Weighted Minimal Hypersurface Reconstruction

Bastian Goldlücke, *Student Member*, *IEEE*, Ivo Ihrke, *Student Member*, *IEEE*, Christian Linz, and Marcus Magnor, *Member*, *IEEE*

Abstract—Many problems in computer vision can be formulated as a minimization problem for an energy functional. If this functional is given as an integral of a scalar-valued weight function over an unknown hypersurface, then the sought-after minimal surface can be determined as a solution of the functional's Euler-Lagrange equation. This paper deals with a general class of weight functions that may depend on surface point coordinates as well as surface orientation. We derive the Euler-Lagrange equation in arbitrary dimensional space without the need for any surface parameterization, generalizing existing proofs. Our work opens up the possibility of solving problems involving minimal hypersurfaces in a dimension higher than three, which were previously impossible to solve in practice. We also introduce two applications of our new framework: We show how to reconstruct temporally coherent geometry from multiple video streams, and we use the same framework for the volumetric reconstruction of refractive and transparent natural phenomena, here bodies of flowing water.

Index Terms—Weighted minimal hypersurfaces, tomography, reconstruction, Euler-Lagrange formulation.

1 Introduction

A popular and successful way to treat many problems in computer vision is to formulate their solution implicitly as a hypersurface which minimizes an energy functional given by a weighted area integral. In this paper, we want to expose, generalize, and solve the mathematical problem which lies at the very heart of all of these methods.

Our aim is to find a k-dimensional regular hypersurface $\Sigma \subset \mathbb{R}^n$ which minimizes the energy functional

$$\mathcal{A}(\Sigma) := \int_\Sigma \Phi(s, \mathbf{n}(s))\, dA(s). \tag{1}$$

We will only investigate the case of codimension one, so throughout this text, $k = n - 1$. Such a surface is called a *weighted minimal hypersurface* with respect to the weight function Φ, which we require to be positive. This function shall be as general as required in practice, so we allow it to depend on surface point coordinates s and local surface normal $\mathbf{n}$. The weight function also has to be defined on the surrounding space, so the domain of Φ is $V \times \mathbb{S}^k$, where $V \subset \mathbb{R}^n$ is the region of interest where we are looking for the minimal surface. In particular, $s \in \mathbb{R}^n$ and $\mathbf{n}(s) \in \mathbb{S}^k$. By dA, we denote the infinitesimal area element of Σ, i.e., the functional $\mathcal{A}(\Sigma)$ is an area integral.

In the following, we derive an elegant and short proof of the necessary minimality condition, stated as:

Theorem 1. *A* k-dimensional *surface* $\Sigma \subset \mathbb{R}^{k+1}$ *which minimizes the functional* $\mathcal{A}(\Sigma) := \int_\Sigma \Phi(s, \mathbf{n}(s))\, dA(s)$ *satisfies the Euler-Lagrange equation*

$$\langle \Phi_s, \mathbf{n} \rangle - \mathrm{Tr}(\mathbf{S})\Phi + \mathrm{div}_\Sigma(\Phi_\mathrm{n}) = 0, \tag{2}$$

where $\mathbf{S}$ *is the shape operator of the surface, also known as the Weingarten map or second fundamental tensor.*

Using standard techniques, a local minimum can be obtained as a stationary solution to the surface evolution corresponding to (2). Since this surface evolution can be implemented and solved in practice, Theorem 1 yields a generic solution to all problems expressible in the form of (1). In this work, we set aside the problems of convergence and local minima. To our best knowledge, the necessary and sufficient conditions for convergence or uniqueness of a solution have not yet been found. For some mathematical background on the existence and uniqueness of solutions to this kind of equations, the reader is referred to [1]. We have, however, conducted a convergence analysis for our second application which uses an elaborate weight function. The results can be found in Section 5.

Theorem 1 offers two novel contributions:

1. *Unification*. A very general class of problems is united into one common mathematical framework. The kind of minimization problems we are concerned with arises in various different contexts in computer vision. A few select examples are given in Section 2. Our theorem yields the correct surface evolution equations for all of them.
2. *Generalization*. Theorem 1 is valid in arbitrary dimension. Previously, it has only been proved for surface dimensions $k = 1$ and $k = 2$. An analysis for $k = 1$ without a normal dependency appeared in the computer vision literature in the work of Caselles et al. [2], which was extended to surface dimension $k = 2$ in [3]. Faugeras and Keriven [4] introduced the dependency of Φ on the surface normal, and proved

- *B. Goldlücke and I. Ihrke are with the Max Planck Institute Informatik, Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany. E-mail: {bg, ihrke}@mpii.de.*
- *C. Linz and M. Magnor are with the Computer Graphics Lab, TU Braunschweig, Mühlenpfordtstr.23, 38106 Braunschweig, Germany. E-mail: {linz, magnor}@cg.cs.tu-bs.de.*

Manuscript received 11 Nov. 2005; revised 27 June 2006; accepted 8 Sept. 2006; published online 18 Jan. 2007.
Recommended for acceptance by S. Soatto.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0614-1105.
Digital Object Identifier no. 10.1109/TPAMI.2007.1146.

the theorem for $k = 2$ using local coordinates on the surface.[1]

While the theorem thus has already been well-known and employed in computer vision for problems $k \leq 2$, the now freely selectable surface dimension opens up new possibilities for novel applications. As one example, we generalize the static 3D reconstruction of a surface toward space-time coherent reconstruction of an evolving surface by regarding the surface as a 3D hypersurface in 4D space-time.

In the special case that the weight function Φ is constant, the problem of minimizing (1) is reduced to finding a standard minimal surface, which is defined to locally minimize area. As we deal with a generalization, it seems reasonable to adopt the same mathematical tools used in that context [5]. We give a brief review of this framework known as the *method of the moving frame* in Section 3.1. Using this framework, we prove Theorem 1 in Section 3.2. Due to space limitation, we have to assume that the reader is somewhat familiar with the differential geometry of frame bundles. The transition from the Euler-Lagrange equation to a level set evolution equation and further on to an explicit surface representation, is addressed in Section 3.3.

In Sections 4 and 5, we present two practical applications of our unifying framework. The first example application concerns spacetime-coherent geometry reconstruction from multiple views [6]. We outline details of a level set-based implementation and show how our technique can be used to reconstruct object surfaces from multiview video footage. In Section 5, we demonstrate how Theorem 1 enables us to reconstruct time-varying, transparent, and refractive natural phenomena like flowing water [7]. We give an overview on related work employing nonlinear computed tomography and describe the general reconstruction problem and its formulation as an energy minimization problem such that it fits into our framework. We again present important details on the construction of the energy functional and its realization using the level set technique. The approach is validated using both synthetic as well as real-world data.

2 Related Work

Weighted minimal surfaces have a wide range of applications in computer vision. Many solutions to computer vision problems can be formulated implicitly as the curve, surface, or volume that minimizes an integral of type (1). In the following, we use the term "surface" in a general sense, in particular, we do not restrict its dimension to two. Thus, a surface can also denote, e.g., a 1D line or 3D volume.

Among the first variational methods successfully applied to computer vision problems was the one now widely known as *Geodesic Active Contours* [2]. Active contours are a reformulation of the classical *snakes* approach [8] and aims to detect the reasonably smooth contour curve of an object in an image $\mathcal{I}$ by minimizing an energy functional. Caselles et al. realized that this energy minimization can be reformulated in terms of a geodesic computation in Riemannian space by means of Maupertuis' Principle. While originally designed for segmentation in 2D, it quickly became clear that it could be generalized to 3D [3], and could also be applied to other tasks. It is particularly attractive for modeling surfaces from point clouds [9], [10].

In [11], Paragios and Deriche extend the idea of Geodesic Active Contours to simultaneous tracking of the boundary curves of moving objects. They integrate a motion tracking term into the slightly modified energy functional used in [2]. Theoretically, well analyzed is also the case of employing minimal surfaces for 3D reconstruction of static objects from multiple views [4]. It is of particular interest and closely related to our spacetime-continuous 3D reconstruction. In their work, Faugeras and Keriven give several functionals of different complexities in dimension $n = 3$. It can be viewed as a space-carving approach generalized from discrete voxels to a continuous surface model [12]. This technique was recently extended to simultaneously estimate the radiance of surfaces, and demonstrated to give good results in practice [13].

All of these minimization problems fit into our unifying framework [14]. In particular, our theorem applies to all of them and yields the correct surface evolution equations.

3 Explicit Reconstruction of Weighted Minimal Hypersurfaces

In order to explicitly compute a hypersurface minimizing (1) in the general case, we first have to extend previous theoretical work [4] to cover the general case for arbitrary dimension. The goal of this section is, hence, to derive a necessary minimality criterion for error functionals of the form (1), in the form of the error functional's Euler-Lagrange equation.

The treatment of the general case requires mathematical tools from the differential geometry of hypersurfaces [5]. The mathematical framework for dealing with minimal surface problems are frame bundles of a variation of the surface. In the following, we introduce the notion of frame bundles of surface variations and make use of a few of their differential geometric properties. Having outlined the necessary mathematical tools, we proceed to derive an Euler-Lagrange equation that constitutes a necessary condition for the weighted minimal hypersurface sought in (1). This equation directly leads to a formulation as a surface evolution which can be implemented using a level set technique.

3.1 Some Background from Differential Geometry

We aim at giving a general proof that surfaces minimizing (1) can be obtained as a solution of the Euler-Lagrange equation (2) for the energy functional. Therefore, we make use of a mathematical tool called the *method of the moving frame*. Any minimal surface Σ of the functional $\mathcal{A}$ is a *critical point* of the functional, i.e., to first order, the value of the functional does not change under a small variation of the surface. This restriction is known as the functional's *Euler-Lagrange equation*. What follows is a, necessarily brief, overview of the mathematical framework in which this equation can be derived. For an excellent and thorough introduction, the reader is referred to [5].

We have to investigate how the functional behaves with respect to first order variations of the surface. To this end, let

$$X : \Sigma \times (-\epsilon, \epsilon) \to \mathbb{R}^n$$

1. The reader familiar with the earlier papers will notice that our result differs for the case $k = 2$ from the previously reported one [4] in that it is considerably simpler, because terms depending on $\langle \Phi_n, \mathbf{n} \rangle$ are missing. The reason for this is the different domain of Φ, see Section 3.2 for further discussion.

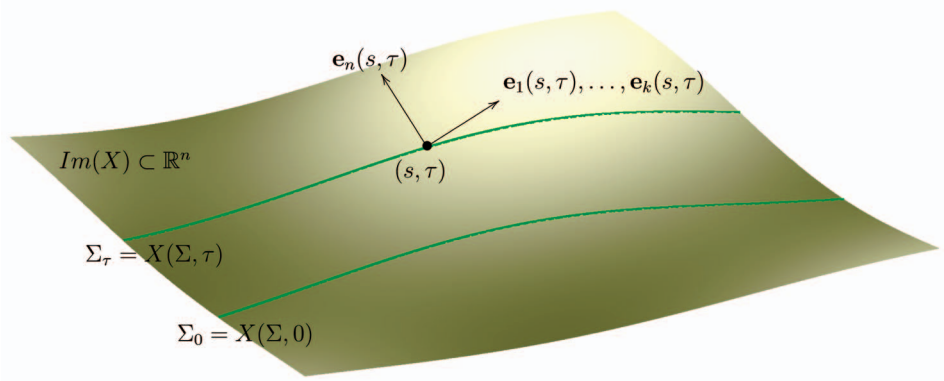


Fig. 1. Illustration of the frame bundle. The sheet represents the image $Im(X)$ of the variation X. The surfaces Σ_τ, depicted as lines, are distortions of Σ_0, where τ represents the variation parameter, defined in a small interval around zero. As a set, $Im(X)$ equals the union of all Σ_τ when τ is varied over this interval. A frame of $Im(X)$ at (s,τ) is given by the tangent vectors $\mathbf{e}_1(s,\tau),\ldots,\mathbf{e}_n(s,\tau)$.

be a variation of Σ with compact support. It then follows that for each $\tau\in(-\epsilon,\epsilon)$, a regular surface $\Sigma_\tau\in\mathbb{R}^n$ is given by $X(\Sigma,\tau)$. For each $(s,\tau)\in\Sigma\times(-\epsilon,\epsilon)$, let

$$\{\mathbf{e}_1(s,\tau),\ldots,\mathbf{e}_n(s,\tau)=:\mathbf{n}(s,\tau)\}$$

be an orthonormal frame for the surface Σ_τ at s with $\mathbf{e}_n=\mathbf{n}$ normal to the tangent plane $T_s\Sigma_\tau$. The restrictions ω^i of the Maurer-Cartan forms of $\mathbb{R}^n$ to this frame are defined by

$$dX=\mathbf{e}_i\,\omega^i. \tag{3}$$

Throughout this text, we use the Einstein convention for sums, which means that we implicitly compute the sum from 1 to n over all indices appearing twice on the same side of an equation. Because the frame is adapted to Σ_τ in the above sense, the forms ω^1 to ω^k are its usual dual forms on the surface. The *connection 1-forms* ω_i^j are defined by

$$d\mathbf{e}_i=\mathbf{e}_j\,\omega_i^j \tag{4}$$

and satisfy the *structure equations*

$$d\omega^i=-\omega_i^j\wedge\omega^j\quad d\omega_i^j=\omega_i^k\wedge\omega_k^j, \tag{5}$$

which can be deduced by differentiating the definitions. The connection forms lend this mathematical tool its true power. They allow us to express derivatives of the frame, in particular of the normal, in terms of objects which are *part of the frame bundle themselves* (Fig. 1). Thus, we can do entirely without local coordinates: All necessary information about the embedding of the surface in space is encoded in the connection forms.

From the Euclidean structure on $\mathbb{R}^n$, it follows that the connection 1-forms are skew-symmetric, $\omega_i^j=-\omega_i^j$. The connection forms ω_i^n can be expressed in the base $\{\omega^1,\ldots,\omega^k,d\tau\}$, courtesy of Cartan's Lemma [15]. To see this, first note that because of definition (3), it follows that

$$\omega^n=\langle dX,\mathbf{n}\rangle=\frac{\partial X}{\partial\tau}\,d\tau=:f\,d\tau. \tag{6}$$

Differentiating this equation yields, together with (5),

$$df\wedge d\tau+\sum_{i=1}^k\omega_i^n\wedge\omega_i=0.$$

Therefore, by Cartan's Lemma, there exist functions h_{ij} such that

$$\begin{bmatrix}\omega_1^n\\ \vdots\\ \omega_k^n\\ df\end{bmatrix}=\begin{bmatrix}h_{11}&\ldots&h_{1k}&f_1\\ \vdots&\ddots&\vdots&\vdots\\ h_{k1}&\ldots&h_{kk}&f_k\\ f_1&\ldots&f_k&f_n\end{bmatrix}\begin{bmatrix}\omega_1\\ \vdots\\ \omega_k\\ d\tau\end{bmatrix}. \tag{7}$$

The top-left part $\mathbf{S}:=(h_{ij})$ of this matrix is called the *shape operator*, and is closely related to the curvature of Σ_τ. In the lower dimensional cases, its entries are commonly known as follows:

- If $k=1$, i.e., Σ_τ is a curve in $\mathbb{R}^2$, the sole coefficient h_{11} equals the scalar-valued curvature usually denoted by κ.
- For $k=2$, i.e., if Σ is a regular surface in $\mathbb{R}^3$, the entries of $\mathbf{S}$ are the coefficients of the second fundamental form of Σ_τ. More precisely,

$$II=\begin{bmatrix}\omega^1&\omega^2\end{bmatrix},$$
$$\mathbf{S}\begin{bmatrix}\omega^1\\ \omega^2\end{bmatrix}=h_{11}(\omega^1)^2+2h_{12}\omega^1\omega^2+h_{22}(\omega^2)^2.$$

  Thus, $H=\frac{1}{k}\mathrm{Tr}(\mathbf{S})=\frac{1}{k}\sum_{i=1}^k h_{ii}$ is the mean curvature of the surface.

The f_i are just the directional derivatives of f in the directions of the $\mathbf{e}_i$. Using the structure equations (5), we immediately deduce an important relation for the area form dA on Σ_τ:

$$dA=:\omega_A=\omega^1\wedge\ldots\wedge\omega^k\Longrightarrow d\omega_A=-\mathrm{Tr}(\mathbf{S})\,\omega_A\wedge\omega^n. \tag{8}$$

We introduce the notation ω_A to remind the reader of the fact that the area element dA indeed is a differential form of degree k. Note that area in our sense does not imply "two-dimensional."

Finally, we need a notion of an "integration by parts" for a surface integral. First, we generalize the usual operators from vector analysis to vector fields $\mathbf{v}$ and functions f on Σ:

$$\mathrm{div}_\Sigma(\mathbf{v}):=\sum_{i=1}^k\frac{\partial v^i}{\partial\mathbf{e}_i}$$

with the expansion $\mathbf{v} = v^i\ \mathbf{e}_i$, and

$$\nabla_\Sigma f := \sum_{i=1}^{k} \frac{\partial f}{\partial \mathbf{e}_i}\ \mathbf{e}_i = \sum_{i=1}^{k} f_i e_i.$$

Using the definitions and the product rule, we derive a generalization of an identity well-known from classical vector analysis,

$$\mathrm{div}_\Sigma(\mathbf{v} f) = \langle \mathbf{v}, \nabla_\Sigma f \rangle + \mathrm{div}_\Sigma(\mathbf{v})\ f, \tag{9}$$

which will be useful later as one possibility of shifting partial derivatives from one object to another. A second possibility is given by Gauss' Theorem for surfaces, which in our context reads

$$\int_\Sigma \mathrm{div}_\Sigma(\mathbf{v})\ dA = -\int_\Sigma \mathrm{Tr}(\mathbf{S}) \langle \mathbf{v}, \mathbf{n} \rangle dA. \tag{10}$$

Note that $\mathbf{v}$ does not have to be tangential to Σ. Since we assume that all of our surfaces are closed, the boundary term usually contributing to the formula has vanished.

We now have collected all the necessary tools to derive the Euler-Lagrange equation (2) from (1). We will do so in the next section. In Section 3.3, this will lead to an evolution equation for the level sets of a function on $\mathbb{R}^n$.

3.2 Euler-Lagrange Equation

We are now in a position to use the moving frame method to derive the Euler-Lagrange equation of the functional $\mathcal{A}$. The derivation can be followed just by abstract manipulation of symbols, without the need to understand all of the reasons which lead to the governing rules presented in the preceding section.

The desired equation characterizes critical points of $\mathcal{A}$. It is given by the derivation of the functional with respect to τ at $\tau = 0$. We assume that $\Phi = \Phi(s, \mathbf{n})$ is a function of the surface point s and its normal $\mathbf{n}(s)$. Since Φ maps from $\mathbb{R}^n \times \mathbb{S}^k$, $\Phi_\mathrm{n}(s, \mathbf{n})$ is tangent to the unit sphere of $\mathbb{R}^n$ at $\mathbf{n}$, so that the important relation $\langle \Phi_\mathrm{n}(s, \mathbf{n}), \mathbf{n} \rangle = 0$ holds. Note that, in the well-known earlier work by Faugeras and Keriven [4], which treated the special case $k = 2$, the domain of Φ was $\mathbb{R}^n \times \mathbb{R}^n$. By restricting ourselves to normal direction, one gains substantial simplification: The final result becomes a lot more transparent because it is expressed in terms of only intrinsic quantities.

Let us now turn to the computation of the Euler-Lagrange equation. As a common convenient notation, we introduce $\mathbf{v} \rightharpoonup \omega := i_\mathbf{v}\omega$ for the inner derivative of a differential form ω with respect to $\mathbf{v}$. Using the Lie-derivative

$$\mathcal{L}_\mathbf{v}\omega = \mathbf{v} \rightharpoonup d\omega + d(\mathbf{v} \rightharpoonup \omega) \tag{11}$$

of a differential form ω in the direction of $\mathbf{v}$, we obtain

$$\begin{aligned}
&\frac{d}{d\tau}\|_{\tau=0}\ \mathcal{A}\Sigma_\tau \overset{(a)}{=} \int_\Sigma \mathcal{L}_{\frac{\partial}{\partial\tau}}(\Phi\ \omega_A) \overset{(b)}{=} \int_\Sigma \frac{\partial}{\partial\tau} \rightharpoonup d(\Phi\ \omega_A) \\
&\overset{(c)}{=} \int_\Sigma \frac{\partial}{\partial\tau} \rightharpoonup (d\Phi \wedge \omega_A + \Phi\ d\omega_A) \\
&\overset{(d)}{=} \int_\Sigma \frac{\partial}{\partial\tau} \rightharpoonup \left(\langle \Phi_s \mathbf{e}_i \rangle \omega^i \wedge \omega_A + \Phi_\mathrm{n}\ d\mathbf{n} \wedge \omega_A - \mathrm{Tr}(\mathbf{S})\Phi\ \omega_A \wedge \omega^n\right) \\
&\overset{(e)}{=} \int_\Sigma \Big[\left(\langle \Phi_s \mathbf{n} - \mathrm{Tr}(\mathbf{S})\Phi \right)\ f\ \omega_A + \frac{\partial}{\partial\tau} \rightharpoonup (\Phi_\mathrm{n}\ d\mathbf{n} \wedge \omega_A) \Big].
\end{aligned} \tag{12}$$

The five equalities above are justified by the following arguments:

1. A generalization of the "Differentiation under the integral"-rule in classic calculus [5].
2. Cartan's rule (11) for expressing the Lie derivative and using the fact that $\omega^1(\mathbf{n}) = \ldots = \omega^k(\mathbf{n}) = 0$. Note that $\frac{\partial}{\partial\tau}$ is parallel to $\mathbf{n}$, so this equation also holds for $\frac{\partial}{\partial\tau}$.
3. Product rule for differential forms, note that Φ is a 0-form.
4. Expansion of
$$d\Phi = \Phi_s\ dX + \Phi_\mathrm{n}\ d\mathbf{n} = \langle \Phi_s \mathbf{e}_i \rangle \omega i + \Phi_\mathrm{n}\ d\mathbf{n}.$$
Here, we inserted the definition (3) of the restrictions ω^i. The last term under the integral is due to (8).
5. Linearity of the inner derivative, and again
$$\omega^1(\mathbf{n}) = \ldots = \omega^k(\mathbf{n}) = 0.$$
From (6), it follows that $\omega^n(\frac{\partial}{\partial\tau}) = f d\tau(\frac{\partial}{\partial\tau}) = f$.

We now turn our attention to the second term of the last integral. Inserting definition (4) of the connection 1-forms, and afterward using the expansion of the connection forms (7) due to Cartan's Lemma, we get

$$\begin{aligned}
&\frac{\partial}{\partial\tau} \rightharpoonup (\Phi_\mathrm{n}\ d\mathbf{n} \wedge \omega_A) = \frac{\partial}{\partial\tau} \rightharpoonup \left(\langle \Phi_\mathrm{n} \mathbf{e}_j \rangle\ \omega_n^j \wedge \omega_A\right) \\
&= \frac{\partial}{\partial\tau} \rightharpoonup (-\langle \Phi_\mathrm{n}, \nabla_\Sigma f \rangle\ d\tau \wedge \omega_A) = -\langle \Phi_\mathrm{n}, \nabla_\Sigma f \rangle\ \omega_A \\
&= \mathrm{div}_\Sigma(\Phi_\mathrm{n})\ f\ \omega_A - \mathrm{div}_\Sigma(\Phi_\mathrm{n}\ f)\ \omega_A.
\end{aligned} \tag{13}$$

In the last equality, we have shifted derivatives using the product rule (9).We can finally compute the integral over the right term using Gauss' Theorem (10):

$$\int_\Sigma -\mathrm{div}_\Sigma(\Phi_\mathrm{n}\ f)\ dA = \int_\Sigma \mathrm{Tr}(\mathbf{S}) \langle \Phi_\mathrm{n}, \mathbf{n} \rangle\ f\ dA = 0.$$

It vanishes due to $\langle \Phi_\mathrm{n}, \mathbf{n} \rangle = 0$. By merging (12) and (13), we arrive at

$$\left.\frac{d}{d\tau}\right|_{\tau=0} \mathcal{A}(\Sigma_\tau) = \int_\Sigma \left(\langle \Phi_s, \mathbf{n} \rangle - \mathrm{Tr}(\mathbf{S})\Phi + \mathrm{div}_\Sigma(\Phi_\mathrm{n})\right)\ f\ dA.$$

Since for a critical point this expression has to be zero for any variation and hence for any f, we have arrived at the Euler-Lagrange equation of the functional

$$\langle \Phi_s, \mathbf{n} \rangle - \mathrm{Tr}(\mathbf{S})\Phi + \mathrm{div}_\Sigma(\Phi_\mathrm{n}) = 0, \tag{14}$$

proving our Theorem 1 (2).

3.3 Level Set Equation

Level sets represent an efficient way to implement a surface evolution [16], [17] and are by now a well-established technique that has found a wide range of applications [18]. We will briefly review the transition from (14) to a surface evolution equation. In the following, let

$$\Psi := \langle \Phi_s, \mathbf{n} \rangle - \mathrm{Tr}(\mathbf{S})\Phi + \mathrm{div}_\Sigma(\Phi_\mathrm{n}).$$

A surface $\hat{\Sigma}$ which is a solution to the Euler-Lagrange equation $\Psi = 0$ is also a stationary solution to a surface

evolution equation, where Ψ describes a force in the normal direction:

$$\frac{\partial}{\partial \tau}\Sigma_\tau = \Psi \mathbf{n}. \tag{15}$$

If we start with an initial surface Σ_0 and let the surface evolve using this equation, it will eventually converge to a local minimum of $\mathcal{A}$. Instead of implementing a surface evolution directly, we can make use of the level set idea. We express the surfaces Σ_τ for each parameter value $\tau \geq 0$ as the zero level sets of a regular function

$$\begin{aligned} &u : \mathbb{R}^n \times \mathbb{R}^{\geq 0} \rightarrow \mathbb{R},\ u(\cdot,\tau)^{-1}\{0\} = \Sigma_\tau, \\ &\text{i.e.}, u(s,\tau) = 0 \ \Leftrightarrow\ s \in \Sigma_\tau. \end{aligned} \tag{16}$$

We require $u(\cdot,\tau)$ to be positive inside the volume enclosed by Σ_τ and negative on the outside. An immediate consequence is this:

Lemma 1. *Let ∇ be the gradient operator for the spatial coordinates of u. Then, we can compute the outer normal and the trace of the shape operator for Σ_τ using*

$$\mathbf{n} = -\frac{\nabla u}{|\nabla u|} \text{ and } \mathrm{Tr}(\mathbf{S}) = \mathrm{div}\left(\frac{\nabla u}{|\nabla u|}\right).$$

Proof. The relationship for the normal is obvious. By definition, the shape operator is given by $\mathbf{S} := -D\mathbf{n}$ and maps the tangential space $T\Sigma_\tau$ into itself. It indeed follows that

$$\mathrm{Tr}(\mathbf{S}) = \mathrm{Tr}(-D\mathbf{n}) = \mathrm{div}(-\mathbf{n}) = \mathrm{div}\left(\frac{\nabla u}{|\nabla u|}\right).$$

Note that we consider the normal to be defined on all level sets of u. □

Taking the derivative of (16) with respect to τ and inserting (15), we derive the evolution equation for u,

$$\frac{\partial}{\partial \tau}u = -\left\langle \nabla u, \frac{\partial}{\partial \tau}\Sigma_\tau \right\rangle = -\langle \nabla u, \mathbf{n}\rangle\ \Psi = \Psi\,|\nabla u|. \tag{17}$$

Using the identities

$$\begin{aligned} &\mathrm{div}\left(\Phi \cdot \frac{\nabla u}{|\nabla u|}\right) = -\langle \Phi_s \mathbf{n}\rangle + \Phi\ \mathrm{div}\left(\frac{\nabla u}{|\nabla u|}\right) \text{ and} \\ &\mathrm{Tr}(\mathbf{S}) = \mathrm{div}\left(\frac{\nabla u}{|\nabla u|}\right) \end{aligned}$$

for the curvature of the level sets of u and the definition of Ψ, we arrive at the final reformulation of (15) in terms of a level set evolution:

$$\frac{\partial}{\partial \tau}u = \left[-\mathrm{div}\left(\Phi \cdot \frac{\nabla u}{|\nabla u|}\right) + \mathrm{div}_\Sigma(\Phi_{\mathrm{n}})\right]|\nabla u|. \tag{18}$$

Note that all necessary derivatives of Φ can be computed numerically. It is therefore not necessary to compute an explicit expression for them manually, which would be very cumbersome for more difficult functionals. Instead, in an existing implementation of the evolution essentially any functional $\Phi(s,\mathbf{n})$ can be plugged in. In particular, we will use the level set formulation introduced beforehand for spacetime-coherent geometry reconstruction as well as for the reconstruction of time-varying, refractive, and transparent natural phenomena like flowing water by defining a suitable functional $\Phi(s,\mathbf{n})$.

4 Application I: Spacetime-Coherent Geometry Reconstruction

After proving that the Euler-Lagrange equation (2), (14) is a necessary condition for the weighted minimal surface defined by (1), we present two novel applications of the variational reconstruction method in the remainder of the paper. In this section, we make use of our results to reconstruct time-varying geometry from a handful of synchronized video sequences in a global, spacetime-coherent fashion. To do so, we introduce a fourth dimension to represent the flow of time in the video sequence. Our goal is to reconstruct a smooth three-dimensional hypersurface embedded in space-time. The intersections of this hypersurface with planes of constant time are two-dimensional surfaces, which represent the geometry of the scene in a single time instant. Our approach defines an energy functional for the hypersurface. The minimum of the functional is the geometry which optimizes photo-consistency as well as temporal smoothness.

4.1 Space-Time 3D Reconstruction

We assume that we have a set of fully calibrated, fixed cameras. The input to our algorithm are the projection matrices for the set of cameras, as well as a video stream for each camera. We want to obtain a smooth surface Σ_t for each time instant t, representing the geometry of the scene at that point in time. The surfaces shall be as consistent as possible with the given video data. Furthermore, as in reality, all resulting surfaces are to vary continuously and smoothly over time (see Fig. 2).

To achieve these desirable properties, we do not consider each frame of the sequences individually. Instead, we regard all two-dimensional surfaces Σ_t to be subsets of one smooth three-dimensional hypersurface $\mathcal{H}$ embedded in four-dimensional space-time. From this viewpoint, the reconstructed surfaces

$$\Sigma_t = \mathcal{H} \cap (\mathbb{R}^3, t) \subset \mathbb{R}^3$$

are the intersections of $\mathcal{H}$ with planes of constant time. Because we reconstruct only one single hypersurface for all frames, the temporal smoothness is intrinsic to our method.

However, we have to take care of photo-consistency of the reconstructed geometry with the given image sequences. We set up an energy functional

$$\mathcal{A}(\mathcal{H})\ := \int_{\mathcal{H}} \Phi\ dA$$

which is defined as an integral of the scalar valued weight function Φ over the whole hypersurface. $\Phi = \Phi(s,\mathbf{n})$ measures the photo-consistency error density, and may depend on the surface point s and the normal $\mathbf{n}$ at this point. The larger the values of Φ, the higher the photo-consistency error, so the surface which matches the given input data best is a minimum of this energy functional. The Euler-Lagrange equation for the functional is given by

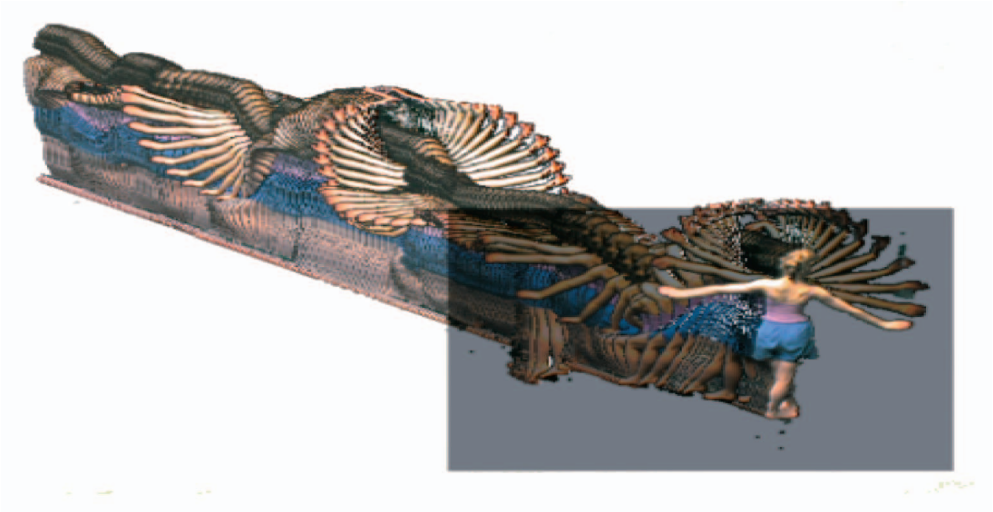


Fig. 2. A surface evolving over time defines a hypersurface $\mathcal{H}$, the space-time geometry of the scene.

Theorem 1, and we demonstrated in Section 3.3 how the Euler-Lagrange equation can be solved in practice using a surface evolution equation implemented via the level set method. In the remainder of this section, we present suitable choices for the error measure Φ.

4.2 Continuous Space-Time Carving

First, however, we need some additional notation for color and visibility of points in space-time. Let t denote a time instant. Then, a time-dependent image $\mathcal{I}_i^t$ is associated with each camera i. The camera projects the scene onto the image plane via a fixed projection $\pi_i : \mathbb{R}^3 \to \mathbb{R}^2$. We can then compute the color c_i^t of every point (s,t) on the hypersurface as

$$c_i^t(s) = \mathcal{I}_i^t \circ \pi_i(s).$$

Here, the image $\mathcal{I}_i^t$ is regarded as a mapping assigning color values to points in the image plane.

In the presence of the surface Σ_t, let $\nu_i^t(s)$ denote whether or not s is visible in camera i at time t. $\nu_i^t(s)$ is defined to be one if s is visible, and zero otherwise.

The most basic error measure can now be defined as

$$\Phi^S(s,t) := \frac{1}{V_{s,t}} \sum_{i,j=1}^{l} \nu_i^t(s)\nu_j^t(s) \| c_i^t(s) - c_j^t(s) \|,$$

l denoting the number of cameras.

The number $V_{s,t}$ of pairs of cameras able to see the point s at time t is used to normalize the function.

If the error function Φ^S is used as the functional, the resulting algorithm is similar to a space carving scheme in each single time step. In that method, as introduced by Kutulakos and Seitz [12], voxels in a discrete voxel grid are carved away if Φ^S lies above a certain threshold value when averaged over the voxel. In our scheme, the discrete voxels are replaced by a continuous surface. In the surface evolution introduced later, this surface will move inward until photo-consistency is achieved. This process is analogous to the carving process [12]. The same functional for regular surfaces in $\mathbb{R}^3$ was introduced by Faugeras and Keriven [4] for static scene reconstruction. As an additional constraint, we enforce temporal coherence in the form of temporal smoothness of the resulting hypersurface, which makes our method ideal for video sequences.

4.3 Normal Optimization

Since Theorem 1 also allows for error functions depending on the surface normal, we are able to optimize the surface normals as well. In their work, Faugeras and Keriven [4] already presented this idea for a static scene. We give a slightly modified version of the error function and work in space-time to enforce temporal smoothness.

In order to set up an appropriate error function, we have to analyze how well a small surface patch at position s with a given normal $\mathbf{n}$ fits the input images at time t. To this end, we assign to each hypersurface point s a small patch $\square_{s,\mathbf{n}}$ within the plane orthogonal to $\mathbf{n}$. How exactly this patch is chosen does not matter. However, the choice should be consistent over time and space and satisfy a few conditions which will become evident soon. In our implementation, we always choose rectangular patches rotated into the target plane by a well-defined rotation.

We will now define a measure how well the patch $\square_{s,\mathbf{n}}$ is in accordance with the images at time t. For that purpose, we employ the normalized cross-correlation of corresponding pixels in the images, a well-established matching criterion in computer vision. Mathematically, the resulting functional for a point $x = (s,t) \in \mathbb{R}^4$ with normal direction $\mathbf{n}$ is defined as follows:

$$\Phi^G(x,\mathbf{n}) := -\frac{1}{V_{s,t}} \sum_{i,j=1}^{l} \nu_i^t(s)\nu_j^t(s) \cdot \frac{\chi_{i,j}^t(s,\mathbf{n})}{A(\square_{s,\mathbf{n}})}$$

with the zero-mean cross-correlation

$$\chi_{i,j}^t(s,\mathbf{n}) := \int_{\square_{s,\mathbf{n}}} \left(c_i^t - \overline{\mathcal{I}}_i^{x,\mathbf{n}}\right)\left(c_j^t - \overline{\mathcal{I}}_j^{x,\mathbf{n}}\right) dA,$$

and the mean color value of the projected patch in the images computed according to

$$\overline{\mathcal{I}}_i^{x,\mathbf{n}} := \frac{1}{A(\square_{s,\mathbf{n}})} \int c_i^t \, dA.$$

Some things have to be clarified. First of all, the correlation measure $\chi_{i,j}^t$ for a pair of cameras is normalized

using the area $A(\square_{s,\mathbf{n}})$ of the patch. Second, it is now clear that we have to choose $\square_{s,\mathbf{n}}$ sufficiently large so that it is projected onto several pixels. On the other hand, it should not be too large; otherwise, only parts of it are visible in the images. As a compromise, we choose its diameter to be equal to the cell diameter of the underlying computation grid, as defined in Section 4.4. Third, the integration of Φ^G in the energy functional involves the normals of $\mathcal{H}$ in 4D space, while $\mathbf{n}$ is supposed to lie in $\mathbb{R}^3$. For that reason, we project normals of $\mathcal{H}$ into the tangent space of Σ_t in order to get $\mathbf{n}$.

When this functional is minimized, two constraints are optimized simultaneously. Each surface Σ_t together with its normals is selected to best match the images at that time instant. Furthermore, a smooth change of the surface Σ_t over time is encouraged because of the curvature term in the Euler-Lagrange equation (2). The error functional can be minimized using a surface evolution implemented via a level set scheme, as derived in Section 3.3. In the next section, we discuss the implementation details involved when the evolution equation is to be solved numerically.

4.4 Parallel Implementation

In order to implement the level set evolution equation (18), the volume surrounding the hypersurface $\mathcal{H}$ has to be discretized. We use a regular four-dimensional grid of evenly distributed cells with variable spatial resolution of usually 64^3 or 128^3 cells. The temporal resolution is naturally equal to the number of frames in the input video sequences. One easily calculates that there is a massive amount of data and computation time involved if the video footage is of any reasonable length. In fact, it is currently not yet possible to store the full data for each grid cell together with all images of a multiview video sequence within the main memory of a standard PC. A parallel implementation distributing the workload and data over several computers is therefore mandatory.

On that account, we choose the narrow band level set method [18] to implement the evolution equation because it is straightforward to parallelize. We start with an initial surface $\mathcal{H}_0$ and the values u_0^{xyzt} of the corresponding level set function u_0 in the centers of the grid cells. A suitable initial surface for our case will be defined at the end of this section. Using the abbreviation

$$\Psi(u) := \left[-\operatorname{div}\left(\Phi \cdot \frac{\nabla u}{|\nabla u|}\right) + \operatorname{div}_\Sigma(\Phi_{\mathrm{n}}) \right],$$

(18) simply reads

$$\frac{\partial}{\partial \tau} u = \Psi(u)\, |\nabla u|.$$

In the discretization, the values of the level set function are updated iteratively using the upwind scheme. At iteration step $i+1$, the new values u_{i+1}^{xyzt} are obtained from the values u_i^{xyzt} of the previous iteration step by a discrete version of (18) using an explicit time step:

$$u_{i+1}^{xyzt} = u_i^{xyzt} + \Psi(u_i^{xyzt})\, |\nabla u_i| \cdot \Delta\tau. \tag{20}$$

Here, $\Psi(u_i^{xyzt})$ is the value of the discretized version of the differential operator Ψ acting on u_i, evaluated in the cell (x, y, z, t). Central differences on the four-dimensional grid are used to compute the derivatives involved in (20). The

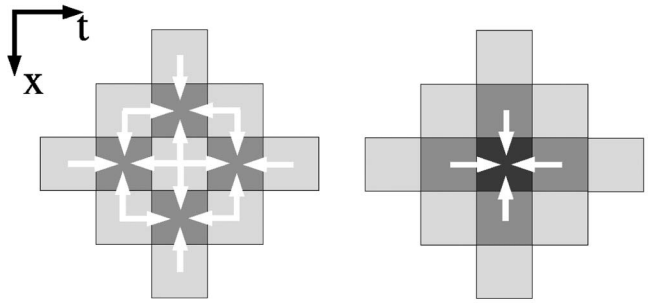


Fig. 3. Evaluation of the differential operator: In the first step, the values of u_i in the light gray cells are used to compute the level set normal $\mathbf{n} \in \mathbb{R}^4$ in the gray cells using central differences. Having computed $\mathbf{n}$, we can also compute Φ for the gray cells. Note that for the purpose of the above 2D illustration, the three spatial dimensions are represented as one. For the second step, we compute the values for the central dark gray cell, also using finite differences. The discrete formula for $\operatorname{div}(\Phi\mathbf{n})$ at position $p = (x, y, z, t)$ is $\sum_{i=1}^{4} \frac{\Phi^{p+e_i} n_i^{p+e_i} - \Phi^{p-e_i} n_i^{p-e_i}}{2}$. We can also compute the curvature $\operatorname{Tr}(\mathbf{S})$ directly by omitting Φ in the above formula. The difficult part is to compute $\operatorname{div}_\Sigma(\Phi_{\mathrm{n}})$ for the dark gray cell. It is equal to the trace of Φ_{ns} restricted to the tangent plane Π orthogonal to the normal at p. So, we first compute Φ_{n} for the gray cells using finite differences, taking the known normal $\mathbf{n}$ of the cell as the center point. With these values, we can set up the 4×4 matrix $U := \Phi_{\mathrm{ns}}$ for the dark gray cell. Choose an arbitrary orthonormal base $\{t_0, t_1, t_2\}$ of the plane Π. The entries for the 3×3 matrix V of the mapping $\Phi_{\mathrm{ns}}|_\Pi$ can then be computed as $v_{ij} = t_i^T U t_j, 1 \le i, j \le 3$.

norm of the discretized gradient $|\nabla u_i|$ is calculated according to the upwind scheme [18]. To ensure stability, the step size $\Delta\tau$ must be chosen such that the level sets of u_i cannot cross more than one cell at a time, i.e., satisfy the CFL-condition

$$\Delta\tau \le \max_{(x,y,z,t)\in\Gamma} \left(\frac{\operatorname{diam\,cell}(x, y, z, t)}{|\Psi(u_i^{xyzt}) \cdot \nabla u|} \right), \tag{21}$$

where Γ denotes the computational grid. The differential operator must be evaluated for each grid cell near the zero level set, so the computations necessary for each cell depend only on the local neighborhood. Therefore, the computation of individual cells can easily be distributed over several processes. In our implementation, each process is responsible for the computation of one single slice of the grid of constant time t_i. This slice corresponds to the geometry of the ith frame of the video sequence. Fig. 3 shows in more detail how the value $\Psi(u_i^{xyzt})$ is numerically evaluated from the values of u_i in the grid cells. According to this figure, we need the values of grid cells up to two cells apart from (x, y, z, t) in order to evaluate the operator. As a consequence, each process P_i also has to access the slices of four other processes $P_{i\pm1}, P_{i\pm2}$. These have to be communicated over the network. In addition, each process needs to store the image data of its own video frame and the two adjacent frames according to Fig. 3.

To summarize, one full iteration consists of the following four steps:

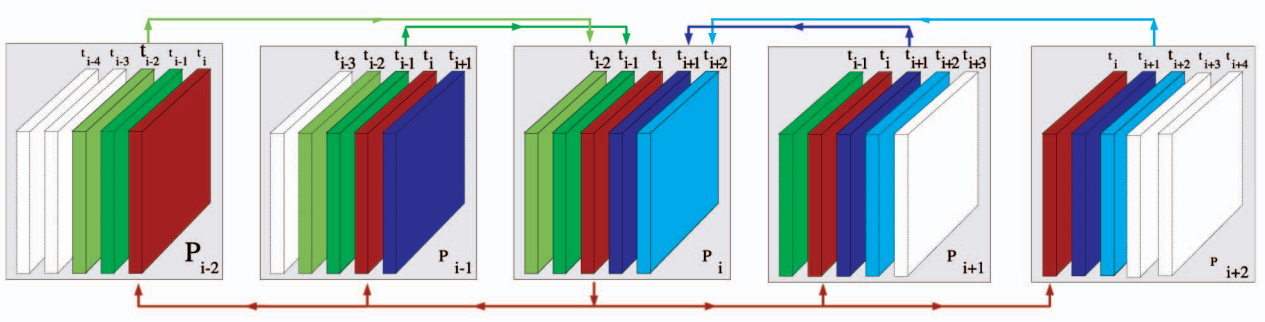


Fig. 4. Data transmission of process P_i before an iteration. Each process stores five slices of constant time and is responsible for the computation of the center slice. P_i computed its slice in the last iteration and now transmits it over the network. On the other hand, it receives the other slices from its neighbors for the next iteration. In the figure, slices of the same color contain the same information after the communication.

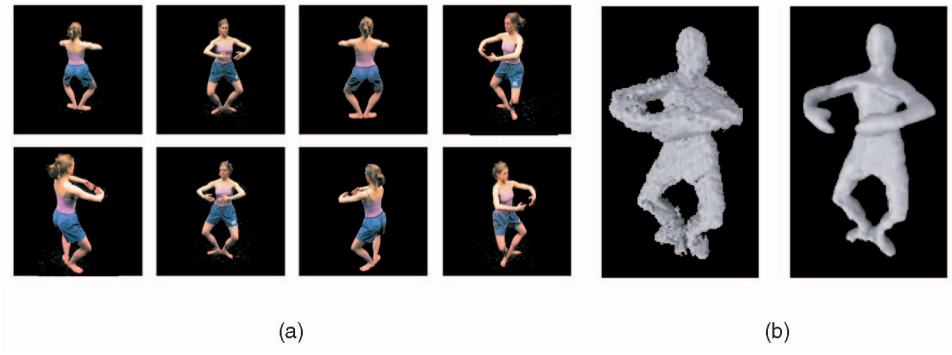


Fig. 5. A volumetric reconstruction of the visual hull serves as initial surface to start the PDE evolution. The final result upon convergence including normal optimization is the weighted minimal surface. (a) Foreground-segmented input images for one time frame. (b) Visual hull initialization. (c) Convergence result.

- Each process transmits its own slice S_i to the adjacent processes and receives the other necessary slices from its four neighbors according to Fig. 4.
- Afterward, each process computes $\Psi(u_i^{xyzt})$ for all cells in its slice near the zero level set of u_i, using the scheme presented in Fig. 3.
- The maximum value of the operator for each process is transmitted to a special server process. From these maxima, the server calculates the optimal step size $\Delta\tau$ allowed by (21).
- The server broadcasts the step size to all processes, which afterward compute the evolution on their slice using (20).

After each iteration, the server process may poll the current geometry from any of the other processes in order to give the user feedback about the current state of the iteration. The iteration stops when the flow field is zero, or may be stopped by the user manually. In our final implementation, it is also possible to assign several processes to a single slice. In that case, they share the computation of the cells near the zero level set equally between each other, assuming that all processes run on similar machines.

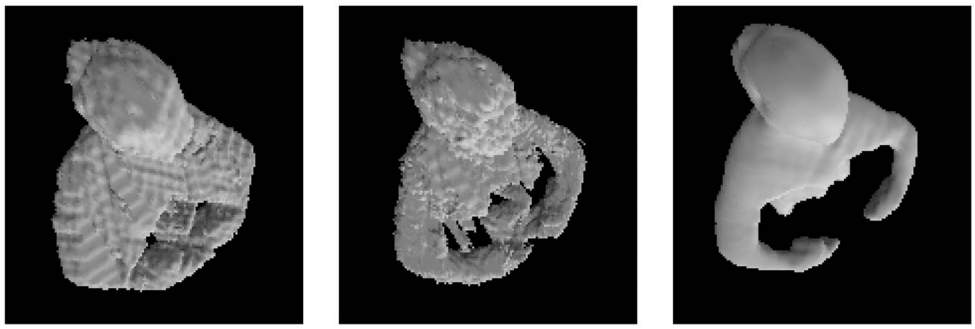
We still have to define a suitable initial surface $\mathcal{H}_0$ to start the iterative routine. For this purpose, we employ the visual hull [19], which is, by definition, always a superset of the correct scene geometry. By evolving $\mathcal{H}_0$ along the negative normal direction, we can be sure to converge toward a minimum. In order to compute a level set representation, we have to choose appropriate values of u_0 for each grid cell. To this end, we fix a grid cell c and select a number of evenly distributed sample points $x_0, \ldots, x_k$ inside it. These points are projected into each source image, and we compute the percentage $p \in [0, 1]$ of the projections falling into the silhouettes of the object to be reconstructed. Finally, the cell c of the initial level set function u_0 is assigned the value $2p - 1$. Since we only have to compute an approximate starting surface, this straightforward method gives sufficiently good results in practice. In particular, the projection of the zero level set of u_0 into the source images very closely resembles the silhouettes of the object if k is sufficiently high.

4.5 Results

In order to test our algorithm, we apply it to real-world 320×240 RGB video sequences of a ballet dancer. All input images are foreground-segmented using a thresholding technique, Fig. 5a. As initial surface, we compute a volumetric representation of the visual hull to get a starting volume for the PDE evolution, Fig. 5b.

For our test runs, we choose a 20 frame long part of the sequence with the depicted frame in the middle. As becomes apparent in Fig. 6, this frame is particularly difficult to reconstruct, because we do not have a camera capturing the scene from above. For that reason, most of the area in-between the arms of the dancer is not carved away in the initial visual hull surface.

When we run a standard space-carving algorithm for this single frame alone, the situation improves only slightly. The shirt of the dancer does not contain much texture



(a) (b) (c)

Fig. 6. Comparison of different reconstruction schemes at a grid resolution of 128^3. (a) The visual hull, as seen from above. Since we do not have a camera capturing the scene from above, most voxels in the area between the arms remain occupied. (b) The result obtained from static space carving. The difficult geometry between the arms is slightly improved. (c) When our algorithm using temporal information is employed, the reconstruction becomes almost optimal.

information, so only parts of the critical region is carved away. Only when we employ the weighted minimal hypersurface formulation, which takes temporal coherence between the geometry of the frames into account, do we get satisfactory results, Fig. 6c.

Our program runs on a Sun Fire 15K with 75 UltraSPARC III+ processors at 900 MHz, featuring 176 GBytes of main memory. It can be observed that the normal optimization, Section 4.3, requires a lot of computation time when compared to the version of Section 4.2 of our algorithm. For that reason, we first let the geometry evolve toward a surface which is very close to the optimal result, as assessed by the operator of the program. We then switch on the normal optimization in order to improve the reconstruction of small surface details. On average, we need around 100 iterations on the initial evolution and 20 more of the normal optimization until the surface has converged to the final result.

In order to speed up the surface evolution, a further term is included in (20), as suggested in [4]. We subtract a multiple $\epsilon \mathrm{Tr}(\mathbf{S})$ of the curvature, where ϵ is a small user-defined constant factor. This forces the resulting hypersurface to be smoother, so larger step sizes $\Delta\tau$ can be considered to evolve the PDE.

5 Application II: Nonlinear Computed Tomography

We now turn to another application of our framework, i.e., the reconstruction of free-flowing bodies of water from multiview video sequences. This work fits into a line of research, different from the traditional diffuse surface reconstruction, recently emerging in the field of computer vision.

Image-based modeling of natural phenomena suitable for free-viewpoint video is performed using sparse view tomographic methods [20], [21] or surface-based methods [22].

Only limited work has been done which directly addresses image-based reconstruction of water. In [23], [24], a time-varying water surface is obtained by analyzing the distortion of a known texture beneath the water surface using optical flow and shape from shading techniques. Morris and Kutulakos [24] handle unknown refractive indices of the liquid. Schultz [25] studies the reconstruction of specular surfaces using multiple cameras. He reports good results on synthetic test data, a simulated water surface under known synthetic illumination. However, these methods can only determine a height field for a rectangular surface area, while our approach is capable of reconstructing fully three-dimensional bodies of water.

Another line of research is refractive index tomography, e.g., [26], [27]. These methods usually need expensive apparatuses and do not lend themselves to image-based modeling. The goal of these methods is also quite different from ours: Whereas refractive index tomography attempts to reconstruct a field of *varying* refractive indices, we reconstruct the surface of a volume with constant refractive index.

Kutulakos and Steger [28] present a theoretical analysis of specular and refractive light transport. They found that it is impossible to obtain a unique solution for light paths that involve more than two refractions or reflections. However, their work does not take spatial continuity of the surface into account. Surface continuity is intrinsic to our method; therefore, we have natural regularization built into our reconstruction algorithm.

The work so far concentrates on nonrefracting media. The problem arises in the context of free-viewpoint video, where we are concerned with the automatic acquisition of dynamic models for computer graphics purposes. The surface structure of water cannot be determined with traditional methods due to refraction effects. We alleviate this problem by using the effect of chemoluminescence. Two chemicals are mixed, causing a reaction that emits light uniformly in all directions. This allows us to directly measure the thickness of the water volume as a column length of a line passing through the water. With this information, we define a weight function Φ that measures photo-consistency between the acquired video frames and an intensity computed using the image formation model and the current surface approximation.

In the following, we first state the reconstruction problem we want to deal with. Again, we will make use of the framework introduced in Section 3. Details on our implementation are given in Section 5.2, followed by a presentation

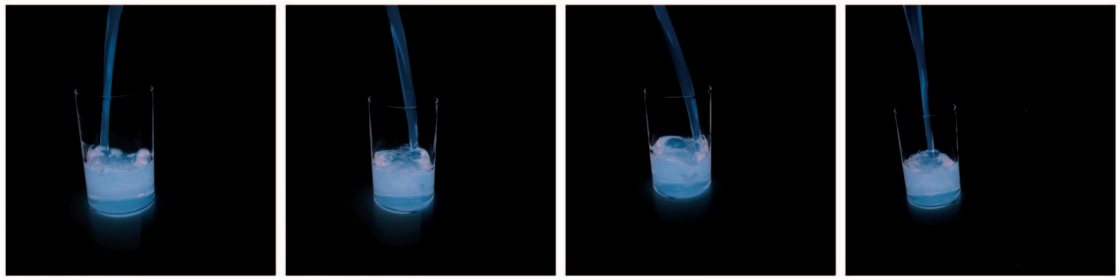


Fig. 7. Four of eight camera views from our test video sequence. The images were taken at the same point in time.

of our results obtained on both synthetic 2D data as well as recorded data of flowing water in Section 5.3.

5.1 Reconstruction Problem

Our goal is to reconstruct the surface area of a possibly moving body of water, using recordings from only a handful of fully calibrated cameras distributed around the scene. In order to be able to work with a well-defined image formation model, special care has to be taken when acquiring the water video data. We employ a chemical reaction that emits light over an extended period of time. Two chemicals are mixed, and the resulting chemical reaction causes light to be emitted uniformly in all directions. Glowsticks are a commercial application and their chemical composition is optimized for brightness and longevity of the chemical reaction. Thus, they are ideally suited for our task. A minor drawback is a higher viscosity of the fluid in comparison to water. Example input images of our input video sequences are shown in Fig. 7.

In the following section, we discuss the image formation model underlying the reconstruction approach. It shows how to generate synthetic views given a certain reconstructed surface Σ, which can be compared to recorded real-world data in order to define a photo-consistency error measure. The "best" surface is determined by employing the framework of Section 3. The numerical solution of the fixed point iteration, (18), is similar to the previous application. After the theoretical discussion in this section, we proceed with the details of the implementation in Section 5.2.

5.1.1 Image Formation Model

We use a chemoluminescent chemical reaction to make the water self-emissive. When the chemicals are evenly dissolved the reaction takes place in a uniform manner. This allows us to assume a constant emissivity throughout the volume. Thus, the accumulated light intensity along a ray traced through the water can be computed by multiplying its total length within the volume with a constant emittance ρ. We perform a photometrical calibration on the cameras, such that they exhibit a linear response to the incoming light intensity, scaling light intensity to image intensity by a factor of γ.

Now, let p be a point in the image plane of camera C, and C be the camera's center of projection. We want to compute the theoretical pixel intensity $I_\Sigma(p)$ in the presence of a surface Σ, enclosing a volume $\mathcal{O}_\Sigma$ of water prepared as above. Let $R(C,p)$ be the ray traced from C in the direction of p through the surface Σ, taking into account correct refraction. We ignore scattering and extinction effects in the water volume. Then,

$$I_\Sigma(p) = \gamma \int_{R(C,p)\cap\mathcal{O}_\Sigma} \rho \, ds = \rho\gamma \int_{R(C,p)\cap\mathcal{O}_\Sigma} ds.$$

The last integral just measures the length the ray traverses through $\mathcal{O}_\Sigma$. In order to avoid having to determine the constant factor $\rho\gamma$ experimentally by acquiring and measuring a calibration scene, we implement an autocalibration scheme. All image intensities are divided by the average intensity of the pixels in the image within the silhouette, and all ray-traced intensities by the average intensity of the rays corresponding to these pixels. The resulting quotients are independent of the quantity $\rho\gamma$.

Now that we are able to compute synthetic views given a surface Σ, we have to determine how well a reconstructed surface fits a given set of input views. If we are able to quantify the error, it can be used to define an energy functional mapping surfaces to real numbers, whose minimum yields an optimal reconstruction result.

5.1.2 Energy Minimization Formulation

We have to observe photo-consistency of a reconstructed surface Σ given the set of source images. We set up an energy functional of the form introduced in (1) with a scalar valued weight function Φ measuring the photo-consistency error density. It may depend on the surface point s and the surface normal $\mathbf{n}$. Because refraction occurs frequently, the dependency of the error measure on the normal is a vital part of our method, in contrast to many other previous applications of weighted minimal surfaces in computer vision.

The question remains how to correctly choose the error measure. Ideally, we would want it to be the difference of the measured intensity in every camera with the theoretical intensity, which would look like

$$\Phi_{\text{naive}}(s,\mathbf{n}) := \sum_{i=1}^{n} \big(I_{\Sigma,i}(s) - I_i \circ \pi_i(s)\big)^2,$$

where $I_{\Sigma,i}(s)$ is the ray-traced image intensity assuming surface Σ, I_i is the ith camera image, and π_i the ith camera's projection mapping.

While the general idea is good and exactly what we implement, it faces several problems in this initial form, the worst being that we have to be able to evaluate the error function away from the surface in order to perform the

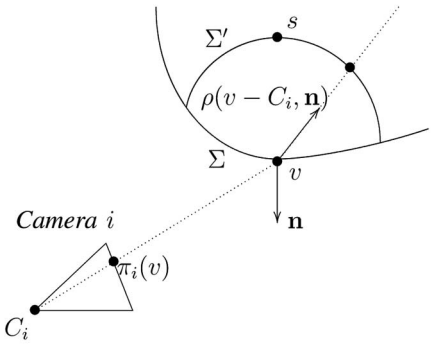


Fig. 8. Evaluation of the partial error function ϕ_i for a single camera. The length difference between rays traced through the distorted surface Σ' and the undistorted surface Σ is just $\|s - v\|$. Note that $\mathbf{n}$ is not necessarily the exact surface normal, it may vary close to it in order to evaluate the derivative of Φ with respect to the normal.

surface evolution. The exact technical definition is presented in the next section. As in Section 4, Theorem 1 yields the Euler-Lagrange equation of the functional, which leads again to the surface evolution equation and level set implementation introduced in Section 3.3.

5.2 Implementation

In the following, we go into the details on how to implement our reconstruction scheme, specifying the construction of the error function. For a stable evolution, we have to make sure that the surface does not shrink inside the image silhouettes. To this end, we introduce a silhouette constraint. We finally describe some nuts and bolts of the implementation of the PDE as a narrow band level set method.

5.2.1 Construction of the Error Function

Of particular difficulty is the evaluation of the error function $\Phi(s, \mathbf{n})$ for a given point s and corresponding normal $\mathbf{n}$. The problem is that this term has to be evaluated away from the current surface Σ in order to numerically compute the derivatives in (18), i.e., for points that do not lie directly on the surface, and with a normal which may be different from the current surface normal. In fact, the question is what local error would arise *if the surface was distorted* such that it lies in s with normal $\mathbf{n}$. For this reason, ray tracing in order to evaluate the error function has to be performed for a distorted surface Σ'. The computation of $\Phi(s, \mathbf{n})$ is thus performed in three steps.

In the first step, we construct the distorted surface Σ' through which rays are traced. We have to change Σ locally in a reasonably smooth manner such that the new surface passes through s. At this moment, we do not yet care about the normal. Assume for now that s lies outside the volume $\mathcal{O}_\Sigma$ enclosed by Σ. The desired result can then be achieved by uniting $\mathcal{O}_\Sigma$ with a sphere B centered in the point v closest to s on Σ, with radius $\|s - v\|$. Vice versa, if s lies inside $\mathcal{O}_\Sigma$, we can achieve the result by subtracting B from $\mathcal{O}_\Sigma$, Fig. 8.

The second step is to define the set of cameras $\mathcal{C} = \{C_1, \ldots, C_k\}$ which contribute to the error measure. Ideally, since the medium is transparent, we would like to consider all cameras we have available. Unfortunately, this requires finding for each camera the ray passing from the camera center to s, possibly refracted multiple times on the way. This computation definitely is too time-consuming. Instead, we only consider those cameras which have a reasonable unobscured view of v with regard to the original surface. More precisely, each camera C_i belonging to $\mathcal{C}$ must meet the following two criteria:

- the straight line from v to the center of projection C_i must not intersect Σ, and
- the ray starting from v in the refracted direction $\rho(v - C_i, \mathbf{n})$ must travel inside $\mathcal{O}_\Sigma$ in the beginning. ρ is computed using Snell's law, using the index of refraction of water for inside the volume, and of vacuum for outside.

In the third step, we finally compute the photo-consistency error ϕ_i for each contributing camera C_i and average those to get the total error Φ. Each individual error is computed as follows: Let $\mathcal{I}_i \circ \pi_i(\mathbf{s})$ be the intensity of the projection of s in image $\mathcal{I}_i$, and $r_i(s, \mathbf{n})$ be the accumulated intensity along a ray traced from s into the refracted direction $\rho(s - C_i, \mathbf{n})$. Then,

$$\phi_i(s, \mathbf{n}) := (\mathcal{I}_i \circ \pi_i(\mathbf{s}) - r_i(s, \mathbf{n}))^2.$$

This corresponds to comparing the image intensity to the ray-traced intensity of a ray cast from the camera to s, refracted by a surface located in s with normal $\mathbf{n}$. Thus, the desired normal $\mathbf{n}$ is also correctly taken into account.

Unfortunately, the resulting weight function Φ is not locally dependent on s and $\mathbf{n}$ because the distortion of Σ changes Φ globally. The silhouette constraint introduced in the next subsection counters this shortcoming and experiments on synthetic test data suggest the feasibility of the reconstruction approach, see Fig. 9 for a qualitative analysis.

5.2.2 Silhouette Constraints

An additional constraint on the photo-consistency of the reconstruction result is that the projection of the reconstruction in each camera image must match the silhouette of the object to be reconstructed [12]. This constraint yields both a stopping term in our evolution equation, as well as an initial surface for the evolution in form of the visual hull [29]. We prohibit the projections from ever shrinking inside any of the silhouettes. A stopping term is therefore added to the surface evolution, which grows very large if a point on the projected boundary of the surface lies inside a silhouette. When computing the visibility of a point v, we can extract from the set of unobscured views $\mathcal{C}$ the set of cameras $\mathcal{B} \subset \mathcal{C}$ in which v lies on or very close to the boundary of the projection. The two criteria for camera C_i in $\mathcal{C}$ to lie in $\mathcal{B}$ as well is that

- the angle between viewing direction $\mathbf{d}_i$ from v to the center of projection C_i and the surface normal $\mathbf{n}(v)$ must be close to 90 degrees and
- the straight line from v in the direction $\mathbf{d}_i$ away from the camera must not intersect the surface.

Then, the boundary stopping term is defined as

$$B(s) := \sum_{C_i \in \mathcal{B}} [\exp(-\beta(\sigma_i \circ \pi_i)(v)) - 1],$$

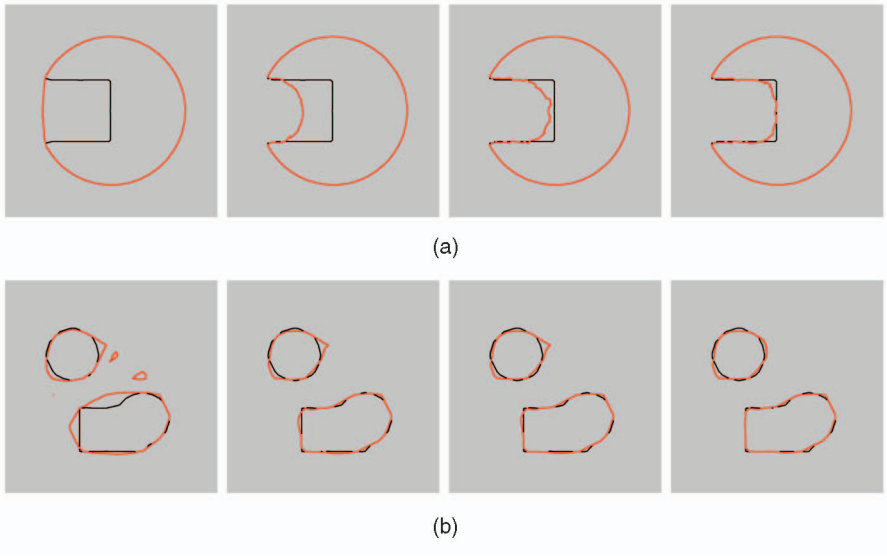


Fig. 9. The best results we achieved using 24 input views, together with several in-between stages of the iteration. (a) Convergence toward the first test volume, after 0, 100, 200, and 300 iterations. (b) Convergence toward the second test volume, after 0, 15, 30, and 45 iterations.

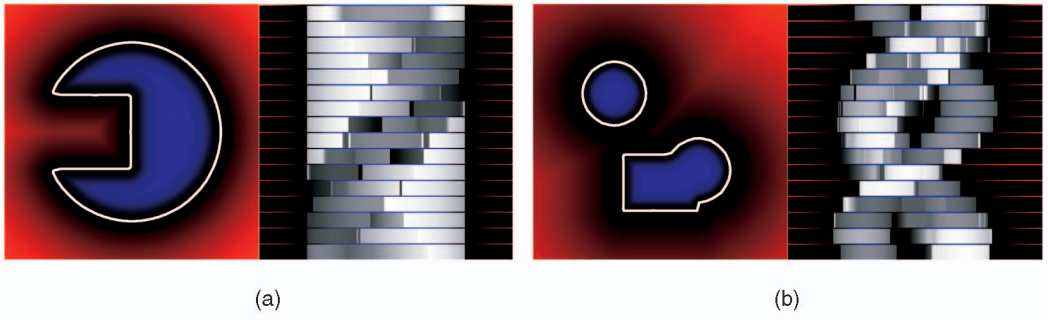


Fig. 10. Synthetic 2D test geometries of Fig. 9 and ray-traced 1D projections (stacked).

where v is again the point closest to s on Σ, and $\beta > 0$ a user-defined weight, which should be set reasonably high. We use $\beta = 10$ throughout all of our tests, where the images are defined to lie in $[0,1]^2$, and the signed distance is normalized accordingly.

5.2.3 PDE Discretization

Similar to Section 4.4, the volume surrounding the surface Σ has to be discretized. We use a regular three-dimensional grid of evenly distributed cells with variable spatial resolution of usually 64^3 or 128^3 cells. The surface is evolved according to the narrow band level set method [18], starting the evolution with the visual hull surface Σ_0 and the values u_0^{xyz} of the corresponding level set function u_0 in the centers of the grid cells. Details on how the evolution equation is implemented were already presented in Section 4.4. However, there are two optimization terms which are added to the values in the cells after each update step (20).

The first one is the boundary term $B(x,y,z)$. The second term is designed to speed up convergence and avoid local minima. It accelerates the shrinking process in regions where the error is excessively high. We add to u_{i+1}^{xyz} the value

$$\epsilon_1 B(x,y,z) - \epsilon_2 L_{\sigma(\Phi)}(\Phi(x,y,z) - m_\Phi),$$

where $L_{\sigma(\Phi)}$ is the stable Leclerc M-estimator for the standard deviation of the error values of all cells, and m_Φ the mean value of the error. $\epsilon_1, \epsilon_2 > 0$ are two user-defined weights. Good choices and their influence on convergence behavior are discussed in the next section.

5.3 Results

5.3.1 Synthetic 2D Experiments

In order to verify that our surface evolution is capable of producing correct results despite the complex problem we want to solve, we first test it on synthetic 2D data. For this purpose, we ray-trace several views of two different test volumes using the image formation model presented. The first volume is designed to test how well the algorithm can recover concavities, while the second volume is not connected and has a mixture of straight and round edges. Both test volumes and resulting 1D views are shown in Fig. 10.

We run our algorithm with different numbers of input views in order to test the dependence of convergence on this critical parameter. Convergence becomes stable if eight or more cameras are available, with 12 views required in the more complex second test case. We also note that there is a quick saturation of reconstruction quality with respect to the number of cameras because the visual hull does not

TABLE 1
Error in the Reconstruction of the Volume Shown in Fig. 9a after 200 Iterations, Depending on Different Choices of ϵ_1 and ϵ_2

		ϵ_1				
		0.01	0.1	0.5	1	5
ϵ_2	1	0.07	U	U	U	U
	10	0.05	0.04	0.06	U	U
	50	0.16	0.07	0.03	0.04	U
	100	0.04	0.05	0.04	0.06	U
	1000	S	S	S	S	0.03

An entry of "U" indicates instability and "S" indicates a stopped evolution.

improve further if more than 16 cameras are used, in accordance with earlier results [30]. In addition, the quality of the reconstruction levels out at around 24 cameras for both test volumes. Our experiments show that more cameras do not yield a better result, which indicates that a good placement of the cameras is at least as important as their sheer number.

In all cases, the algorithm runs with the same parameter values of $\epsilon_1 = 0.1$ and $\epsilon_2 = 100$. These values give stable behavior against parameter changes using 24 cameras to estimate the first test volume. As a rule of thumb, there is a certain threshold value for the speedup term above which it accelerates the evolution above a stable limit, causing the surface to shrink inside the silhouettes. Too low a choice of ϵ_1 has no ill effects on stability, but slows down convergence. ϵ_2 can safely be chosen somewhere between 10 and 100 without much effect, but may cause the surface to be stuck at an undesirable spot if set too high. Table 1 shows the reconstruction error, i.e., the difference between the ground truth area (Fig. 9) and the area enclosed by the reconstructed surface, after 200 iterations for the first test volume and different choices of ϵ_1 and ϵ_2.

5.3.2 Real-World Water Videos

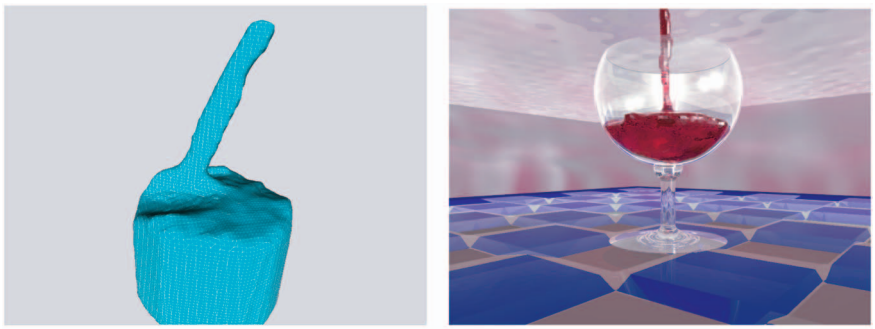
For the real-world tests, we use a multivideo recording setup consisting of eight CCD-cameras with a resolution of $1,004 \times 1,004$ pixels. The cameras record at 45 frames per second. The cameras are geometrically and photometrically calibrated. We acquire our test sequences in the dark, the chemiluminescent water being the only source of light. This allows for simple background subtraction. We record a dark sequence and measure the noise distribution of the cameras' CCD chips. Pixels within a range of two standard deviations of the mean noise value are classified as background. We perform a clean-up of the foreground masks using morphological operations. The reconstruction is performed on an equidistant, uniform grid of 128^3 voxels. An example of a reconstructed water surface is shown in Fig. 11.

6 Summary and Future Work

We have derived the Euler-Lagrange equations for weighted minimal hypersurfaces in arbitrary dimensions. We allowed for weight functions general enough to cover many variational problems frequently encountered in computer vision research. Compared to existing proofs which are restricted to dimensions two or three, our approach is valid in arbitrary dimension. We believe that the presented results pave the way for new applications that rely on higher dimensional representations.

As one application exploiting arbitrary dimensionality, we showed in the second part how to reconstruct temporally coherent geometry from multiple video streams using a level set technique. The idea is to optimize photo-consistency with all given data as well as to enforce temporal smoothness. Our method is formulated as a weighted minimal surface problem posed for a 3D hypersurface in space-time. The energy functional defining the minimization problem enforces photo-consistency, while temporal smoothness is intrinsic to our method. Significant improvements compared to space carving approaches which lack temporal coherence can be observed. As future work along this line of research, we plan to include global optimization of surface reflectance properties into the same unifying framework.

As a second application of our theoretical framework, we have presented a method for the reconstruction of flowing water surfaces. A novel recording methodology and a corresponding image formation model allow us to define a photo-consistency constraint on the reconstructed surface taking refraction into account. We again utilize weighted minimal surfaces to refine the visual hull of the water using



(a) (b)

Fig. 11. (a) Reconstructed water surface for a fixed time instant, see Fig. 7. (b) Reconstructed flow of another water volume rendered in a virtual environment.

constraints based on thickness measurements of the real surface. Real-world experiments demonstrate the suitability of our method for the reconstruction of water. Next, we intend to develop a hierarchical representation of the underlying computational grid to achieve higher resolution reconstruction which allows to resolve finer details.

ACKNOWLEDGMENTS

This work was partially founded by the German Research Foundation DFG under contract number MA2555/1.

REFERENCES

[1] Y. Chen, Y. Giga, and S. Goto, "Uniqueness and Existence of Viscosity Solutions of Generalized Mean Curvature Flow," *J. Differential Geometry,* vol. 33, pp. 749-786, 1991.
[2] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic Active Contours," *Proc. Int'l Conf. Computer Vision,* pp. 694-699, 1995, citeseer.nj.nec.com/caselles95geodesic.html.
[3] V. Caselles, R. Kimmel, G. Sapiro, and C. Sbert, "Three Dimensional Object Modeling Via Minimal Surfaces," *Proc. European Conf. Computer Vision,* vol. 1, pp. 97-106, Apr. 1996.
[4] O. Faugeras and R. Keriven, "Variational Principles, Surface Evolution, PDE's, Level Set Methods and the Stereo Problem," *IEEE Trans. Image Processing,* vol. 3, no. 7, pp. 336-344, Mar. 1998.
[5] J. Clelland, "MSRI Workshop on Lie Groups and the Method of Moving Frames," Dept. of Math., Univ. of Colorado, July 1999, http://spot.Colorado.EDU/~jnc/MSRI.html.
[6] B. Goldluecke and M. Magnor, "Space-Time Isosurface Evolution for Temporally Coherent 3D Reconstruction," *Proc. Conf. Computer Vision and Pattern Recognition,* pp. 350-355, July 2004.
[7] I. Ihrke, B. Goldluecke, and M. Magnor, "Reconstructing the Geometry of Flowing Water," *Proc. Int'l Conf. Computer Vision,* pp. 1055-1060, 2005.
[8] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active Contour Models," *Int'l J. Computer Vision,* vol. 1, pp. 321-331, 1988, citeseer.nj.nec.com/zhao01fast.html.
[9] V. Caselles, R. Kimmel, G. Sapiro, and C. Sbert, "Minimal Surfaces Based Object Segmentation," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 19, no. 4, pp. 394-398, Apr. 1997.
[10] H. Zhao, S. Osher, and R. Fedkiw, "Fast Surface Reconstruction Using the Level Set Method," *Proc. First IEEE Workshop Variational and Level Set Methods,* vol. 80, no. 3, pp. 194-202, 2001, citeseer.nj.nec.com/zhao01fast.html.
[11] N. Paragios and R. Deriche, "Geodesic Active Contours and Level Sets for the Detection and Tracking of Moving Objects," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 3, pp. 266-280, Mar. 2000.
[12] K.N. Kutukalos and S.M. Seitz, "A Theory of Shape by Space Carving," *Int'l J. Computer Vision,* vol. 38, no. 3, pp. 197-216, July 2000.
[13] H. Jin, S. Soatto, and A.J. Yezzi, "Multi-View Stereo beyond Lambert," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* vol. I, pp. 171-178, June 2003.
[14] B. Goldluecke and M. Magnor, "Weighted Minimal Hypersurfaces and Their Applications in Computer Vision," *Proc. European Conf. Computer Vision,* pp. 366-378, May 2004.
[15] R. Sharpe, *Differential Geometry.* Springer, 1997.
[16] S. Osher and J. Sethian, "Fronts Propagating with Curvature Dependent Speed: Algorithms Based on the Hamilton-Jacobi Formulation," *J. Computational Physics,* vol. 79, pp. 12-49, 1988.
[17] D. Chop, "Computing Minimal Surfaces Via Level Set Curvature Flow," *J. Computational Physics,* vol. 106, pp. 77-91, 1993.
[18] J.A. Sethian, *Level Set Methods and Fast Marching Methods,* second ed. Cambridge Univ. Press, 1999.
[19] A. Laurentini, "The Visual Hull Concept for Silhouette-Based Image Understanding," *IEEE Trans. Pattern Analysis and Machine Recognition,* vol. 16, no. 2, pp. 150-162, Feb. 1994.
[20] I. Ihrke and M. Magnor, "Image-Based Tomographic Reconstruction of Flames," *Proc. ACM Siggraph/Eurographics Symp. Computer Animation,* pp. 367-375, June 2004.
[21] L. Ahrenberg, I. Ihrke, and M. Magnor, "Volumetric Reconstruction, Compression and Rendering of Natural Phenomena from Multi-Video Data," *Proc. Int'l Workshop Volume Graphics,* June 2005.
[22] S.W. Hasinoff and K.N. Kutulakos, "Photo-Consistent 3D Fire by Flame-Sheet Decomposition," *Proc. Ninth IEEE Int'l Conf. Computer Vision (ICV '03),* pp. 1184-1191, 2003.
[23] H. Murase, "Surface Shape Reconstruction of a Nonrigid Transparent Object Using Refraction and Motion," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 14, no. 10, pp. 1045-1052, Oct. 1992.
[24] N.J.W. Morris and K.N. Kutulakos, "Dynamic Refraction Stereo," *Proc. Int'l Conf. Computer Vision,* pp. 1573-1580, 2005.
[25] H. Schultz, "Retrieving Shape Information from Multiple Images of a Specular Surface," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 16, no. 2, pp. 195-201, Feb. 1994.
[26] C. Pintavirooj, A. Romputtal, A. Ngamlamiad, W. Withayachumnankul, and K. Hamamoto, "Ultrasonic Refractive Index Tomography," *J. Winter School of Computer Graphics,* vol. 12, no. 2, pp. 333-339, Feb. 2004.
[27] A.V. Zvyagin, K.K.M.B.D. Silva, S.A. Alexandrov, T.R. Hillman, and J.J. Armstrong, "Refractive Index Tomography of Turbid Media by Bifocal Optical Coherence Refractometry," *Optics Express,* vol. 11, no. 25, pp. 3503-3517, Dec. 2003.
[28] K.N. Kutulakos and E. Steger, "A Theory of Refractive and Specular 3D Shape by Light-Path Triangulation," *Proc. Int'l Conf. Computer Vision,* pp. 1448-1455, 2005.
[29] A. Laurentini, "The Visual Hull Concept for Silhouette-Based Image Understanding," *IEEE Trans. Pattern Analysis and Machine Recognition,* vol. 16, no. 2, pp. 150-162, Feb. 1994.
[30] W. Matusik, C. Buehler, R. Raskar, S. Gortler, and L. McMillan, "Image-Based Visual Hulls," *Proc. ACM SIGGRAPH Conf.,* pp. 369-374, 2000.

Bastian Goldlücke received the diploma degree in mathematics from the University of Marburg, Germany, in 2001. He is a research assistant and PhD candidate at the Max Planck Institute Informatik in Saarbrücken, Germany. In 2002, he joined the Independent Research Group "Graphics-Optics-Vision" at the Max Planck Institute Informatik in Saarbrücken, Germany. His research interests include interactive rendering and 3D reconstruction. He is a student member of the IEEE.

Ivo Ihrke received the MSc degree in scientific computing from the Royal Institute of Technology, Stockholm, Sweden, in 2002. He is a research assistant and PhD candidate at the Max Planck Institute Informatik in Saarbrücken, Germany. He joined the Independent Research Group "Graphics-Optics-Vision" at the Max Planck Institute Informatik in Saarbrücken, Germany in 2003. His current work focuses on the reconstruction of time-varying natural phenomena like fire and water from given video footage. He is a student member of the IEEE.

Christian Linz received the diploma degree in computer science from the Saarland University, Germany, in 2005. He is a research assistant and PhD candidate in the Computer Graphics Lab of the Computer Science Department at the Technical University Braunschweig. His research interests comprise 3D reconstruction and video-based rendering.

Marcus Magnor received the BA (1995) and MS degrees (1997) in physics from the University of Würzburg and the University of New Mexico, respectively, and the PhD degree (2000) in electrical engineering from the Telecommunications Lab at the University of Erlangen. He heads the Computer Graphics Lab in the Computer Science Department at the Technical University Braunschweig. For his postgraduate studies, he joined the Computer Graphics Lab at Stanford University. In 2002, he established the Independent Research Group Graphics-Optics-Vision at the Max-Planck-Institut Informatik in Saarbrücken. There, he completed his habilitation and received the venia legendi in computer science from Saarland University in 2005. His research interests encompass the entire visual information processing pipeline, from image formation, acquisition, and analysis to image synthesis, display, and cognition. Recent and ongoing research topics include video-based rendering, 3D-TV, augmented vision, video editing, simulation of optical phenomena, as well as astrophysical visualization. He is a member of the IEEE.