KYBERNETIKA — VOLUME *46* (2010), NUMBER 2, PAGES 207–225

NUMERICAL ALGORITHMS FOR PERSPECTIVE SHAPE FROM SHADING

Michael Breuss, Emiliano Cristiani, Jean-Denis Durou, Maurizio Falcone and Oliver Vogel

The Shape-From-Shading (SFS) problem is a fundamental and classic problem in computer vision. It amounts to compute the 3-D depth of objects in a single given 2-D image. This is done by exploiting information about the illumination and the image brightness. We deal with a recent model for Perspective SFS (PSFS) for Lambertian surfaces. It is defined by a Hamilton–Jacobi equation and complemented by state constraints boundary conditions. In this paper we investigate and compare three state-of-the-art numerical approaches. We begin with a presentation of the methods. Then we discuss the use of some acceleration techniques, including cascading multigrid, for all the tested algorithms. The main goal of our paper is to analyze and compare recent solvers for the PSFS problem proposed in the literature.

Keywords: hyperbolic partial differential equation, Hamilton–Jacobi equation, finite difference method, semi-Lagrangian scheme, Shape-from-Shading

Classification: 35L60, 65N06, 65N12, 68U10

1. INTRODUCTION

The Shape-From-Shading (SFS) problem consists in the reconstruction of the 3-D depth of depicted objects of a single given grey value image. Thereby, SFS makes use of the image brightness as well as of information about the direction of the light source. It is a classic inverse problem in computer vision with many potential applications; see e. g. [8, 13, 14, 24] and references therein.

We deal with a modern model for SFS which includes perspective deformations and that can be cast into the format of a partial differential equation (PDE), see [18]. Three different methods have been proposed in the recent literature to tackle the corresponding problem, see [7, 18, 23]. The question arises, which of these methods one should use for future developments in this direction of research in SFS. In this paper, we are addressing this question, comparing and evaluating the three mentioned schemes. In contrast to the basic original versions of the algorithms, we consider various numerical acceleration techniques that are relatively easy to implement and of practical relevance for computer vision applications. Doing this, we extend some of the methods used before in this field. Our extensive numerical experiments show that an extended version of the algorithm proposed in [23] gives

the best performance; however, also for the other algorithms a significant efficiency gain can be achieved.

Perspective Shape-From-Shading. Two key issues in mathematical models of SFS are the surface reflectance and the camera model. We will employ the classic assumption of a *Lambertian surface reflectance* [10]: the light intensity at some point M on an object surface perceived by the observer linearly depends on the cosine of the angle between the light source direction ω and the normal to the surface at M. Compared to other possible approaches, this surface model is relatively easy to access for modelling purposes and theoretical analysis. The *camera model* is concerned with the projection performed when mapping the 3-D real world to 2-D images. In early SFS models, this projection is assumed to be orthographic. Concerning this type of models, let us mention the pioneering work of Horn [12] who was also the first who modelled SFS via a PDE. However, orthographic models were in practice not too successful, as shown in two recent survey papers [8, 24]. As an attempt to improve SFS results, the orthographic camera model has been substituted by employing a more realistic perspective projection [4, 16, 22].

In this paper, we use the perspective approach (PSFS). We also consider a point light source located at the optical center as within some of the mentioned works concerned with PSFS, and we use the so-called *light attenuation term*. This model for PSFS was shown to be well-posed in [18, 19] under some assumptions which include the differentiability of the surface (see also [3] where discontinuous brightness functions have been considered).

Mathematical formulation of PSFS. The PDE that arises is a static, hyperbolic *Hamilton–Jacobi (HJ) equation*. Making use of the Legendre transform, one can formulate it equivalently as a *Hamilton–Jacobi–Bellman (HJB) equation* and solve the corresponding optimal control problem. In both cases, the equations are complemented by state constraint boundary conditions.

Algorithms. The first important developments for the PSFS model with light attenuation are based on the control-theoretic formulation. In [18] and related works of Prados and his co-workers, the dynamic programming principle is used in the form of a top-down process, leading to an iterative algorithm. The pointwise arising optimal control problem is solved analytically in this approach. Then, in [7], a semi-Lagrangian method was developed also based on the HJB equation. In this method, the domain of the optimal control is discretised. An artificial iteration variable introduced into the HJB equation gives here a recursive method. The third numerical approach of importance was proposed in [23], where it was suggested to use the HJ equation as a basis of the discretisation without refering to the optimal control problem. Using an artificial time variable, also this approach is iterative. In our paper, we investigate these methods and their algorithmic extensions obtained by acceleration techniques.

Related work. This paper complements and extends previous conference papers of some of the authors [2, 7, 23]. In [7, 23] basic versions of two of the investigated algorithms were introduced. In a later conference contribution [2], the three algorithms discussed in the current paper were compared and extended to some degree. However, no experimental convergence analysis was given. The emphasis in that work was on the influence of parameters like the stopping condition for the iterative schemes. Also, a cascading multigrid (CM) method was only investigated for the direct method from [23] there. The reason for this was that the multigrid approach seems to be not popular for solving hyperbolic first-order HJB equations; the papers one finds in the literature are mainly based on [11] dealing with multigrid schemes for second-order elliptic HJB equations.

Our contribution. In this paper we build upon the conference contribution [2]. We use here the efficient algorithmic variants that were identified for all three approaches in that work, namely with Gauß–Seidel–type updating and fast sweeping. For these modifications, we briefly compile the most important results from [2]. Based on the modified schemes, we extend the conference paper here in the following directions:

(i) We study the numerical convergence properties of the schemes. This investigation is particularly interesting since the algorithms differ very much in the numbers of iterations needed for convergence as well as in the computational effort per iteration.

(ii) We analyse experimentally the convergence properties of the CM acceleration method for all the schemes. This investigation gives some surprising results. We show that a generic parameter setting for the CM method does not work well with the optimal control approaches.

(iii) Based on the results of the first two points, we refine the CM approach by identifying a good choice for the stopping condition employed on the coarse grid levels.

With this systematic study of algorithms for the PSFS model we complement recent investigations of schemes for other SFS models performed in [8]. Furthermore, we conjecture that our paper can be an important landmark for works on numerical schemes for PSFS, as it summarizes and extends the recent efforts of several research groups.

Paper organisation. In Section 2, we briefly review the PSFS model with light attenuation. The three numerical approaches of interest are introduced in Section 3, where we also discuss the acceleration techniques used in our tests. In Section 4, we are concerned with the comparative study of numerical methods and in Section 5 we perform some additional experiments about the speed of convergence of the methods and the effect of the acceleration techniques. The last section contains our conclusions.

2. THE MATHEMATICAL MODEL

In this section we recall the model for PSFS introduced in [15] with light attenuation term.

Let $(x, y) \in \mathbb{R}^2$ be in the image domain Ω, where Ω is an open set. Furthermore:

- $I = I(x, y)$ is the normalised brightness function. We have $I = \frac{E(x,y)}{\sigma}$, where E is the greylevel of the given image and σ is the product of the surface albedo (which tells us to which extent the surface reflects light) by the light source intensity.
- f is the focal length, i. e. the distance between the optical center C of the camera and the 2-D plane to which the scene of interest is mapped.

Let M be a generic point on the surface Σ. The unknown of the problem is the function $u : \Omega \to \mathbb{R}$ such that

$$M = M(x, y) = u(x, y)\, m' \tag{1}$$

where

$$m' = \frac{\mathsf{f}}{\sqrt{x^2 + y^2 + \mathsf{f}^2}}\, m \quad \text{and} \quad m = (x, y, -\mathsf{f})\,, \tag{2}$$

see Figure 1. Note that, according to these notations, $u > 0$ holds as the depicted scene is in front of the camera.

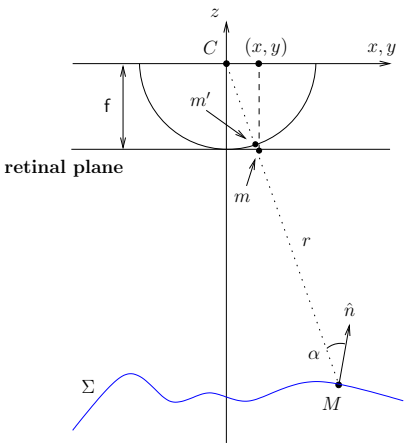


Fig. 1. The perspective SFS model with a point light source at the optical center.

We denote by $r(x, y)$ the distance between the point light source and the point $M(x, y)$ on the surface. It holds $u(x, y) = r(x, y)/\mathsf{f}$, since the light source location coincides with the optical center.

The PDE associated to this PSFS model is obtained writing down the *image irradiance equation*:

$$R(\hat{n}(x,y)) = I(x,y), \tag{3}$$

making explicit the normal $\hat{n}$ to the surface and the reflectance function R which gives the value of the light reflection on the surface as a function of its normal.

We denote by $\omega(x,y)$ the unit vector representing the light source direction at the point $M(x,y)$ (note that in the classic SFS problem this direction is constant),

$$\omega(x,y) = \frac{(-x,-y,\mathsf{f})}{\sqrt{x^2+y^2+\mathsf{f}^2}}. \tag{4}$$

Taking into account additionally the assumption of a *Lambertian surface*, the function R is defined as

$$R(\hat{n}(x,y)) = \frac{\omega(x,y)\cdot\hat{n}(x,y)}{r(x,y)^2}. \tag{5}$$

In order to write down the corresponding PDE, it is useful to introduce the new unknown $v = \ln(u)$ (remember that $u > 0$). Equation (5) can be written as a *static HJ equation*:

$$\frac{1}{Q(x,y)} I(x,y)\,\mathsf{f}^2\,W(x,y) - e^{-2v(x,y)} = 0 \tag{6}$$

where

$$W(x,y,\nabla v) := \sqrt{\mathsf{f}^2|\nabla v(x,y)|^2 + (\nabla v(x,y)\cdot(x,y))^2 + Q(x,y)^2} \tag{7}$$

and

$$Q(x,y) := \frac{\mathsf{f}}{\sqrt{x^2+y^2+\mathsf{f}^2}}. \tag{8}$$

The same equation admits also a 'control formulation'. Indeed, it is proved in [17, 18, 19] that v is defined as the solution of the following *HJB equation*:

$$-e^{-2v(x,y)} + \sup_{a\in B(0,1)} \{-b(x,y,a)\cdot\nabla v(x,y) - \ell(x,y,a)\} = 0 \tag{9}$$

where $B(0,1)$ denotes the closed unit ball in $\mathbb{R}^2$ and the other terms in (9) are defined as follows:

$$b(x,y,a) := -JG^T DGa\,, \quad \ell(x,y,a) := -I(x,y)\,\mathsf{f}^2\sqrt{1-\|a\|^2}\,, \tag{10}$$

$$J(x,y) := I(x,y)\mathsf{f}\sqrt{\mathsf{f}^2+x^2+y^2} \tag{11}$$

where $\|\cdot\|$ denotes the Euclidean vector norm, and G and D are the 2×2 matrices:

$$G(x,y) := \frac{1}{\sqrt{x^2+y^2}}\begin{pmatrix} y & -x \\ x & y \end{pmatrix}, \quad D(x,y) := \begin{pmatrix} \mathsf{f} & 0 \\ 0 & \sqrt{\mathsf{f}^2+x^2+y^2} \end{pmatrix}. \tag{12}$$

3. ALGORITHMS

We will proceed here as follows. First, we describe the set-ups of all the three methods. Then, we proceed by giving relevant details of the discretisations and the speed-up mechanisms we generally use for all the schemes. In the last parts of this paragraph, we describe the CM acceleration method we investigate in detail in this work, and we comment on the boundary condition.

In order to distinguish between the different algorithms, we will use from now on the following abbreviations:

- PF denotes the numerical scheme introduced by Prados and Faugeras [16] based on the optimal control approach using dynamic programming, complemented with an upwind finite difference discretisation.
- CFS denotes the numerical scheme introduced by Cristiani, Falcone and Seghini [7]. It is based on the same approach as PF but it uses a semi-Lagrangian discretisation (see [9] for details) instead of finite differences. We also refer to [6] for the approximation of a different PSFS model.
- VBW denotes the direct Hamilton–Jacobi–based method of Vogel, Breuß and Weickert [23].

Upwind-type discretisation of spatial derivatives. In the PF and VBW methods, the discretization of spatial derivatives is made by using the stable upwind-type discretisation of Rouy and Tourin [21]. This is as follows. Let h_x and h_y be the spatial mesh widths in x- and y-direction, respectively. Denoting then by $v_{i,j}$ the value of v at the mesh point $(ih_x, jh_y)^T$, the approximate upwind differences of [21] read as

$$\widetilde{\nabla}_x v(ih_x, jh_y) \approx \min\left(0, \frac{v_{i+1,j} - v_{i,j}}{h_x}, \frac{v_{i-1,j} - v_{i,j}}{h_x}\right), \tag{13}$$

$$\widetilde{\nabla}_y v(ih_x, jh_y) \approx \min\left(0, \frac{v_{i,j+1} - v_{i,j}}{h_y}, \frac{v_{i,j-1} - v_{i,j}}{h_y}\right). \tag{14}$$

In the case of the VBW scheme, if the third argument in (13) or (14) is chosen, one needs to set then $\widetilde{\nabla}_x v(ih_x, jh_y) := (v_{i,j} - v_{i-1,j})/h_x$ or $\widetilde{\nabla}_y v(ih_x, jh_y) := (v_{i,j} - v_{i,j-1})/h_y$, respectively, in order to ensure the consistency of the approximation.

In (13) – (14) neither iteration nor time levels of the values of v are specified yet. The reason for this missing specification is that we use a Gauß-Seidel-type updating and a sweeping technique in order to accelerate convergence. This combination of techniques leads to a different choice of data labels for each sweeping direction.

Let us remark that in the original versions of the PF, the CFS and the VBW schemes, acceleration techniques were barely considered: only the PF scheme is originally used together with a Gauß-Seidel-type updating strategy, see [18].

3.1. The PF scheme

The PF scheme proposed by Prados et al. [16, 17, 18, 19] is based on the HJB formulation, see (9) – (12). For the numerical solution of the optimal control problem, one has

(i) to discretise the occurring partial derivatives of v,

(ii) to discretise the source term e^{-2v}, and

(iii) to find an optimal control $a \in B(0,1)$.

We address these points in this section. Having established the discrete formulation, the equation is solved pointwise in an iterative way.

(i) In the PF scheme the first-order derivatives part of ∇v are discretised by using appropriate upwind discretisations as described in [21]. These are summarized in Section 3.4 below.

(ii) The PF method is practically a semi-implicit scheme where the implicitness stems from the corresponding treatment of the source term. Newton's method is used for every fixed point iteration.

(iii) In order to solve the maximization problem in (9) we have to search for an optimal a in the entire unit ball. This is done in the PF scheme by computing the analytical solution for a over $B(0,1)$, which is a quite complicated procedure, as one needs to take into account the case distinctions w.r.t. the occurring upwind directions.

3.2. The CFS scheme

The CFS scheme developed in [7] is also an optimal control technique. The HJB equation is also solved by means of a specific iterative fixed point procedure. Although the resulting method is close to the PF scheme, the scheme is based on semi-Lagrangian approximation of the directional derivatives, which automatically gives the upwind correction and results to be easier to implement than PF.

Introducing an artificial time step τ, and an artificial time-dependency into the process (indicated by a lower index τ), one may obtain from (9) the semi-discrete scheme

$$-v_\tau(x,y) + \min_{a\in B(0,1)} \left\{ v_\tau\big((x,y) + \tau b(x,y,a)\big) + \tau \ell(x,y,a) \right\} + \tau e^{-2v_\tau(x,y)} = 0\,. \quad (15)$$

This equation can be solved iteratively by employing a sequence $v_\tau^{(k)}$, $k = 0, 1, \ldots$, marching in pseudo-time to infinity until a steady state of v_τ is reached. To make the scheme fully-discrete, a grid $\{(x_i, y_j)\}_{i,j=1,\ldots,N}$ is introduced and equation (15) is considered only at the grid nodes (x_i, y_j). Moreover, since the term $(x_i, y_j) + \tau b(x_i, y_j, a)$ is not in general sitting on the grid, the value of v_τ at that point is computed by linear interpolation using the three nearest grid nodes (note that other more accurate but more expensive choices are possible for the interpolation [9]).

Also the CFS algorithm is semi-implicit and employs Newton's method for resolving the source term.

Again the optimal control a has to be sought within the entire unit ball in $\mathbb{R}^2$. This is obtained in the CFS scheme via a sampling procedure, making use of 8 directions with 3 points in each direction additionally to the origin.

3.3. The VBW scheme

In the VBW scheme introduced in [23], the Hamilton–Jacobi equation (6)–(8) is used directly, without resorting to the optimal control approach.

Also the VBW scheme relies on the use of the upwind discretisation of Rouy and Tourin [21], see the next paragraph.

The means to obtain an iterative scheme with this approach is to augment the depth variable $v(x, y)$ with an artificial time variable t, i. e. $v = v(x, y, t)$. One then solves instead of (6) the time-dependent PDE

$$\frac{\partial}{\partial t} v(x, y, t) + \frac{I(x, y)}{Q(x, y)} \mathsf{f}^2 W(x, y, \nabla v) \;=\; e^{-2v(x,y,t)} \tag{16}$$

for the steady state defined by $\frac{\partial}{\partial t} v = 0$, thus retrieving (6) for $t \to \infty$. Denoting by n the time iteration, the scheme then reads

$$v_{ij}^{n+1} = v_{ij}^n + \tau \left(-\frac{I_{ij}}{Q_{ij}} \mathsf{f}^2 \sqrt{\mathsf{f}^2 |\widetilde{\nabla} v_{ij}^n|^2 + (\widetilde{\nabla} v_{ij}^n \cdot (x_i, y_j))^2 + Q_{ij}^2} + e^{-2v_{ij}^n} \right).$$

In the paper [23] the source term e^{-2v} is treated implicitly, using Newton's method in each point and time step to solve the corresponding fixed point equations. In a later work [2] an explicit discretisation of the source term is used, a variant we also investigate here.

However, due to the direct Hamilton–Jacobi approach no optimal control needs to be determined.

3.4. Acceleration techniques

We discuss here the acceleration techniques for the three methods.

The Gauß–Seidel–type updating. The idea behind this acceleration technique is the same as with the Gauß–Seidel method for solving linear systems of equations. It is based on the observation that at pixel (i, j) the data from the pixels (i, j), $(i \pm 1, j)$ and $(i, j \pm 1)$ contribute in the upwind formulae. Assume for instance that we iterate from left to right and from top to bottom over the mesh points. Thus, ascending in i and descending in j, we incorporate the available computed values into the computation of derivatives from (13)–(14).

Using the iteration levels n and $n + 1$ together with a time step size τ, this

procedure gives the formulae

$$\widetilde{\nabla}_x v(ih_x, jh_y, n\tau) \approx \min\left(0, \frac{v^n_{i+1,j} - v^n_{i,j}}{h_x}, \frac{v^{n+1}_{i-1,j} - v^n_{i,j}}{h_x}\right), \tag{17}$$

$$\widetilde{\nabla}_y v(ih_x, jh_y, n\tau) \approx \min\left(0, \frac{v^{n+1}_{i,j+1} - v^n_{i,j}}{h_y}, \frac{v^n_{i,j-1} - v^n_{i,j}}{h_y}\right). \tag{18}$$

Note again the proceeding for the VBW scheme concerning the third arguments, cf. the comment after (13)–(14). Let us emphasize that the values $v^{n+1}_{i,j+1}$ and $v^{n+1}_{i-1,j}$ in (17)–(18) were already computed via the described method, so that the Gauß–Seidel idea can be applied.

Sweeping. We also employ a sweeping technique, see [25]. The motivation of the sweeping method can be given as follows on a completely discrete level.

Let us consider the situation when one iterates in only one direction over the points, e. g. always ascending in i and descending in j as described above. By application of the Gauß–Seidel idea, one uses the values $v^{n+1}_{i-1,j}$ from the left and $v^{n+1}_{i,j+1}$ from above in order to accelerate convergence. This means that information is quickly propagated from top-left to bottom-right, but this is a one-sided advantage.

As a remedy, it is obvious to proceed by iterating over the grid in a cyclic way:

1. Left → Right, and Top → Bottom
2. Top → Bottom, and Right → Left
3. Right → Left, and Bottom → Top
4. Bottom → Top, and Left → Right

This procedure is called *sweeping* in the literature. As is easily seen, depending on the actual sweeping direction within the above cycle, different values $v^{n+1}_{i\pm1,j\pm1}$ have to be taken into account in (17)–(18).

The Cascading Multigrid Method. In addition to the described improvements, we employ a cascading multigrid (CM) method, see [1].

The CM routine is a relatively easy-to-use algorithm. Practically, it is a coarse-to-fine strategy, where we start from a coarse level and iterate up to the finest level identical with the original image domain. Thereby, the refinement is always implemented by doubling the number of grid points in each direction, involving linear interpolation from known values to the newly inserted nodes. Of course, this implies that the original image must be given in a size identical to a power of two. In our experiments, we always start at the coarsest level with 2×2 pixels.

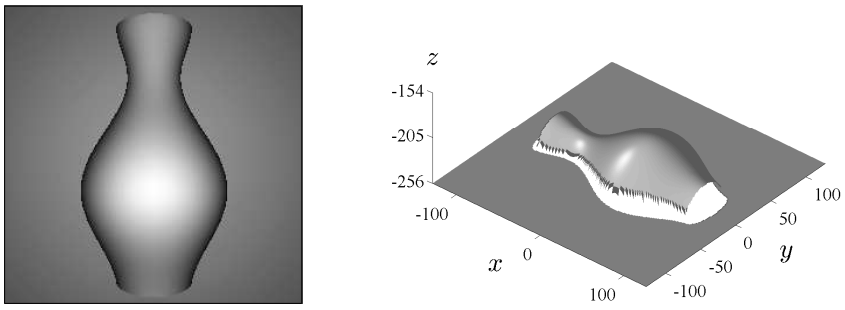
The CM method is closely related to other well-known strategies in image processing like e. g. the Gaussian image pyramids, cf. [20]. Thus, it is a good candidate for a numerical acceleration technique for applications in the field we consider.

3.5. Boundary Conditions

An important issue is always the definition of correct boundary conditions and their numerical implementation. In the PSFS model the presence of the attenuation term $1/r^2$ makes suitable to use state constraint boundary conditions which can be easily implemented in the form of homogeneous Dirichlet boundary conditions $u(x, y) = \overline{u}$ where $\overline{u} = \max_{(x,y)\in\Omega} u(x, y)$ (this is our choice for all tests of the paper). In fact, in the context of the upwind differencing employed within the considered schemes, the correct, so-called state constraints b.c. are satisfied automatically because of the effect of the minimization procedure within the upwinding formulae. Other choices are also possible: in particular situations the solution is known at the boundary, then exact Dirichlet boundary conditions are the best choice. Alternatively, homogeneous Neumann boundary conditions can be used whenever this information is not available.

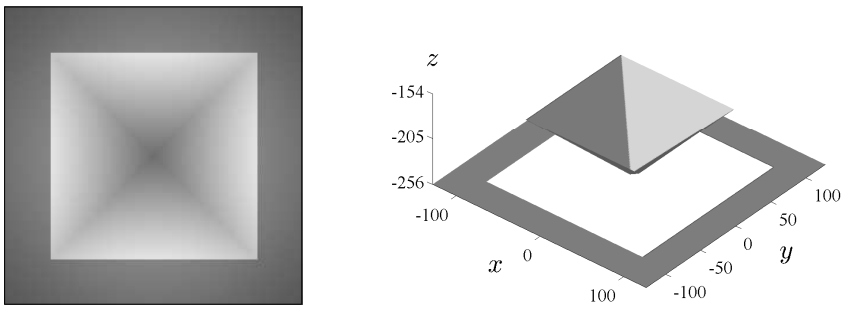
4. COMPARISONS OF THE ALGORITHMS

For comparisons we use synthetic images, so that all the parameters for the camera, the illumination and the true depth are known. The test surfaces are *(i)* a newly rendered version of a 'classic' benchmark in SFS [8, 24], namely a *Synthetic Vase* on a flat background and *(ii)* an *Upside-down Pyramid* on a flat background already used in [5, 7] to compare the PSFS model we use here with the one where the light source is assumed to be at infinite distance [4]. See Figures 2 – 3 for the corresponding input images (256×256 pixels) and the surface we expect to reconstruct. Note that the white part of the background in the 3-D surfaces corresponds to the region not visible from the optical center (because it is hidden by the vase or by the pyramid itself). In the top and bottom side of the vase the effect of the perspective deformation is clearly visible, this is due to the fact that the camera is not at infinite distance from the object.



(a) (b)

Fig. 2. Synthetic vase. (a) Input image, 256×256 pixels. It is rendered using the parameters $\mathsf{f} = 256$ and $\sigma = 165.10$. (b) Ground truth.



(a) (b)

Fig. 3. Synthetic upside-down pyramid. (a) Input image, 256×256 pixels. It is rendered using the parameters $\mathsf{f} = 256$ and $\sigma = 155.74$. (b) Ground truth.

For initialising the iterative process for all algorithms we solve the optimal control problem analytically for the null control $a = (0, 0)^T$.

In all scheme variants we use the Gauß–Seidel idea as well as Sweeping. The stopping criterion is satisfied if the difference of two successive iterates is less than 10^{-4} in the maximum-norm. This choice is reasonable for PSFS computations as identified in [2].

In our numerical tests, all the algorithms return qualitatively the same result, so we report just one outcome. In Figure 3 we show the result for the synthetic vase in two cases: 1. the computation is performed only on the vase, 2. the computation is performed in the whole square domain (including the background). We show the function $-u$ (the reversed solution of the PSFS equation), as well as the reconstructed surface computed by means of (1).

As it can be seen, the reconstruction of the vase itself is quite good, while the background is estimated as a continuation of the vase boundary, leading to a large error on the top and the bottom of the vase. In Figure 4 we show the results for the upside-down pyramid (with and without background). As in the previous test, the pyramid itself is approximated quite well, and the background is again estimated as a continuation of the pyramid boundary.

The computational times we present were obtained using an implementation in C on a standard PC (Linux, Pentium IV, 3.2 GHz, 2 GB RAM). Numerical errors (in L^1 and L^∞ norms) are computed comparing the approximate function u (see Eq. (1)) with its exact value rather than comparing the exact and approximate final surface. This is done because the surface is given in parametric form and it is not defined on a regular grid. Errors are given in terms of the *relative depth error*, i. e. in percentages of the true depth, as this is meaningful in the context of the SFS task. We also report the number of iterations needed. Note that one iteration consists of four sweeps. The use of the cascading multigrid algorithm is indicated by 'CM'. In this context, the number of iterations only represents the iterations on

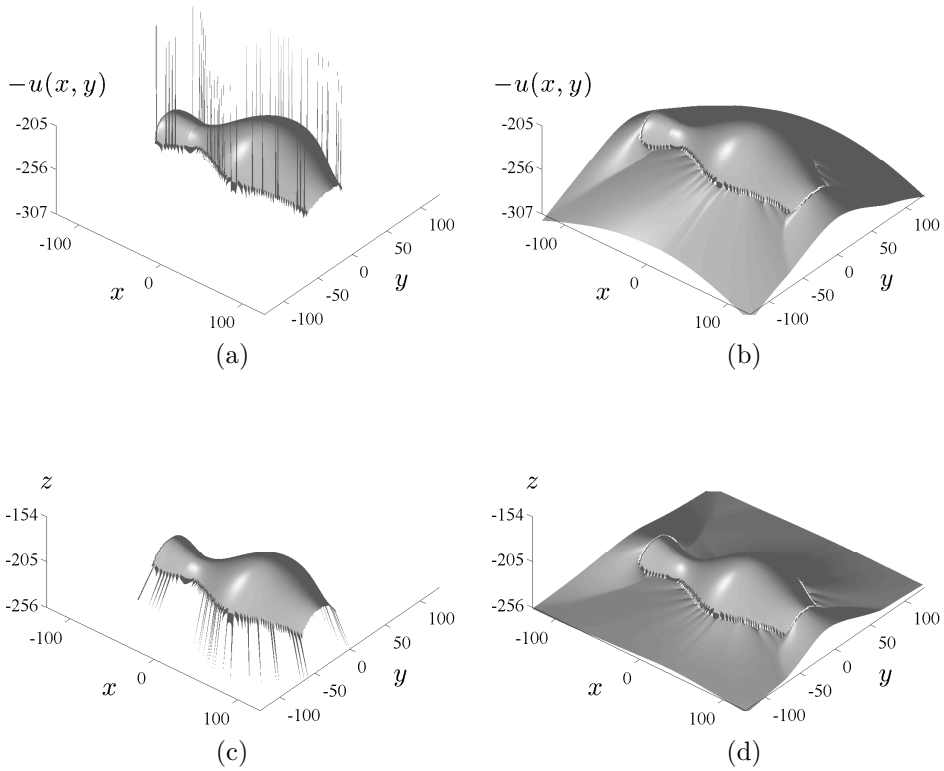


Fig. 4. Results of the numerical tests for the synthetic vase.
First row: function $-u$ (without and with background).
Second row: reconstructed surface (without and with background).

the finest grid. The results of our comparison (vase and pyramid, with and without background) are summarized in Tables 1 – 4.

Table 1. Schemes comparison for the vase experiment, without background.

Algorithm	CM	L^1 error [%]	L^∞ error [%]	Time [s]	Iterations
PF	no	0.06	0.21	16.9	27
CFS	no	0.47	1.71	9.7	19
VBW	no	0.17	3.04	3.2	62

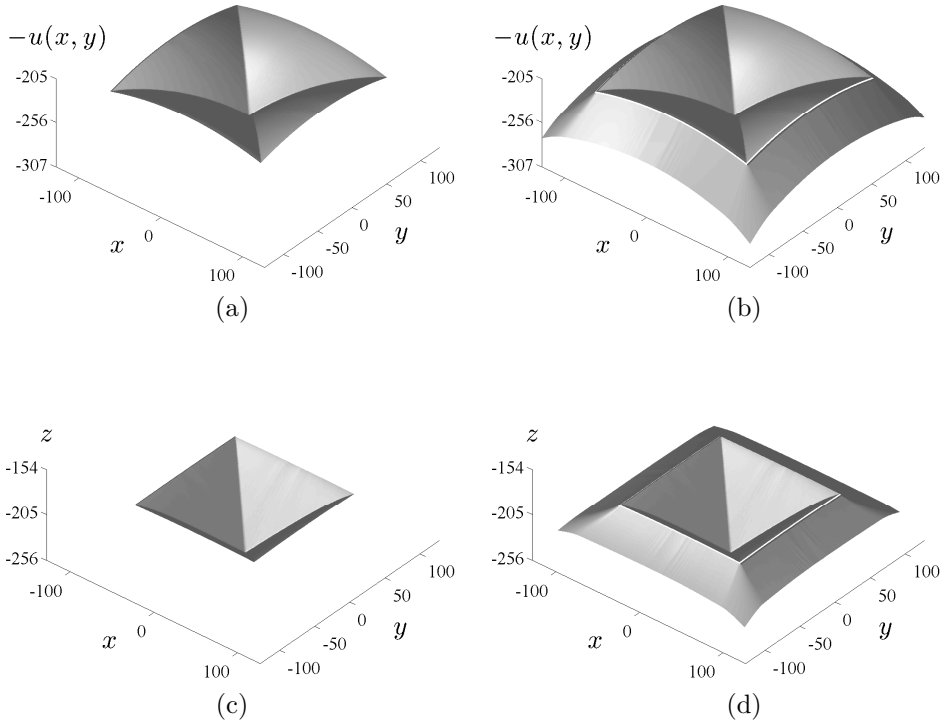


Fig. 5. Results of the numerical tests for the upside-down pyramid.
First row: function $-u$ (without and with background).
Second row: reconstructed surface (without and with background).

Table 2. Schemes comparison for the vase experiment, with background.

Algorithm	CM	L^1 error [%]	L^∞ error [%]	Time [s]	Iterations
PF	no	1.10	10.42	125.5	38
CFS	no	1.17	10.25	70.2	26
VBW	no	1.98	10.41	20.4	100
VBW	yes	1.90	10.04	8.1	29

Table 3. Schemes comparison for the pyramid experiment, without background.

Algorithm	CM	L^1 error [%]	L^∞ error [%]	Time [s]	Iterations
PF	no	2.95	5.16	46.1	23
CFS	no	3.20	5.55	27.5	17
VBW	no	3.01	5.47	6.9	63

Table 4. Schemes comparison for the pyramid experiment, with background.

Algorithm	CM	L^1 error [%]	L^∞ error [%]	Time [s]	Iterations
PF	no	10.06	22.52	90.8	24
CFS	no	9.49	22.40	55.4	18
VBW	no	10.01	22.48	12.6	63
VBW	yes	10.19	21.51	8.2	28

As observable, the results for the images without background (continuous surfaces) are very accurate for all methods, indicating that the model has reached some level of maturity. On the contrary, results for images with background are largely inaccurate, this is due to the inability to catch the discontinuities of the surface. As a side note, the use of the CM algorithm has a negligible influence on the accuracy, cf. Tables 2 and 4. Generally speaking, minor differences between scheme variants with and without CM depend on the stopping accuracy and the example.
VBW is the fastest method, even without the CM correction which gives an important additional speed-up. PF is the most accurate method for the continuous case, but it is the slowest one. Its accuracy is probably due to the fact that the optimal control is computed analytically. Finally, we note that the differences between the errors are not so relevant as those for CPU times.

5. EVALUATION OF ACCELERATION TECHNIQUES

In the previous experiments we used a fast sweeping technique for all the methods. Now, we will analyse how big the impact of this technique is on the computation times of the different methods. In addition, since we discretised the source term in the VBW explicitly, we will investigate the effect of discretising it implicitly as done in the other methods. We will do this on the upside-down pyramid test image (with background).

Table 5. Analysis of the effect of fast sweeping and implicit discretisation of the source term on the computation time for the pyramid image.

Method	Sweeping (y/n)	explicit/implicit	Time [s]	Iterations
PF	n	i	139.71	147
CFS	n	i	91.4	123
VBW	n	e	14.8	282
VBW	n	i	31.8	256
PF	y	i	90.8	24
CFS	y	i	55.4	18
VBW	y	e	12.6	63
VBW	y	i	28.7	63

Table 5 shows the computation times and number of iterations needed for these experiments. Note that we do not report the depth errors here, since none of the modifications will have any impact on them. In addition, one iteration using the fast sweeping technique means iterating over the image in four directions, i. e. one iteration here corresponds to four iterations without sweeping. We observe that for the VBW method, the effect is rather small, while the other methods are more sensitive to this acceleration technique. In fact, the speed-up (which is the ratio between the two CPU times) is 1.54 for PF, 1.65 for CFS and 1.17 for VBW.

Discretising the source term implicitly slows down the VBW method quite a bit, but it remains the fastest method.

In the next step, we analyse the convergence properties of the methods. As we have already seen, the VBW method needs more iterations than the methods employing an optimal control approach, but requires less computation time. Obviously, the iteration steps are a lot cheaper for this method. Here, we aim at understand how fast the algorithms approach the solution. Again, we do this on the pyramid image. We stop each method as soon as the logarithmic depth change is less than 10^{-4} in all the pixels.

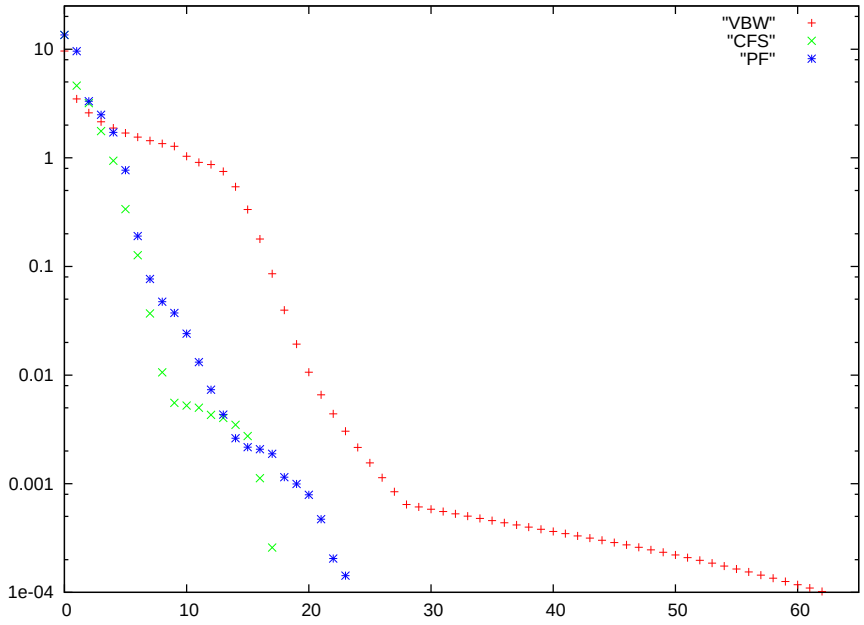


Fig. 6. Evolution of the maximal change in depth by iteration of the three methods on the pyramid image (with fast sweeping).

Figure 6 shows a graph of the convergence of the methods. As we can observe, the VBW method converges relatively fast up to a maximal change per pixel of about 10^{-3}, and then converges very slowly. The optimal control methods are a

bit slower in the beginning, in particular if we remember the iterations to be much more expensive, but at very small changes, they converge extremely fast. The CFS method reaches a maximum change of 10^{-4} within two iterations from a maximum change of 10^{-3}, while the VBW method takes more than 30 iterations for this. Figure 7 shows the same experiment without fast sweeping. We observe the same behaviour here.

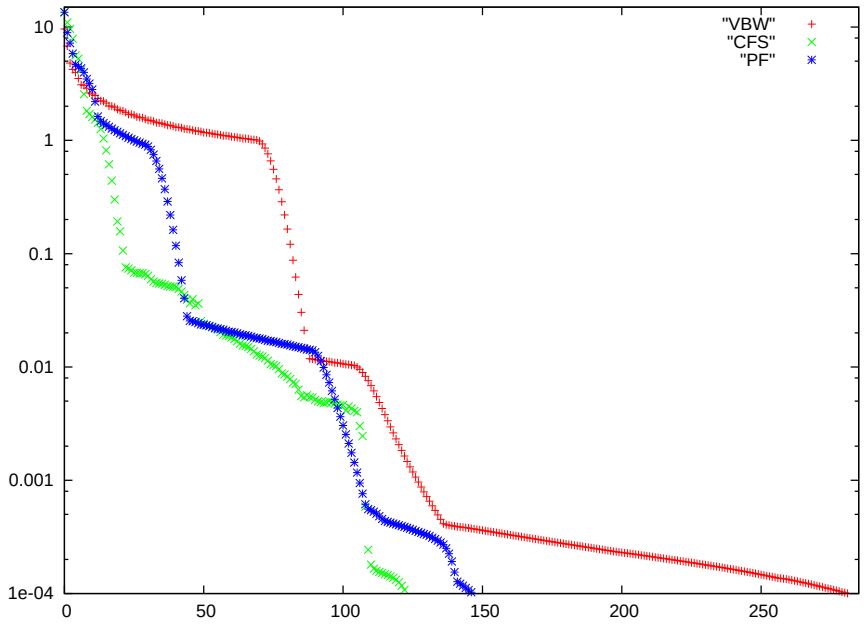


Fig. 7. Evolution of the maximal change in depth by iteration of the three methods on the pyramid image (without fast sweeping).

Finally, we investigate the effect of using a cascading multigrid scheme as a solver. We will do that in the same way for all three methods, employing fast sweeping as above, and we discuss the gain for each method. We do each experiment twice: In the first experiment we iterate on the coarser levels until convergence (10^{-4} as before). As usual, we begin at the coarsest level with a 2×2 resolution. Then, we repeat the experiment doing only 5 iterations – or until convergence if this happens earlier – on the coarser levels. We only iterate until convergence on the finest level, imposing a limit of 100 on the number of iterations that is by far not reached in practice.

Table 6 shows the computation times for both experiments and all the methods. We observe that the best choice in every case is to stop earlier the computation on the coarser levels. The overall gain for the optimal control approaches is very small, while for the VBW method we obtain a significant speed-up here.

Table 6. Analysis of the effect of a cascading multigrid method on the computation time for the pyramid image.

Method	CM (y/n)	coarse level iteration limit	Time [s]	Iterations (finest level)
PF	n	-	90.8	24
CFS	n	-	55.4	18
VBW	n	-	12.6	63
PF	y	none	89.6	20
CFS	y	none	54.8	15
VBW	y	none	8.9	28
PF	y	5	84.6	20
CFS	y	5	52.1	15
VBW	y	5	8.4	28

CONCLUSIONS AND FUTURE WORK

We have shown that the considered schemes deliver visually equivalent results of convincing quality, at least on continuous synthetic images. We have found that the direct approach based on the Hamilton–Jacobi PDE is more efficient compared to the schemes relying on the Hamilton–Jacobi–Bellman equation but the former is not always the most reactive to the acceleration techniques. Regarding the errors, outcomes are more test-dependent, and it is impossible to say which method performs best. As expected, all methods give convincing results for differentiable surfaces, while the errors increase in presence of edges and, even more, if the surface is discontinuous.

ACKNOWLEDGEMENT

Jean-Denis Durou would like to acknowledge the support of the CNRS in making possible his sabbatical at the CMLA (Centre de Mathématiques et de Leurs Applications).

(Received March 3, 2010)

REFERENCES

[1] F. Bornemann and P. Deuflhard: Cascadic multigrid methods. In: Domain Decomposition Methods in Sciences and Engineering (R. Glowinski, J. Periaux, Z. Shi, and O. Widlund, eds.), John Wiley, New York 1997, pp. 205–212.

[2] M. Breuß, O. Vogel, and J. Weickert: Efficient numerical techniques for perspective shape from shading. In: Proc. Algoritmy 2009, Podbanské 2009 (A. Handlovičová, P. Frolkovič, K. Mikula, and D. Ševcovič, eds.), Slovak University of Technology, Bratislava 2009, pp. 11–20.

[3] F. Camilli and E. Prados: Shape-from-Shading with discontinuous image brightness. Appl. Numer. Math. *56* (2006), 9, 1225–1237.

[4] F. Courteille, A. Crouzil, J.-D Durou, and P. Gurdjos: Towards shape from shading under realistic photographic conditions. In: Proc. 17th Internat. Conf. Patt. Recog. (vol. II), Cambridge 2004, pp. 277–280.

[5] E. Cristiani: Fast Marching and Semi-Lagrangian Methods for Hamilton–Jacobi Equations with Applications. Ph.D. Thesis, Dipartimento di Metodi e Modelli Matematici per le Scienze Applicate, Università di Roma "La Sapienza", Rome 2007.

[6] E. Cristiani, M. Falcone, and A. Seghini: Numerical solution of the perspective Shape-from-Shading problem. In: Proc. Control Systems: Theory, Numerics and Applications, Rome 2005. Proceedings of Science (CSTNA2005) 008, `http://pos.sissa.it/`

[7] E. Cristiani, M. Falcone, and A. Seghini: Some remarks on perspective Shape-from-Shading models. In: Proc. 1st Internat. Conf. Scale Space and Variational Methods in Comput. Vis., Ischia 2007 (F. Sgallari, A. Murli, and N. Paragios, eds., Lecture Notes in Comput. Sci. *4485*), Springer, Berlin 2008, pp. 276–287.

[8] J.-D. Durou, M. Falcone, and M. Sagona: Numerical methods for Shape-from-Shading: A new survey with nenchmarks. Comp. Vis. and Image Underst. *109* (2008), 1, 22–43.

[9] M. Falcone and R. Ferretti: Semi–Lagrangian schemes for Hamilton–Jacobi equations, discrete representation formulae and Godunov methods. J. Comput. Phys. *175* (2002), 2, 559–575.

[10] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes: Computer Graphics: Principles and Practice. Addison–Wesley, Reading 1996.

[11] R. H. W. Hoppe: Multigrid methods for Hamilton–Jacobi–Bellman equations. Numer. Math. *49* (1986), 2-3, 239–254.

[12] B. K. P. Horn: Obtaining shape from shading information. In: The Psychology of Computer Vision (P. H. Winston, ed.), McGraw-Hill, New York 1975, Ch. 4, pp. 115–155.

[13] B. K. P. Horn: Robot Vision. MIT Press, Cambridge Mass. 1986.

[14] B. K. P. Horn and M. J. Brooks: Shape from Shading. Artificial Intelligence Series, MIT Press, Cambridge Mass.1989.

[15] T. Okatani and K. Deguchi: Shape reconstruction from an endoscope image by Shape from Shading Technique for a point light source at the projection center. Comp. Vis. and Image Underst. *66* (1997), 2, 119–131.

[16] E. Prados and O. Faugeras: "Perspective Shape from Shading" and viscosity solutions. In: Proc. 9th IEEE Internat. Conf. Comp. Vis. (vol. II), Nice 2003, pp. 826–831.

[17] E. Prados and O. Faugeras: Unifying approaches and removing unrealistic assumptions in Shape From Shading: Mathematics can help. In: Proc. 8th Eur. Conf. Comp. Vis. (vol. IV), Prague 2004, Lecture Notes in Comp. Sci.*3024*, pp. 141–154.

[18] E. Prados, F. Camilli, and O. Faugeras: A unifying and rigorous shape from shading method adapted to realistic data and applications. J. Math. Imag. and Vis. *25* (2006), 3, 307–328.

[19] E. Prados, F. Camilli, and O. Faugeras: A viscosity solution method for shape-from-shading without image boundary data. M2AN Math. Model. Numer. Anal. *40* (2006), 2, 393–412.

[20] A. Rosenfeld: Multiresolution Image Processing and Analysis. Springer, Berlin 1984.

[21] E. Rouy and A. Tourin: A Viscosity solutions approach to Shape-from-Shading. SIAM J. Numer. Anal. *29* (1992), 3, 867–884.

[22] A. Tankus, N. Sochen, and Y. Yeshurun: A new perspective [on] Shape-from-Shading. In: Proc. 9th IEEE Internat. Conf. Comp. Vis. (vol. II), Nice 2003, pp. 862–869.

[23] O. Vogel, M. Breuß, and J. Weickert: A direct numerical approach to perspective Shape-from-Shading. In: Proc. Vision, Modeling, and Visualization Workshop 2007, Saarbrücken 2007 (H. Lensch, B. Rosenhahn, H.-P. Seidel, P. Slusallek, and J. Weickert, eds.), pp. 91–100.

[24] R. Zhang, P.-S. Tsai, J. E. Cryer, and M. Shah: Shape from Shading: A survey. IEEE Trans. Patt. Anal. Mach. Intell. *21* (1999), 8, 690–706.

[25] H. Zhao: A fast sweeping method for eikonal equations. Math. Comp. *74* (2004), 250, 603–627.

Michael Breuß, Faculty of Mathematics and Computer Science, Saarland University, Building E1.1, 66041 Saarbrücken. Germany.
e-mail: breuss@mia.uni-saarland.de

Emiliano Cristiani, CEMSAC, Università di Salerno, Fisciano (SA). Italy.
e-mail: emiliano.cristiani@gmail.com

Jean-Denis Durou, Centre de Mathématiques et de Leurs Applications, École Normale Supérieure de Cachan, 61 avenue du président Wilson, 94235 Cachan Cédex. France. (also: Institut de Recherche en Informatique de Toulouse, Université Paul Sabatier, 118 route de Narbonne, 31062 Toulouse Cédex 9. France.)
e-mail: durou@cmla.ens-cachan.fr

Maurizio Falcone, Dipartimento di Matematica, Università di Roma "La Sapienza", Piazzale Aldo Moro 2, 00185 Rome. Italy.
e-mail: falcone@mat.uniroma1.it

Oliver Vogel, Faculty of Mathematics and Computer Science, Saarland University, Building E1.1, 66041 Saarbrücken. Germany.
e-mail: vogel@mia.uni-saarland.de