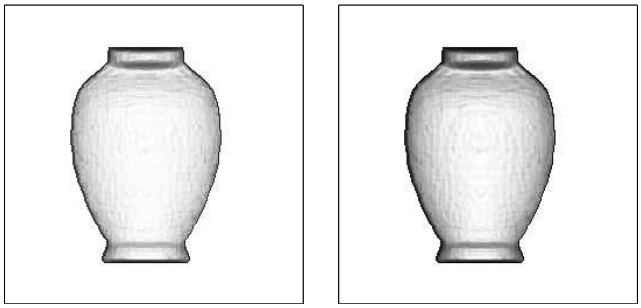
(a) (b)

Figure 22: FS on RV 256×256: computed images with (a) $u = 0$ and (b) $u = g_{\text{RV}}$ on the boundary.



(a) (b)

Figure 23: DD on RV 256×256: computed images with (a) $u = 0$ and (b) $u = g_{\text{RV}}$ on the boundary.



(a) (b)

Figure 24: TS on RV 256×256: computed images with (a) 5 and (b) 20 iterations.

Table 16: RV 256×256: errors on the computed shape and CPU time.

method	boundary condition	$\lvert\Delta u\rvert_1$	$\lvert\Delta u\rvert_2$	$\lvert\Delta u\rvert_\infty$	CPU
FS	$u = 0$	7.6917e-01	8.2515e-01	1.7351e+00	111.1
FS	$u = g_{\mathrm{RV}}$	5.9975e-01	6.3824e-01	1.0558e+00	97.1
DD	$u = 0$	9.6423e-01	9.9563e-01	1.9027e+00	9.82
DD	$u = g_{\mathrm{RV}}$	1.5982e-01	2.3143e-01	5.5699e-01	9.97
TS (5 it.)	$u = 0$	1.2788e+00	1.3721e+00	1.9471e+00	0.30
TS (20 it.)	$u = 0$	4.0640e-01	4.6392e-01	1.2744e+00	1.17

Table 17: RV 256×256: errors on the computed normals.

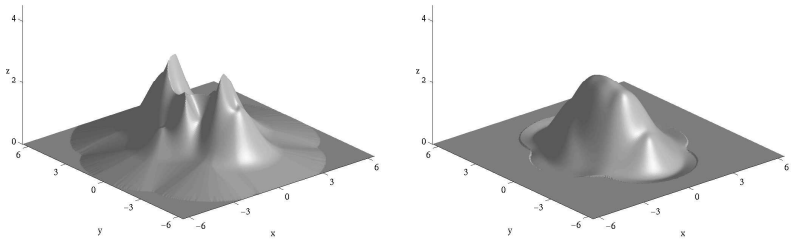
method	boundary condition	$\lvert\Delta n\rvert_1$	$\lvert\Delta n\rvert_2$	$\lvert\Delta n\rvert_\infty$
FS	$u = 0$	8.7987e-01	9.4258e-01	1.8112e+00
FS	$u = g_{\mathrm{RV}}$	8.8779e-01	9.4742e-01	1.9548e+00
DD	$u = 0$	2.0691e-01	2.9205e-01	1.1767e+00
DD	$u = g_{\mathrm{RV}}$	1.4648e-01	1.9883e-01	1.1624e+00
TS (5 it.)	$u = 0$	3.2854e-01	4.1963e-01	1.8056e+00
TS (20 it.)	$u = 0$	2.5840e-01	3.9742e-01	1.9489e+00

Table 18: RV 256×256: errors on the computed image.

method	boundary condition	$\lvert\Delta I\rvert_1$	$\lvert\Delta I\rvert_2$	$\lvert\Delta I\rvert_\infty$
FS	$u = 0$	9.0322e-03	2.7889e-02	5.7544e-01
FS	$u = g_{\mathrm{RV}}$	1.4107e-02	5.7083e-02	6.2418e-01
DD	$u = 0$	4.8166e-02	7.7248e-02	6.7843e-01
DD	$u = g_{\mathrm{RV}}$	5.3361e-02	8.3276e-02	5.8039e-01
TS (5 it.)	$u = 0$	1.6722e-01	1.8949e-01	7.0980e-01
TS (20 it.)	$u = 0$	1.2244e-01	1.6358e-01	7.8039e-01

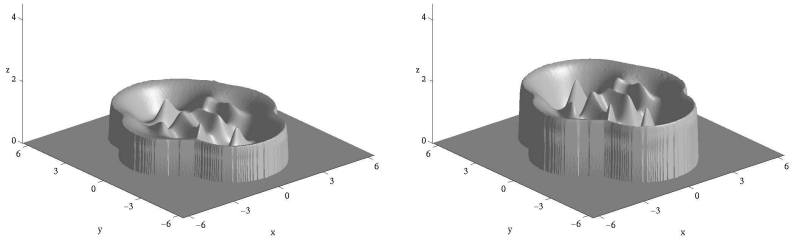
4.4 Test 4: Digital Elevation Model

This test uses the image of a DEM (see 5(b)). It is first noted that the qualitative behaviour of the reconstructed shapes is very different for the three methods. The FS method computes the maximal solution, which does not coincide with the real surface, while the DD method computes one of the admissible minimum energy configurations. Note that the TS shapes reflect a surprising behaviour on the boundary. In any case, we can observe that the order of magnitude of the errors on the shapes and on the normals is the same for the three methods (see Tables 19 and 20), while Table 21 shows the different behaviour of the image errors as can also be seen in Figs. 27 and 28. In this test CPU times underline the slow behaviour of the FS method.



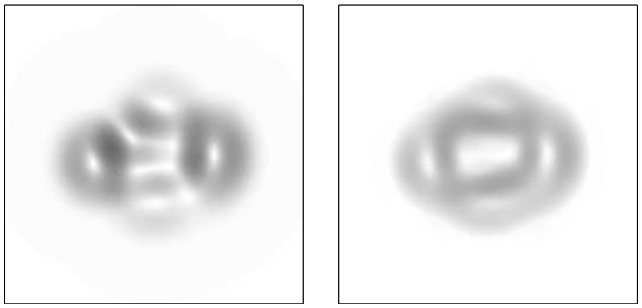
(a) (b)

Figure 25: DEM 256×256: computed shapes with (a) FS and (b) DD.



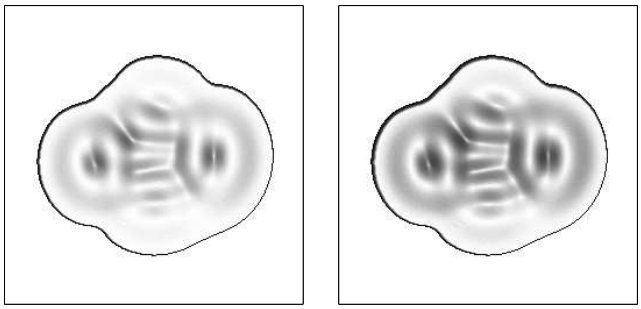
(a) (b)

Figure 26: TS on DEM 256×256: computed shapes with (a) 5 and (b) 20 iterations.



(a) (b)

Figure 27: DEM 256×256: computed images with (a) FS and (b) DD.



(a) (b)

Figure 28: TS on DEM 256×256: computed images with (a) 5 and (b) 20 iterations.

Table 19: DEM 256×256: errors on the computed shape and CPU time.

method	boundary condition	$\|\Delta u\|_1$	$\|\Delta u\|_2$	$\|\Delta u\|_\infty$	CPU
FS	$u = 0$	4.4378e-01	7.3677e-01	3.3683e+00	508.9
DD	$u = 0$	5.8636e-01	8.8070e-01	2.5147e+00	10.76
TS (5 it.)	$u = 0$	8.4777e-01	9.4934e-01	2.6841e+00	0.30
TS (20 it.)	$u = 0$	1.2647e+00	1.3803e+00	3.3035e+00	1.18

Table 20: DEM 256×256: errors on the computed normals.

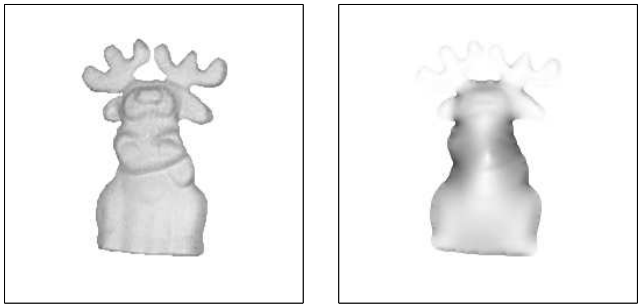
method	boundary condition	$\|\Delta n\|_1$	$\|\Delta n\|_2$	$\|\Delta n\|_\infty$
FS	$u = 0$	3.0029e-01	4.7975e-01	1.8015e+00
DD	$u = 0$	5.9660e-01	7.0774e-01	1.5120e+00
TS (5 it.)	$u = 0$	6.4650e-01	7.3523e-01	1.4573e+00
TS (20 it.)	$u = 0$	7.4149e-01	8.2825e-01	1.5992e+00

Table 21: DEM 256×256: errors on the computed image.

method	boundary condition	$\|\Delta I\|_1$	$\|\Delta I\|_2$	$\|\Delta I\|_\infty$
FS	$u = 0$	4.4879e-03	8.8001e-03	1.1227e-01
DD	$u = 0$	6.0825e-02	8.1257e-02	3.0980e-01
TS (5 it.)	$u = 0$	1.2361e-01	2.2017e-01	9.6078e-01
TS (20 it.)	$u = 0$	1.5492e-01	2.4792e-01	9.7255e-01

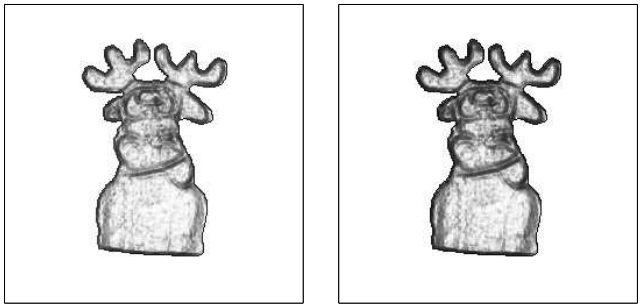
4.5 Test 5: Elk

This test deals with a real greylevel image representing the plaster of an elk (Fig. 7(a) was kindly provided by Daniel, and it has been used in [16] as a test problem). For this image we have no additional information either for the exact surface or for the normals, so only computed images will be compared. The pecularity of this example is the complex shape (as well as the complex silhouette). In Table 22, I errors are shown, from which it can be noted that the FS method produces slightly better results compared to the DD and TS methods. This is confirmed from computed images shown in Figs. 29 and 30. FS again shows a slow performance, even if CPU times are in this case closer than those in other tests.



(a) (b)

Figure 29: Elk 256×256: computed images with (a) FS and (b) DD.



(a) (b)

Figure 30: TS on elk 256×256: computed images with (a) 5 and (b) 20 iterations.

Table 22: Elk 256×256: errors on the computed image and CPU time.

method	boundary condition	$\lvert\Delta I\rvert_1$	$\lvert\Delta I\rvert_2$	$\lvert\Delta I\rvert_\infty$	CPU
FS	$u = 0$	2.1501e-02	3.9609e-02	4.4411e-01	51.3
DD	$u = 0$	1.1754e-01	1.5165e-01	4.9412e-01	9.15
TS (5 it.)	$u = 0$	1.8686e-01	2.4668e-01	7.8431e-01	0.29
TS (20 it.)	$u = 0$	2.4675e-01	3.1291e-01	8.1176e-01	1.19

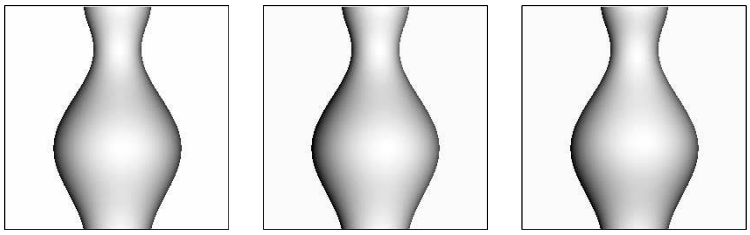
4.6 Test 6: Non-Frontal Lighting

In this last test we wanted to investigate the stability of the three methods with respect to perturbations in the light source direction. We considered three images of the synthetic vase that were obtained using three different non-frontal light source directions, *viz.*:

$$\begin{cases} \omega^1 = (\sin(\pi/36)\cos(\pi/2), \sin(\pi/36)\sin(\pi/2), \cos(\pi/36)), \\ \omega^2 = (\sin(\pi/18)\cos(\pi/2), \sin(\pi/18)\sin(\pi/2), \cos(\pi/18)), \\ \omega^3 = (\sin(\pi/18)\cos(3\pi/4), \sin(\pi/18)\sin(3\pi/4), \cos(\pi/18)). \end{cases} \tag{35}$$

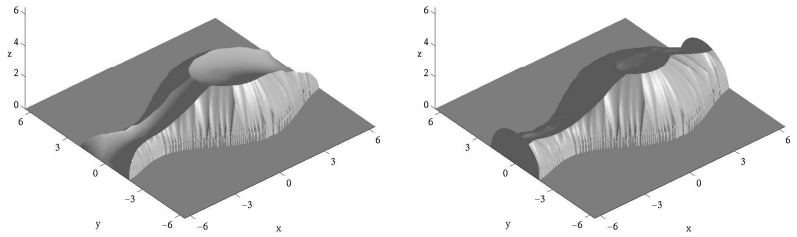
The three images are represented in Fig. 31. Note that the shadows in the background were not taken into account, since the background is outside Ω_{SV}. These images were processed as if the light source had been frontal. The two boundary conditions $u = 0$ (wrong) and $u = g_{\mathrm{SV}}$ (right) were considered. Since the numerical results showed that the behaviour of the three methods is similar for the three light source directions, we limit ourselves to commenting on the Figures and Tables relating to ω^1 as a representative example.

The first remark to make is that the order of magnitude of errors on shapes and normals is the same for all three methods (see Tables 23 and 24) while Table 25 shows some differences in errors on the computed images. Examining Figs. 32, 33 and 34, it seems that the DD method is reasonably accurate, while the FS and TS methods present different problems on the neck and the belly of the vase. However, the qualitative behaviour of the images is well represented in the solution obtained by the three methods (Figs. 35, 36 and 37). CPU times confirm the better speed of DD and TS compared to FS.



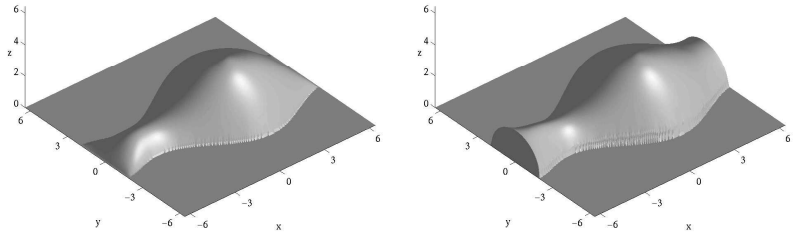
(a) (b) (c)

Figure 31: SV 256×256 images: (a) $\omega = \omega^1$, (b) $\omega = \omega^2$ and (c) $\omega = \omega^3$.



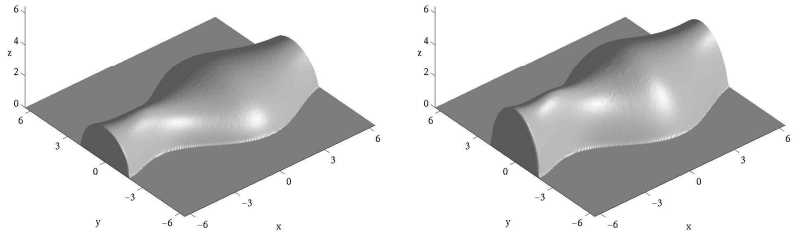
(a) (b)

Figure 32: FS on SV ($\omega = \omega^1$): computed shapes with (a) $u = 0$ and (b) $u = g_{\mathrm{SV}}$ on the boundary.



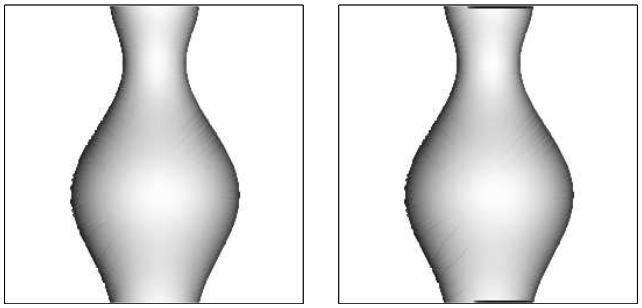
(a) (b)

Figure 33: DD on SV ($\omega = \omega^1$): computed shapes with (a) $u = 0$ and (b) $u = g_{\mathrm{SV}}$ on the boundary.



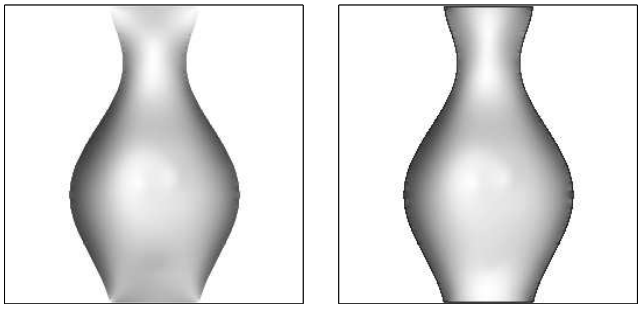
(a) (b)

Figure 34: TS on SV ($\omega = \omega^1$): computed shapes with (a) 5 and (b) 20 iterations.



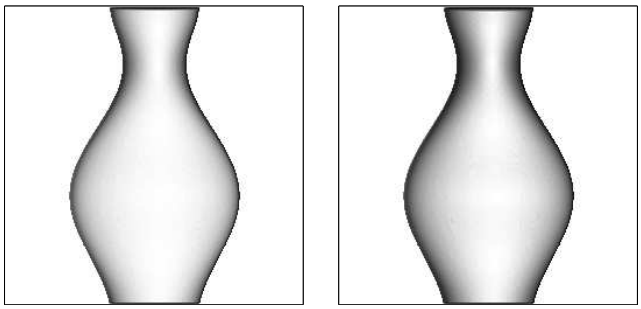
(a) (b)

Figure 35: FS on SV ($\omega = \omega^1$): computed images with (a) $u = 0$ and (b) $u = g_{\mathrm{SV}}$ on the boundary.



(a) (b)

Figure 36: DD on SV ($\omega = \omega^1$): computed images with (a) $u = 0$ and (b) $u = g_{\mathrm{SV}}$ on the boundary.



(a) (b)

Figure 37: TS on SV ($\omega = \omega^1$): computed images with (a) 5 and (b) 20 iterations.

Table 23: SV ($\omega = \omega^1$): errors on the computed shape and CPU time.

method	boundary condition	$\lvert\Delta u\rvert_1$	$\lvert\Delta u\rvert_2$	$\lvert\Delta u\rvert_\infty$	CPU
FS	$u = 0$	8.5447e-01	1.0385e+00	1.9417e+00	202.9
FS	$u = g_{\mathrm{SV}}$	4.5097e-01	5.4056e-01	1.1515e+00	164.1
DD	$u = 0$	6.0531e-01	7.1198e-01	1.9413e+00	16.09
DD	$u = g_{\mathrm{SV}}$	2.2491e-01	2.7927e-01	6.1746e-01	16.52
TS (5 it.)	$u = 0$	8.1592e-01	9.7035e-01	1.7008e+00	0.31
TS (20 it.)	$u = 0$	5.2965e-01	6.5663e-01	1.7534e+00	1.27

Table 24: SV ($\omega = \omega^1$): errors on the computed normals.

method	boundary condition	$\lvert\Delta n\rvert_1$	$\lvert\Delta n\rvert_2$	$\lvert\Delta n\rvert_\infty$
FS	$u = 0$	8.3284e-01	9.2490e-01	1.9722e+00
FS	$u = g_{\mathrm{SV}}$	8.4394e-01	9.2414e-01	1.9722e+00
DD	$u = 0$	2.2894e-01	3.0732e-01	1.3111e+00
DD	$u = g_{\mathrm{SV}}$	1.5513e-01	2.0146e-01	1.4219e+00
TS (5 it.)	$u = 0$	2.9807e-01	3.4139e-01	1.4192e+00
TS (20 it.)	$u = 0$	2.6144e-01	3.3374e-01	1.4270e+00

Table 25: SV ($\omega = \omega^1$): errors on the computed image.

method	boundary condition	$\|\Delta I\|_1$	$\|\Delta I\|_2$	$\|\Delta I\|_\infty$
FS	$u = 0$	1.1020e-02	3.6455e-02	8.1754e-01
FS	$u = g_{\mathrm{SV}}$	1.5390e-02	7.7637e-02	9.1307e-01
DD	$u = 0$	7.4405e-02	1.1557e-01	9.8431e-01
DD	$u = g_{\mathrm{SV}}$	4.9549e-02	8.3209e-02	9.7255e-01
TS (5 it.)	$u = 0$	1.1799e-01	1.4993e-01	9.6863e-01
TS (20 it.)	$u = 0$	8.2467e-02	1.2607e-01	9.8039e-01

5 Conclusion

We finish the paper by drawing some conclusions based on the analyses and numerical tests presented in the previous sections.

As far as the mathematical properties of the three methods is concerned, we note that the FS and DD methods always converge to a solution of the SFS problem, whereas convergence is not guaranteed by the TS method. As can be seen by the tests in Section 4, it is not simply a question of style. In fact, increasing the number of iterations in TS often does not improve the solution in terms of any of our error indicators. However, it should be noted that the DD algorithm converges in general to a local minimum, and the search for a global minimum would require much more computational effort. The FS method converges to the maximal solution which, in general, will not coincide with the real surface.

The minimization method suggested by Daniel and Durou seems to produce accurate results for smooth surfaces, and the Dirichlet boundary conditions are easily included in the algorithm. The method put forward by Tsai and Shah gives its best results on smooth surfaces but suffers in some cases, particularly when the surface have several maxima and minima (as in Test 4). Moreover, it seems impossible to change the Dirichlet boundary condition in that method. Finally, Falcone and Sagona's method seems to be well adapted to a variety of different situations which include smooth and non-smooth surfaces and can cope with Dirichlet boundary conditions. However, its weak point seems to be CPU time and the absence of smooting in the surfaces which in some cases produced larger errors in the reconstruction of the normals.

The three methods appear to be stable in presence of a perturbation in the light source direction. They all compute a reasonable solution although FS seems to be more sensitive to this kind of perturbation.

The analyses contained in this paper could give rise to several interesting developments and improvements. The first improvement is related to the use of fast marching methods for the solution of the eikonal equation to reduce CPU time in the FS method. As far as the minimization methods are concerned, it would be desirable to reduce the regularization term in order to keep the corners of the shapes. For all the methods, it would be interesting to deal with "black shadows" (generated by an oblique light source) avoiding additional conditions on the boundaries separating light and shadow regions. We hope to return to these problems in our future research.

Acknowledgments

This work was partially supported by the MURST Funds "Scientific Computing" and by the TMR Network "Viscosity Solutions and Applications" (contract FMRX-CT98-0234). We thank Alain Crouzil for his help with the manuscript, which was given its final form by James Munnick of the *Centre de Traduction*, Université Paul Sabatier, Toulouse, France.

References

[1] M. Bardi and M. Falcone. An Approximation Scheme for the Minimum Time Function. *SIAM Journal of Control and Optimization*, 28(4):950–965, July 1990.

[2] M. Bardi and M. Falcone. Discrete Approximation of the Minimal Time Function for Systems with Regular Optimal Trajectories. In A. Bensoussan and P.-L. Lions, editors, *Analysis and Optimization of Systems*, volume 144 of *Lecture Notes in Control and Information Sciences*, pages 103–112. Springer-Verlag, 1990.

[3] G. Barles. *Solutions de viscosité des équations de Hamilton-Jacobi*. Springer-Verlag, 1994.

[4] A. Blake, A. Zisserman, and G. Knowles. Surface Descriptions from Stereo and Shading. *Image and Vision Computing*, 3(4):183–191, 1985.

[5] J.-F. Bonnans, J.-C. Gilbert, C. Lemaréchal, and C. A. Sagastizábal, editors. *Numerical Optimization: Theoretical and Practical Aspects*. Springer-Verlag, 2002.

[6] A. R. Bruss. The Eikonal Equation: Some Results Applicable to Computer Vision. *Journal of Mathematical Physics*, 23(5):890–896, May 1982.

[7] F. Camilli and M. Falcone. An Approximation Scheme for the Maximal Solution of the Shape-from-Shading Problem. In *Proceedings of the 3^{rd} International Conference on Image Processing*, volume I, pages 49–52, Lausanne, Switzerland, September 1996.

[8] F. Camilli and L. Grüne. Numerical Approximation of the Maximal Solutions for a Class of Degenerate Hamilton-Jacobi Equations. *SIAM Journal on Numerical Analysis*, 38(5):1540–1560, 2000.

[9] F. Camilli and A. Siconolfi. Discontinuous Solutions of a Hamilton-Jacobi Equation with Infinite Speed of Propagation. *SIAM Journal on Mathematical Analysis*, 28(6):1420–1445, 1997.

[10] F. Camilli and A. Siconolfi. Maximal Subsolutions of a Class of Degenerate Hamilton-Jacobi Problems. *Indiana University Mathematics Journal*, 48:1111–1131, 1999.

[11] M. G. Crandall, H. Ishii, and P.-L. Lions. User's Guide to Viscosity Solutions of Second Order Partial Differential Equations. *Bulletin of the American Mathematical Society*, 27(1):1–67, July 1992.

[12] M. G. Crandall and P.-L. Lions. Two Approximations of Solutions of Hamilton-Jacobi Equations. *Mathematics of Computation*, 43(167):1–19, 1984.

[13] A. Crouzil, X. Descombes, and J.-D. Durou. A Multiresolution Approach for Shape from Shading Coupling Deterministic and Stochastic Optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(11):1416–1421, November 2003.

[14] P. Daniel. *Peut-on extraire le relief d'une seule image ?* Thèse de doctorat, Université Paul Sabatier, Toulouse, France, January 2000. (in French).

[15] P. Daniel and J.-D. Durou. Creation of Real Images which are Valid for the Assumptions Made in Shape From Shading. In *Proceedings of the 10^{th} International Conference on Image Analysis and Processing*, pages 418–423, Venice, Italy, September 1999.

[16] P. Daniel and J.-D. Durou. From Deterministic to Stochastic Methods for Shape From Shading. In *Proceedings of the 4^{th} Asian Conference on Computer Vision*, pages 187–192, Taipei, Taiwan, January 2000.

[17] J.-D. Durou and H. Maître. On Convergence in the Methods of Strat and of Smith for Shape from Shading. *International Journal of Computer Vision*, 17(3):273–289, March 1996.

[18] J.-D. Durou and D. Piau. Ambiguous Shape from Shading with Critical Points. *Journal of Mathematical Imaging and Vision*, 12(2):99–108, April 2000.

[19] M. Falcone, T. Giorgi, and P. Loreti. Level Sets of Viscosity Solutions: Some Applications to Fronts and Rendez-Vous Problems. *SIAM Journal of Applied Mathematics*, 54(5):1335–1354, October 1994.

[20] M. Falcone and M. Sagona. An Algorithm for the Global Solution of the Shape-from-Shading Model. In *Proceedings of the 9^{th} International Conference on Image Analysis and Processing*, volume I, pages 596–603, Florence, Italy, September 1997.

[21] R. T. Frankot and R. Chellappa. A Method for Enforcing Integrability in Shape from Shading Algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(4):439–451, July 1988.

[22] H. B. Griffiths. *Surfaces*. Cambridge University Press, 1981.

[23] H. Hayakawa, S. Nishida, Y. Wada, and M. Kawato. A Computational Model for Shape Estimation by Integration of Shading and Edge Information. *Neural Networks*, 7(8):1193–1209, October 1994.

[24] B. K. P. Horn. Obtaining Shape from Shading Information. In P. H. Winston, editor, *The Psychology of Computer Vision*, chapter 4, pages 115–155. McGraw-Hill, 1975.

[25] B. K. P. Horn. Height and Gradient from Shading. *International Journal of Computer Vision*, 5(1):37–75, August 1990.

[26] B. K. P. Horn and M. J. Brooks. The Variational Approach to Shape From Shading. *Computer Vision, Graphics, and Image Processing*, 33(2):174–208, February 1986.

[27] B. K. P. Horn and M. J. Brooks, editors. *Shape from Shading*. MIT Press, 1989.

[28] B. K. P. Horn, R. Szeliski, and A. L. Yuille. Impossible Shaded Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(2):166–170, February 1993.

[29] K. Ikeuchi and B. K. P. Horn. Numerical Shape from Shading and Occluding Boundaries. *Artificial Intelligence*, 17(1–3):141–184, August 1981.

[30] H. Ishii and M. Ramaswamy. Uniqueness Results for a Class of Hamilton-Jacobi Equations with Singular Coefficients. *Communications in Partial Differential Equations*, 20:2187–2213, 1995.

[31] R. Kimmel and A. M. Bruckstein. Tracking Level Sets by Level Sets: A Method for Solving the Shape from Shading Problem. Center for Intelligent System Report CIS9319, Technion-Israel Institute of Technology, Haifa, Israel, May 1993.

[32] R. Kimmel and A. M. Bruckstein. Global Shape from Shading. In *Proceedings of the 12th International Conference on Pattern Recognition*, volume I, pages 120–125, Jerusalem, Israel, October 1994.

[33] R. Kimmel and J. A. Sethian. Optimal Algorithm for Shape from Shading and Path Planning. *Journal of Mathematical Imaging and Vision*, 14(3):237–244, May 2001.

[34] R. Kozera. A Note on Existence and Uniqueness in Shape from Shading. In *Proceedings of the 4th IEEE International Conference on Computer Vision*, pages 507–511, Berlin, Germany, May 1993.

[35] R. Kozera and R. Klette. Finite Difference Based Algorithms in Linear Shape from Shading. *Machine Graphics and Vision*, 6(2):157–201, 1997.

[36] Y. G. Leclerc and A. F. Bobick. The Direct Computation of Height from Shading. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 552–558, Maui, Hawaii, USA, June 1991.

[37] D. Lee. A Provably Convergent Algorithm for Shape from Shading. In *Proceedings of the Image Understanding Workshop*, pages 489–496, Miami Beach, Florida, USA, December 1985.

[38] K. M. Lee and C.-C. J. Kuo. Shape from Shading with a Linear Triangular Element Surface Model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(8):815–822, August 1993.

[39] P.-L. Lions. *Generalized Solutions of Hamilton-Jacobi Equations*. Pitman, 1982.

[40] P.-L. Lions, E. Rouy, and A. Tourin. Shape-from-Shading, Viscosity Solutions and Edges. *Numerische Mathematik*, 64:323–353, 1993.

[41] J. Malik and D. Maydan. Recovering Three-Dimensional Shape from a Single Image of Curved Objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(6):555–566, June 1989.

[42] J. Oliensis. Uniqueness in Shape from Shading. *International Journal of Computer Vision*, 6(2):75–104, June 1991.

[43] J. Oliensis and P. Dupuis. A Global Algorithm for Shape from Shading. In *Proceedings of the 4th IEEE International Conference on Computer Vision*, pages 692–701, Berlin, Germany, May 1993.

[44] S. J. Osher and J. A. Sethian. Fronts Propagating with Curvature-Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations. *Journal of Computational Physics*, 79(1):12–49, 1988.

[45] A. P. Pentland. Local Shading Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(2):170–187, March 1984.

[46] A. P. Pentland. Linear Shape from Shading. *International Journal of Computer Vision*, 4(2):153–162, March 1990.

[47] E. Prados, O. Faugeras, and E. Rouy. Shape from Shading and Viscosity Solutions. In *Proceedings of the 7th European Conference on Computer Vision*, volume II, pages 790–804, Copenhagen, Denmark, May 2002.

[48] T. Rindfleisch. Photometric Method for Lunar Topography. *Photometric Engineering*, 32(2):262–277, March 1966.

[49] E. Rouy and A. Tourin. A Viscosity Solutions Approach to Shape from Shading. *SIAM Journal on Numerical Analysis*, 29(3):867–884, June 1992.

[50] M. Sagona. *Numerical methods for degenerate Eikonal type equations and applications*. Tesi di dottorato, Dipartimento di Matematica dell'Università di Napoli "Federico II", Napoli, Italy, November 2001.

[51] J. A. Sethian and A. Vladimirsky. Fast Methods for the Eikonal and Related Hamilton-Jacobi Equations on Unstructured Meshes. *Proceedings of the National Academy of Sciences of the USA*, 97(11):5699–5703, 2000.

[52] T. M. Strat. A Numerical Method for Shape from Shading from a Single Image. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, 1979.

[53] R. Szeliski. Fast Shape from Shading. *Computer Vision, Graphics, and Image Processing: Image Understanding*, 53(2):129–153, March 1991.

[54] P.-S. Tsai and M. Shah. Shape from Shading Using Linear Approximation. *Image and Vision Computing*, 12(8):487–498, October 1994.

[55] G. Ulich. Provably Convergent Methods for the Linear and Nonlinear Shape from Shading Problem. *Journal of Mathematical Imaging and Vision*, 9(1):69–82, July 1998.

[56] J. Van Diggelen. A Photometric Investigation of the Slopes and Heights of the Ranges of Hills in the Maria of the Moon. *Bulletin of the Astronomical Institute of the Netherlands*, 11(423):283–290, July 1951.

[57] R. L. Wildey. Radarclinometry for the Venus Radar Mapper. *Photogrammetric Engineering and Remote Sensing*, 52(1):41–50, January 1986.

[58] P. L. Worthington and E. R. Hancock. New Constraints on Data-Closeness and Needle Map Consistency for Shape-from-Shading. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(12):1250–1267, December 1999.

[59] Z. Wu and L. Li. A Line-Integration Based Method for Depth Recovery from Surface Normals. *Computer Vision, Graphics, and Image Processing*, 43(1):53–66, July 1988.

[60] R. Zhang, P.-S. Tsai, J. E. Cryer, and M. Shah. Shape from Shading: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(8):690–706, August 1999.