DOI: 10.1111/j.1467-8659.2011.02058.x

Pacific Graphics 2011
Bing-Yu Chen, Jan Kautz, Tong-Yee Lee, and Ming C. Lin
(Guest Editors)

Volume 30 (2011), Number 7

SSD: Smooth Signed Distance Surface Reconstruction

F. Calakli and G. Taubin

School of Engineering, Brown University, Providence, RI, USA

Abstract

We introduce a new variational formulation for the problem of reconstructing a watertight surface defined by an implicit equation, from a finite set of oriented points; a problem which has attracted a lot of attention for more than two decades. As in the Poisson Surface Reconstruction approach, discretizations of the continuous formulation reduce to the solution of sparse linear systems of equations. But rather than forcing the implicit function to approximate the indicator function of the volume bounded by the implicit surface, in our formulation the implicit function is forced to be a smooth approximation of the signed distance function to the surface. Since an indicator function is discontinuous, its gradient does not exist exactly where it needs to be compared with the normal vector data. The smooth signed distance has approximate unit slope in the neighborhood of the data points. As a result, the normal vector data can be incorporated directly into the energy function without implicit function smoothing. In addition, rather than first extending the oriented points to a vector field within the bounding volume, and then approximating the vector field by a gradient field in the least squares sense, here the vector field is constrained to be the gradient of the implicit function, and a single variational problem is solved directly in one step. The formulation allows for a number of different efficient discretizations, reduces to a finite least squares problem for all linearly parameterized families of functions, and does not require boundary conditions. The resulting algorithms are significantly simpler and easier to implement, and produce results of quality comparable with state-of-the-art algorithms. An efficient implementation based on a primal-graph octree-based hybrid finite element-finite difference discretization, and the Dual Marching Cubes isosurface extraction algorithm, is shown to produce high quality crack-free adaptive manifold polygon meshes.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, surface, solid, and object representations G.1.2 [Numerical Analysis]: Approximation—Approximation of surfaces and contours

1. Introduction

The problem of reconstructing watertight surfaces from oriented point clouds has received an immense amount of attention since the mid 80's [BV91]. Oriented point clouds are nowadays usually obtained using optical measuring devices such as laser scanners and inexpensive structured lighting systems, by other computational means such as multi-view stereo reconstruction, and also result from large scale simulations. Dense oriented point clouds have become a pervasive surface representation in Computer Graphics [KB04].

The main challenges in this problem domain are: how to extrapolate to areas where the sampling is uneven, how to handle missing and noisy data, how to fill holes, and how to develop simple and efficient algorithms which gracefully scale up to very large data sets. As we show in section 9 the formulation introduced in this paper performs particularly well on unevenly sampled data sets. The primary contribution of this paper is a simple variational formulation of the problem developed from elementary geometric concepts. Various discretizations are proposed based on popular surface representations. The particular hybrid FE/FD discretization described in section 6 results in a simple algorithm which competes in quality and speed with the state-of-the-art methods.

The prior art in surface reconstruction methods is extensive. We discuss only some of the existing methods, and refer the reader to [SS05] for a survey on recent developments. Despite the long history, the area is still very active. One family of algorithms produces interpolating polygon meshes

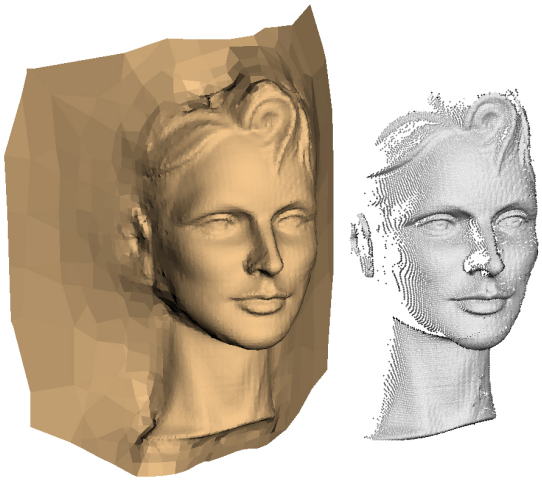


Figure 1: *A surface reconstructed by the proposed algorithm on an octree of depth 9, and the input point cloud.*

where all or some of the input points become vertices of the polygons [BMR*99]. Most of these algorithms are combinatorial in nature, are based on constructing Voronoi diagrams and Delaunay triangulations of the data points, and many come with guaranteed reconstruction quality [Dey07].

It is often more desirable to produce approximating surfaces rather than meshes supported on the input points. Various representations for the reconstructed surface have been considered. Since the implicit representation guarantees watertight surfaces, most surface reconstruction methods in this category produce implicit surfaces. Early work in this group, going back to the late 80's, include methods to fit algebraic surfaces to unoriented point clouds [Pra87, Tau91, MW90, BIW93, JF02], and PDE-based techniques [ZOMK00]. More recent methods include [Kaz05, KBH06, ACSTD07, MPS08] which reconstruct a binary indicator function; [HDD*92, CL96, BC02] which estimate a signed distance function; and [CBC*01, OBS03, ABCO*03, OBA*03, SOS04, FCOS05, MPS08] based on local function fitting and blending, or moving least squares. Most of these methods reduce the surface reconstruction problem to the solution of a large scale numerical optimization problem, and very often to the solution of a sparse linear system.

Even though the implicit functions produced by some of these methods have explicit analytic forms [CBC*01, OBS03, OBA*03, MPS08], since polygon meshes are usually needed for visualization or further computation purposes, the estimated implicit function is in the end evaluated on a regular grid of sufficient resolution, and an isosurface is extracted using an algorithm such as Marching Cubes [LC87], resulting in a polygonal mesh approximation of the desired level set. To reduce the computational and storage complexity, as well as the running time of the isosurface extraction algorithm, many of the proposed methods use octree-based schemes [KBH06, BKBH07, MPS08, ZGHG11], and so do we in some of the algorithms resulting from discretizing our continuous formulation.

Our formulation is most closely related to the Poisson Reconstruction approach [KBH06], which has become a leading contender in this field lately, among other reasons because of the high quality implementations made available by the authors. Streaming [BKBH07], parallel [BKBH09], and even GPU-based implementations [ZGHG11] are now available. We devote section 10 of this paper to establish similarities and differences between the two formulations. As a proof of concept, in section 7 we present an in-core CPU-based implementation. Figure 1 shows one example of surface reconstructed with the proposed method, along with the input point cloud. Note the good behavior near boundaries, holes, and uneven sampling. Efficient out-of core and GPU-based implementations of our algorithms are out of scope for this paper, and will be addressed in the near future.

2. Continuous Formulation

We are concerned with the problem of reconstructing a surface S defined by an implicit equation $S = \{x : f(x) = 0\}$, approximating a finite set of oriented points $\mathcal{D} = \{(p_1,n_1),\ldots,(p_N,n_N)\}$, where p_i is as surface location sample, and n_i is the corresponding surface normal sample oriented towards the outside of the object. If the function is continuous, then this surface is watertight (closed). Without loss of generality we will assume that $f(x) < 0$ inside and $f(x) > 0$ outside of the object. As a reference, in both FFT Surface Reconstruction [Kaz05] and Poisson Reconstruction [KBH06] the implicit function is forced to be the indicator function of the volume bounded by the surface S. This function is identically equal to zero outside, to one inside, and discontinuous exactly on S, as shown in a 2D example in Figure 2.

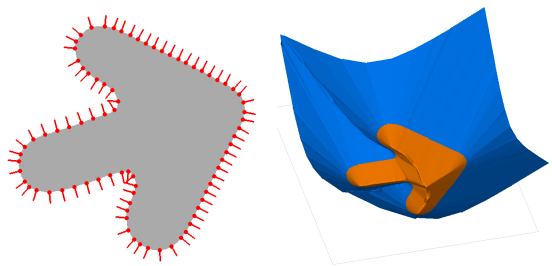


Figure 2: *A 2D oriented point cloud as a sample of a 2D curve represented as a level set of an indicator function(gray=1,white=0). The graph of the smooth signed distance function estimated by our algorithm from the same point cloud.*

Usually we are only interested in reconstructing the surface S within a bounded volume V, such as a cube containing all the data points, which may result in an open surface at the boundaries of the volume V, particularly when the points only sample one side of the object, there are holes in the data, or more generally if the points are not uniformly distributed along the surface of the solid object. Point clouds produced by 3D scanning systems are viewpoint dependent, and to cover the whole surface of the object several scans need to be registered and merged. We are not covering these topics here. Instead, we assume that point multiple scans have already been registered, and we seek a method that works well on regularly sampled as well as unevenly sampled data.

The gradient $\nabla f(x)$ of a function $f(x)$ with first order continuous derivatives is a vector field normal to the level sets of the function, and in particular to the surface S. If the gradient $\nabla f(x)$ does not vanish on the surface S, then S is a manifold surface with no singular points. Without loss of generality, we will further assume that the gradient field on the surface S is unit length, which allows us to directly compare the gradient of the function with the point cloud normal vectors.

Since the points p_i are regarded as samples of the surface S, and the normal vectors as samples of the surface normal at the corresponding points, for an interpolatory scheme the implicit function should satisfy $f(p_i) = 0$ and $\nabla f(p_i) = n_i$ for all the points $i = 1,\ldots,N$ in the data set. Interpolating schemes, which require parameterized families of functions with very large numbers of degrees of freedom, are not desirable in the presence of measurement noise. For an approximating scheme, which is what we are after, we require that these two interpolating conditions be satisfied in the least-squares sense. As a result, we consider the problem of minimizing the following data energy

$$\mathcal{E}_{\mathcal{D}}(f) = \lambda_0 \mathcal{E}_{\mathcal{D}_0}(f) + \lambda_1 \mathcal{E}_{\mathcal{D}_1}(f) \tag{1}$$

where

$$\mathcal{E}_{\mathcal{D}_0}(f) = \frac{1}{N}\sum_{i=1}^{N} f(p_i)^2 \quad \mathcal{E}_{\mathcal{D}_1}(f) = \frac{1}{N}\sum_{i=1}^{N} \|\nabla f(p_i) - n_i\|^2$$

and λ_0 and λ_1 are positive constants used to give more or less weight to each one of the two energy terms. The normalization by the number of points is introduced to make these two parameters independent of the total number of points. At this point this normalization has no effect, but plays a role after we add a third regularization term.

Depending on which family $\mathcal{F}$ of functions $f(x)$ is considered as the space of admissible solutions, the problem of minimizing the energy function $\mathcal{E}_{\mathcal{D}}(f)$ of equation (1) may or may not be well conditioned. In particular, this energy function does not specify how the function should behave away from the data points. Many parameterized families of functions have parameters highly localized in space, and the energy function of equation (1) may not constrain all the parameters. If the unconstrained parameters are arbitrarily set to zero, the estimated surface S may end up containing spurious components located far away from the data points, as we can observe in figures 5, 6, and 8. To address this problem we add the following regularization term to the energy function

$$\mathcal{E}_{\mathcal{R}}(f) = \frac{1}{|V|}\int_V \|Hf(x)\|^2\, dx \tag{2}$$

where $Hf(x)$ is the Hessian matrix of f, the 3×3 matrix of second order partial derivatives of $f(x)$, and the norm of the matrix is the Frobenius matrix norm, i.e., the sum of the squares of the nine elements of the matrix. The integral is over the volume V where we are interested in reconstructing the surface, $|V| = \int_V dx$ is the measure of this volume.

The total energy function in the proposed formulation is a weighted average of the data and regularization terms

$$\mathcal{E}(f) = \lambda_0 \mathcal{E}_{\mathcal{D}_0}(f) + \lambda_1 \mathcal{E}_{\mathcal{D}_1}(f) + \lambda_2 \mathcal{E}_{\mathcal{R}}(f)\,, \tag{3}$$

and λ_2 is another positive parameter. Increasing λ_2 with respect to λ_0 and λ_1 produces a smoother result. Recommended values for these parameters are discussed in section 9. Note that the three columns of the Hessian matrix $Hf(x)$ are the partial derivatives of the gradient $\nabla f(x)$ with respect to the three cartesian variables:

$$Hf(x) = \left[\frac{\partial \nabla f(x)}{\partial x_1}\ \frac{\partial \nabla f(x)}{\partial x_2}\ \frac{\partial \nabla f(x)}{\partial x_3}\right]$$

As a result, by forcing the square norm of the Hessian matrix to be close to zero, the regularization term makes the gradient of the function almost constant away from the data points. In the vicinity of the data points the data energy terms dominate the total energy, and make the function to approximate the signed distance function. Away from the data points the regularization energy term dominates the total energy and tends to make the gradient vector field $\nabla f(x)$ constant.

Section 6 describes the actual discretization that we propose. Sections 3, 4, and 5 are included primarily to motivate the approach.

3. Linearly Parameterized Families

In this section we are concerned with the existence and uniqueness of solution for the variational problem defined by the energy function $\mathcal{E}(f)$ of equation 3

$$\frac{\lambda_0}{N}\sum_{i=1}^{N} f(p_i)^2 + \frac{\lambda_1}{N}\sum_{i=1}^{N} \|\nabla f(p_i) - n_i\|^2 + \frac{\lambda_2}{|V|}\int_V \|Hf(x)\|^2\, dx$$

We restrict the analysis here to families of functions linearly parameterized by a finite number of parameters, because in most of these cases the problem has a unique solution, which can be computed by solving a system of linear equations.

By a linearly parameterized family of functions $\mathcal{F}$ we mean a finite dimensional vector space of functions. Explic-

itly, a member of this family can be written as a linear combination of certain basis functions

$$f(x) = \sum_{\alpha \in \Lambda} f_\alpha \phi_\alpha(x) \tag{4}$$

where α denotes an index which belongs to a finite set Λ, say with K elements, $\phi_\alpha(x)$ is a basis function, and f_α is the corresponding coefficient. The K basis functions are chosen in advance, and are not regarded as variables in the problem. After selecting a particular order for the K indices, the function can also be written as an inner product of two K-dimensional column vectors

$$f(x) = \Phi(x)^t F \tag{5}$$

where $F = (f_\alpha : \alpha \in \Lambda)$ and $\Phi(x) = (\phi_\alpha(x) : \alpha \in \Lambda)$ are K-dimensional column vectors.

4. Smooth Basis Functions

Even though this is not what we advocate in this paper, for the sake of developing the ideas, let's first consider the case of basis functions with continuous derivatives up to second order, so that the gradient exists and the second order derivatives are integrable in V. Radial basis functions [CBC*01], compactly supported basis functions [OBS03, OBA*03], polynomials [Pra87, Tau91, KTFC01, JF02] trigonometric polynomials [Kaz05], wavelets [MPS08], B-splines [KBH06], and many other popular families of functions are included in this category.

Restricting the energy function $\mathcal{E}(f)$ of equation 3 to one of these families results in a positive-semidefinite, non-homogeneous, quadratic function $F^t A F - 2b^t F + c$ in the K-dimensional parameter vector F. Except for very singular configurations, the matrix A is usually symmetric and positive definite, and the resulting finite dimensional minimization problem has a unique minimum. The global minimum is determined by solving the system of linear equations $AF = b$. To complete the process we need to compute the coefficients $A_{\alpha\beta}$ and b_α of the matrix A and vector b, and then select an appropriate algorithm to solve the linear equations.

Note that the three terms of the energy function $\mathcal{E}(f)$ contribute to the matrix A, but since the first data term and regularization terms are homogeneous quadratic functions of F, only the second data term contributes to the vector b.

Keeping in mind the parameterization described by equation (5), the contribution of the first data term $\mathcal{E}_{\mathcal{D}_0}(f)$ to the matrix A is

$$A^0_{\alpha\beta} = \frac{1}{N} \sum_{i=1}^{N} \phi_\alpha(p_i)\phi_\beta(p_i) \tag{6}$$

Since

$$\|\nabla f(p_i) - n_i\|^2 = \|\nabla f(p_i)\|^2 - 2n_i^t \nabla f(p_i) + \|n_i\|^2$$

the contribution of the second data term $\mathcal{E}_{\mathcal{D}_1}(f)$ to the matrix A is

$$A^1_{\alpha\beta} = \frac{1}{N} \sum_{i=1}^{N} \nabla\phi_\alpha(p_i)^t \nabla\phi_\beta(p_i) \tag{7}$$

and to the vector b is

$$b^1_\alpha = \frac{1}{N} \sum_{i=1}^{N} n_i^t \nabla\phi_\alpha(p_i) \tag{8}$$

The contribution of the second data term to the constant c is equal to λ_1, because the normal vectors are assumed to be unit length, but this is irrelevant since an additive constant in the energy function does not play any role in the optimization.

The contribution of the regularization term $\mathcal{E}_{\mathcal{R}}(f)$ to the matrix A is

$$A^2_{\alpha\beta} = \frac{1}{|V|} \int_V \langle H\phi_\alpha(x), H\phi_\beta(x) \rangle \, dx \tag{9}$$

where $\langle .,. \rangle$ is the inner product associated with the Frobenius norm

$$\langle H\phi_\alpha(x), H\phi_\beta(x) \rangle = \sum_{i=1}^{3} \sum_{j=1}^{3} \frac{\partial^2 \phi_\alpha(x)}{\partial x_i \partial x_j} \frac{\partial^2 \phi_\beta(x)}{\partial x_i \partial x_j}$$

Finally, we have $A = \lambda_0 A^0 + \lambda_1 A^1 + \lambda_2 A^2$ and $b = \lambda_1 b^1$. While accumulating the contributions A^0, A^1 and b^1 of the two data terms requires the evaluation of the basis functions and their derivatives on all the data points, for fixed basis functions which are known in analytic form, the contribution A^2 of the regularization term can be precomputed and saved in a static table, which can be hardcoded in the implementation. If the basis functions are constructed as functions of the data set, then the integrals (9) have to be evaluated during the program execution. Numerical approximations of these integrals are usually acceptable, although for many of the linear families mentioned above, these integrals can be evaluated analytically in closed form.

5. Sparsity

If the number of parameters K is small, as in the case of polynomial basis functions, solving the linear system $AF = b$ is straightforward. One of many direct linear solvers for matrices represented as arrays can be used, including the QR method. However, in most surface reconstruction problems, a large number of degrees of freedom, typically $O(N)$, is required for acceptable quality results. For example, the radial basis functions used in [CBC*01] and trigonometric polynomials [Kaz05] are non-zero almost everywhere, and result in a dense matrix A. This is not only a problem in terms of storage, but also in terms of speed, since out-of-core storage of matrix and vector may be required along with simple and slow iterative schemes. On the other hand, the matrix A is sparse when the basis functions are compactly supported, so that the support of only a small number of basis functions overlaps at any given point in the volume V. This property

also results in many of the integrals shown in equation (9) associated with the regularization term being zero: if the supports of ϕ_α and ϕ_β do not intersect, then the integral $A^2_{\alpha\beta}$ is equal to zero.

For example, the linear families used in the following methods [OBS03, OBA*03, MPS08, KBH06], which all use compactly supported basis functions, would be appropriate for this approach. However, since computing the integrals associated with the regularization term can result in significant implementation complexity, we are not advocating for any of these families. Instead, we propose a hybrid FE/FD discretization with discontinuous gradient, which results in a much simpler regularization term.

6. Discretization with Discontinuous Gradient

The approach described in the previous section can be extended to the case of basis functions with derivatives continuous only up to first order, and with second order derivatives which are integrable in V in the sense of generalized functions. As long as the discretized function $f(x)$, gradient $\nabla f(x)$, the Hessian $Hf(x)$ can be written as homogeneous linear functions of the same parameters F, the problem still reduces to the solution of linear equations $AF = b$. In fact, even independent discretizations for $f(x)$, $\nabla f(x)$ and $Hf(x)$ can be used, again, as long as all these expressions are homogeneous linear functions of the same parameters F. We propose in this section one such hybrid discretization, where a finite element discretization is used for the function $f(x)$, but finite differences discretizations are used for gradient $\nabla f(x)$ and the Hessian $Hf(x)$. The results presented in section 9, show that this discretization turns out to be particularly simple to implement, is is applicable to adaptive grids, and produces results of high quality at high speed.

To simplify the notation, we first consider a regular hexahedral grid partition of space. We assume that the volume V is a unit cube $V = [0,1] \times [0,1] \times [0,1]$, and that each of the three axes is split into M equally long segments, resulting in M^3 hexahedral cells and $(M+1)^3$ vertices. In the general case a bounding box for the data points has to be computed first, perhaps expanded by a certain amount to prevent boundary-related artifacts, and all the coordinates need to be scaled. If the scaling is anisotropic in the three coordinates, then the normal vector coordinates need to be scaled as well and then renormalized to unit length.

In this discretization, the indices $\alpha \in \Lambda$ are the grid vertex multi-indices $\alpha = (i,j,k)$, for $0 \le i,j,k \le M$. The coefficients f_α are then the values of the function $f(x)$ at the grid vertices $p_\alpha = (i/M, j/M, k/M)$. Since the result of this computation is a scalar field defined on the vertices of a regular hexahedral grid, the construction of an output isosurface using Marching Cubes [LC87] does not require any further function evaluations. The grid cells C_α are also indexed by multi-indices $\alpha = (i,j,k)$, except that in this case $0 \le i,j,k < M$. The indices of the eight vertices of a cell C_α are depicted in Figure 3. and a similar relation can be found to identify the (up to eight) cells which contain a given vertex as a corner.

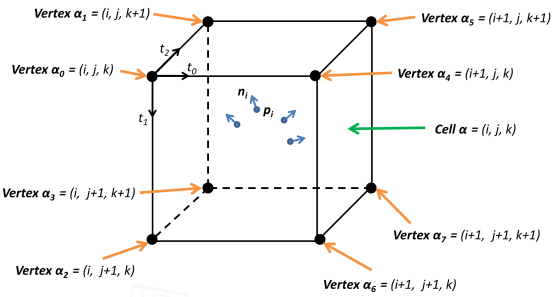


Figure 3: *Vertices of an hexahedral cell*

We use finite differences discretizations for the gradient and Hessian. Within each cell C_α we use trilinear interpolation as the discretization of the function $f(x)$. This is a finite element discretization. Determining which cell C_α each point p_i belongs to reduces to quantization. If $p_i \in C_\alpha$ then

$$f(p_i) \approx \sum_{h=0}^{7} w_h f_{\alpha_h} \tag{10}$$

where $w_0, \ldots, w_7$ are the trilinear coordinates of p_i.

Within each cell C_α we use a constant average of finite differences discretization, denoted $\nabla_\alpha f$, for the gradient $\nabla f(x)$

$$\frac{1}{4\Delta_\alpha} \begin{pmatrix} f_{\alpha_4} - f_{\alpha_0} + f_{\alpha_5} - f_{\alpha_1} + f_{\alpha_6} - f_{\alpha_2} + f_{\alpha_7} - f_{\alpha_3} \\ f_{\alpha_2} - f_{\alpha_0} + f_{\alpha_3} - f_{\alpha_1} + f_{\alpha_6} - f_{\alpha_4} + f_{\alpha_7} - f_{\alpha_5} \\ f_{\alpha_1} - f_{\alpha_0} + f_{\alpha_3} - f_{\alpha_2} + f_{\alpha_5} - f_{\alpha_4} + f_{\alpha_7} - f_{\alpha_6} \end{pmatrix} \tag{11}$$

where Δ_α is the side length of cell C_α. If the dimensions of the bounding box are different along each of the three coordinate axes, then different lengths must be used to normalize each of the three coordinates of $\nabla_\alpha f$. If $p_i \in C_\alpha$ then

$$\nabla f(p_i) \approx \nabla_\alpha f \tag{12}$$

The piecewise constant gradient leads to zero second derivatives within each cell. In this case the square norm of the Hessian matrix $Hf(x)$ is a generalized function supported on the square faces shared by neighboring voxels (as Dirac's δ "function"), and the integral over the volume V reduces to a finite sum over the faces. Hence, we only need an approximation to $\|Hf(x)\|^2$ at the faces shared by cells. If C_α and C_β are two such cells that share a face, then we use finite differences of discrete gradients

$$H_{\alpha\beta} f = \frac{1}{\Delta_{\alpha\beta}} \left(\nabla_\alpha f - \nabla_\beta f \right) \tag{13}$$

where $\Delta_{\alpha\beta}$ is the Euclidean distance between the centers of

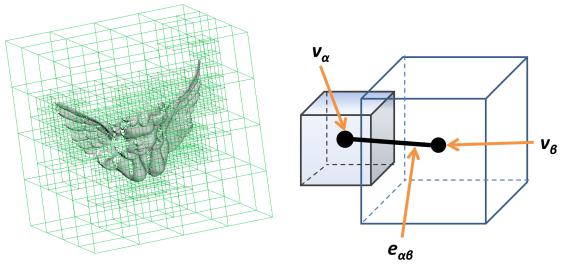
the cells. As a result, the integral in the regularization term becomes a sum over all the neighboring cell pairs

$$\mathcal{E}_{\mathcal{R}}(f) \approx \frac{1}{|V|} \sum_{(\alpha,\beta)} |V|_{\alpha\beta} \, \| H_{\alpha\beta} f \|^2 \qquad (14)$$

where $|V|_{\alpha\beta}$ is the area of the face shared by cells C_α and C_β, and $|V| = \sum_{(\alpha,\beta)} |V|_{\alpha\beta}$.

7. Octree-Based Implementation

As in many of the prior works cited above, a regular grid formulation results in high storage requirements and slow algorithms. Also, as in many of the cited works, we adaptively construct an octree based on the location of the data points. In the recursive subdivision algorithm, a cell is split if the number of points contained is larger than a prespecified value, and the cell depth does not exceed another prespecified value. The data set is recursively partitioned into subsets of points $\mathcal{D}_\alpha$ associated with cells C_α. For octrees the multi-indices $\alpha = (L,i,j,k)$ must be augmented with a level (or depth) value L, both for cells and vertices, where $0 \le L \le L_{\mathrm{MAX}}$. The correspondence between indices α and cells is 1-1, but not for vertices, since (L,i,j,k) and $(L+1,2i,2j,2k)$ refer to the same vertex. However we use these relations in our implementation, along with hashing algorithms, to efficiently traverse the primal and dual graphs of an octree. Figure 4(a) shows an oriented point cloud and the primal graph of an octree constructed in this manner.



(a) (b)

Figure 4: *(a) The primal graph of an octree, and (b) an illustration of dual edges.*

The finite-differences discretization presented in the previous section for regular hexahedral grids extends in a natural way to octree grids, without any changes. There is one function value f_α associated with each primal vertex of the octree. As a result, within each hexahedral cell we use trilinear interpolation to discretize the function $f(x)$, and the constant gradient $\nabla_\alpha f$ of equation 11. Finally, we use the discretization of the Hessian $Hf(x)$ of equation 13, where the dual edge weight $|V|_{\alpha\beta}$ is now the area of the common face. Figure 4(b) illustrates this concept.

The formulation described above reduces to the solution of a linear system $AF = b$. The data set is partitioned during the recursive subdivision process used to construct the octree. The elements of the matrix A and vector b are accumulated by traversing the list of octree leaf cells. For each point in an non-empty cell C_α new terms are added to $A_{\alpha_i\alpha_j}$ and to b_{α_i}, where α_i and α_j are indices of the eight vertices of the cell C_α. To accumulate the contribution of the regularization term the octree dual graph is traversed. For each dual edge connecting cells C_α and C_β, terms are added to the elements $A_{\alpha_i\beta_j}$ of the matrix A, where α_i is an index of a vertex of cell C_α, and β_j is an index of a vertex of cell C_β. After all the coefficients are accumulated, for reasonably sized problems one can use a direct solver. Instead, since octrees inherently have multiresolution structure, we follow an iterative cascading multigrid approach. We start by solving the problem on a much coarser level than desired using a simple conjugate gradient solver. Then use it to initialize the solution at the next level, which is then refined with the conjugate gradient solver.

8. Primal and Dual Marching Cubes

Once the coefficients f_α are determined, we are faced with the final problem of constructing a polygonal approximation of the isolevel zero of the discretized implicit function. Marching Cubes [LC87] can be implemented in such a way that it can be applied to any volumetric grid composed of hexahedral cells, and in particular, to regular voxel grids and to octree grids. When it is applied to a scalar field defined on the primal vertices of an octree, a mesh with so-called "cracks" typically result. Since the dual grid of an octree hexahedral grid is composed of cells which can be regarded as hexahedral cells with collapsed edges, we use the same general implementation of Marching Cubes to instantiate the Dual Marching Cubes algorithm [SW05]. Dual Marching Cubes requires values of the implicit function at the vertices of the dual graph, i.e., at the centroids of the primal octree hexahedral cells. The results of the optimization algorithm for the proposed finite differences discretization is defined on the vertices of the primal graph. An implicit function value is computed for the centroid of each primal cell by averaging the values associated with the eight cell corners.

9. Results

We compare the proposed Smooth Signed Distance (SSD) surface reconstruction method with Multi-Level Partition of Unity Implicits (MPU) [OBA*03], Poisson Surface Reconstruction (Poisson) [KBH06], and Streaming Surface Reconstruction Using Wavelets (D4 Wavelets) [MPS08]. We present the reconstruction results in terms of speed, memory, efficiency, and accuracy. We show that, in terms of the quality of reconstruction, our method outperforms these other methods even on data sets associated with extreme complications: non-uniform sampling, sensor noise, and missing data. In addition, we observe that its performance is still

comparable to in-core CPU implementations of the alternative methods. We consider efficient streaming and GPU based implementations out-of-scope for this paper, but we plan to address these issues in near future.

We apply our algorithm to point clouds retrieved either from a single partial scan, or from multiple scans that have been aligned with respect to a common reference frame. We assume that each point is associated with a surface normal vector. In the absence of this information, we efficiently estimate the normals from partially triangulated scans, and determine their orientations from the viewpoint direction to the scanner. If this process introduces significant errors, one can alternatively estimate a point sample's normal vector from the positions of the local neighbors using local fitting. Our initial test case is the head of Michalengelo's David raw dataset of 1 million samples. This dataset is assembled from many range images acquired by a laser scanner. Since the points are almost uniformly sampled, we ignore the noisy triangles provided by the scanner, and use local surface fitting to estimate a sample's normal. As a result, we have uniformly distributed points with accurate normals. Figure 5 compares different reconstructions. All methods operate on the same data set at a maximal octree depth of 8, and produce surfaces of comparable quality. Table 1 summarizes the performance characteristics of each algorithm. Our second test case is the left eye of Michalengelo's David raw dataset with 187,526 samples. This dataset is cropped from the head of Michalengelo's David raw dataset. We estimated normals from the noisy triangles provided by the scanner. The resulting oriented point set has uniformly distributed points with many *inaccurate* normals. Figure 6 compares different reconstructions at a maximal octree depth of 10. Since the data is noisy, MPU results in reconstruction with spurious surface sheets. Poisson, D4 Wavelets, and our SSD Reconstruction all accurately reconstruct the surface, although the one reconstructed with Poisson is a bit smoother. Our third test case is composed of 100,000 oriented point samples obtained by sampling a virtual horse model with a sampling density proportional to curvature, giving a set of *non-uniformly* distributed points with accurate normals. Figure 7 compares different reconstructions at a maximal octree depth of 9. The D4 Wavelets Reconstruction fails to reconstruct the surface accurately. MPU, Poisson, and our SSD Reconstruction all accurately reconstruct the surface with subtle differences. Our final test case is the Chiquita model raw dataset comprising 705,375 samples. This dataset was acquired by scanning the real world object with an inexpensive 3D structured lighting system [CLS*06]. It is an example of *non-uniformly* distributed points with *inaccurate* normals. Figure 8 compares different reconstructions at a maximal octree depth of 9. MPU produces spurious surface sheets. Although D4 Wavelets performs much better, it can not fill in gaps reasonably. Poisson, and our SSD Reconstruction both result in pleasing surfaces with subtle differences.

Method	Time (Sec)	Memory (MB)	Polygons
MPU	27	148	378925
Poisson	43	283	319989
D4	17	63	365974
SSD	72	264	346652

Table 1: *The running time, the peak memory usage, and the number of triangles in the reconstructed surface of the David's head generated with different methods.*

To evaluate the numerical accuracy of the reconstruction results we follow the same strategy as in [MPS08]. We first sample points from a known model, then reconstruct surfaces using each method with this point set. We then compute the Hausdorff distance between each reconstructed model and the known model using Metro tool [CRS96]. Table 2 summarizes the result. Note that, in all cases, our SSD reconstruction recovers the surfaces with higher degree of accuracy than the Poisson reconstruction. And in two cases (Horse, and Igea) it also outperforms D4 Wavelets.

Model	MPU	Poisson	D4	SSD
Armadillo	1.0000	0.4617	0.2383	0.3514
Dragon	0.8779	1.0000	0.5301	0.6810
Horse	0.0752	0.0827	1.0000	0.0551
Igea	1.0000	0.7761	0.5701	0.4018

Table 2: *Hausdorff distance between real surfaces and reconstructed surfaces. Each row is normalized by the maximum error to provide relative comparison (lower is better).*

10. Relation to Poisson Surface Reconstruction

As in our formulation, the Poisson Surface Reconstruction approach [KBH06] looks for an implicit function which should be zero at the data points, and such that its gradient evaluated at the data points equals the corresponding normal vectors. The set of oriented data points $\mathcal{D}$ is regarded as a sparse sampling of a continuous vector field defined on the bounding volume V. As a result, the vector field samples are first extended to a continuous three dimensional vector field $v(x)$ defined on the volume V. The implicit function $f(x)$ is subsequently recovered integrating the vector field. i.e, solving the functional equation $\nabla f = v$. But since not every vector field is the gradient of a function the problem may or may not have solution. The necessary and sufficient condition for a smooth vector field to be the gradient of a function is that its curl be identically zero, or equivalently, that the integral of the vector field along any closed curve be equal to zero. Since these conditions are difficult to impose as constraints, an alternative least-squares solution is proposed, where the functional equation is approximately solved by minimizing

Figure 5: *Input point cloud (leftmost) of David's head, and reconstructions using (from second-left to right) MPU, Poisson, D4 Wavelets, and our SSD reconstruction.*

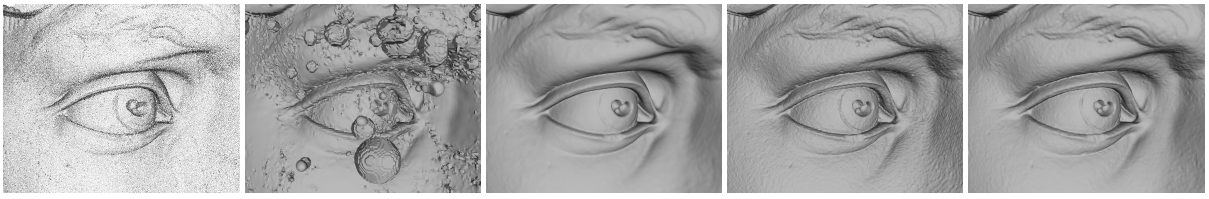


Figure 6: *Input point cloud (leftmost) of David's eye, and reconstructions using (from second-left to right) MPU, Poisson, D4 Wavelets, and our SSD reconstruction.*

the integral equation

$$\int_V \|\nabla f(x) - v(x)\|^2 \, dx \tag{15}$$

with respect to the function $f(x)$, while keeping the vector field constant. Solving this variational problem is equivalent to solving the classical Poisson equation $\Delta f = \nabla \cdot v$ in V. From the linear algebra point of view, the solution of this problem is the orthogonal projection of the vector field v onto the subspace of gradient vector fields, with the solution represented in parametric form. The solution is usually unique modulo an additive constant f_0, which is then determined by minimizing the fitting error $\sum_{i=1}^{N}(f(p_i) - f_0)^2$ with $f(x)$ being a minimizer of the integral equation (15). The solution to this last problem is the average of the function on the data points $f_0 = \frac{1}{N}\sum_{i=1}^{N} f(p_i)$. Note the the whole process involves three minimization problems rather than one, performed in a squential manner.

If we introduce the vector field v as an additional variable in our formulation, and we remove the first data term from the total energy of equation (3) we obtain a vector field energy

$$\frac{\lambda_1}{N}\sum_{i=1}^{N} \|v(p_i) - n_i\|^2 + \frac{\lambda_2}{|V|}\int_V \|Dv(x)\|^2 \, dx \tag{16}$$

which can be minimized independently. In this equation Dv denotes the Jacobian matrix of the vector field v. Note that if $\nabla f = v$, then $Hf = Dv$. The result is a vector field defined on the volume V which extends the oriented points, as in the Poisson Surface Reconstruction approach. Then the same other two steps, of solving the Poisson equation, and estimating the optimal isolevel by minimizing the data term originally removed from the total energy of equation (3) can be performed. In fact, this approach was shown to work in [ST05] to solve the surface reconstruction problem, for the function and vector field discretized on a regular voxel grid, and with the integral term of the energy discretized by finite differences.

The most important difference is that in both the FFT [Kaz05] and Poisson [KBH06] surface reconstruction approaches the problem is formulated for an indicator implicit function. The choice of an indicator function for the implicit function creates complications, since the gradient of the implicit function does not exist in the traditional sense at the sample points, where the function is discontinuous. As a result the two methods estimate a low-pass filtered version of the indicator function. It is noted in [KBH06] that in practice, care should be taken in choosing the smoothing filter, and that for non-uniformly distributed point samples the smoothing kernel width needs to be adapted. The adaptation process is only explained within the concept of the octree-based implementation. It is also reported in a recent surface reconstruction benchmark study [BLN*11] that these two methods tend to oversmooth the data, as we can also observe in some of the results presented herein. In the FFT approach, a truncated Fourier expansion automatically provides an ideal low-pass filtered version of the indicator function, but the restriction to a regular voxel grid discretization

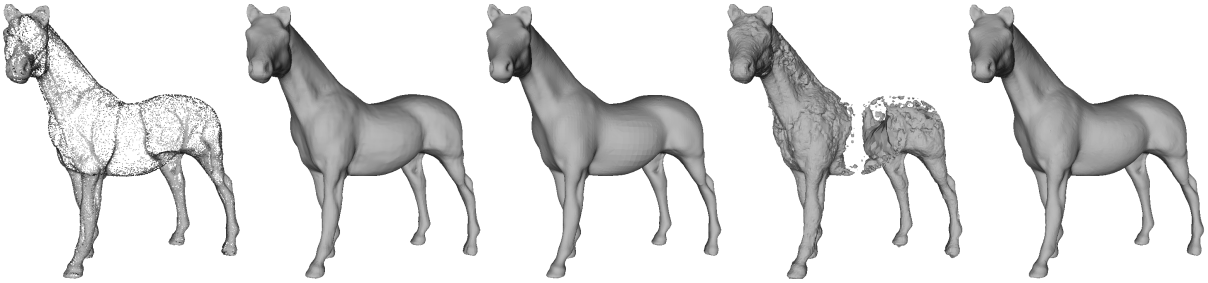


Figure 7: *Input point cloud (leftmost) of the Horse model, and reconstructions using (from second-left to right) MPU, Poisson, D4 Wavelets, and our SSD reconstruction.*

Figure 8: *Input point cloud (leftmost) of the Chiquita model, and reconstructions using (from second-left to right) MPU, Poisson, D4 Wavelets, and our SSD reconstruction.*

limits its apeal, as in the case of [ST05]. In the Poisson Surface Reconstruction approach implemented on an octree, the vector field is constructed as a linear combination of compactly supported smooth basis functions associated with the octree cells. As a result, the function obtained by solving the Poisson equation is equally smooth. Since the vector field should approximate the low-pass filtered gradient of the indicator function, it is forced to be zero away from the data points.

In the end, the proposed approach provides an interesting alternative to Poisson Reconstruction and other state-of-the-art methods, which is much simpler to implement and performs particularly well on unevenly sampled data sets, at comparable computational cost.

11. Conclusion

In this paper we have introduced a new variational formulation for the problem of reconstructing a watertight surface defined by an implicit equation from a finite set of oriented points. As in other surface reconstruction approaches discretizations of this continuous formulation reduce to the solution of sparse least-squares problems. Rather than forcing the implicit function to approximate the indicator function of the volume bounded by the surface, in our formulation the implicit function is a smooth approximation of the signed distance function to the surface. We introduced a very simple hybrid FE/FD discretization, which together with an octree partitioning of space, and the Dual Marching Cubes algorithm produces accurate and adaptive meshes. The software implementation and data sets used to create the figures shown in this paper are available for download from [CT11].

12. Acknowledgments

The authors would like to thank the members of the Digital Michelangelo project, in particular, Szymon Rusinkiewicz and Benedict Brown, for providing non-rigid body aligned David head data. This material is based upon work supported by the National Science Foundation under Grant No. IIS-0808718, CCF-0729126, and CCF-0915661.

References

[ABCO*03] ALEXA M., BEHR J., COHEN-OR D., FLEISHMAN S., LEVIN D., SILVA C.: Computing and Rendering Point Set Surfaces. *IEEE Transactions on Visualization and Computer Graphics* (2003), 3–15. 2

[ACSTD07] ALLIEZ P., COHEN-STEINER D., TONG Y., DESBRUN M.: Voronoi-based variational reconstruction of unoriented point sets. In *Proceedings of the fifth Eurographics symposium on Geometry processing* (2007), Eurographics Association, pp. 39–48. 2

[BC02] BOISSONNAT J., CAZALS F.: Smooth surface reconstruction via natural neighbour interpolation of distance functions. *Computational Geometry 22*, 1 (2002), 185–203. 2

[BIW93] BAJAJ C., IHM I., WARREN J.: Higher-Order Interpolation and Least-Squares Approximation Using Implicit Algebraic Surfaces. *ACM Transactions on Graphics (TOG) 12*, 4 (1993), 327–347. 2

[BKBH07] BOLITHO M., KAZHDAN M., BURNS R., HOPPE H.: Multilevel Streaming for Out-Of-Core Surface Reconstruction. In *Proceedings of the 5th. Eurographics Symposium on Geometry Processing* (2007), Eurographics Association, pp. 69–78. 2

[BKBH09] BOLITHO M., KAZHDAN M., BURNS R., HOPPE H.: Parallel Poisson Surface Reconstruction. In *Proceedings of the 5th International Symposium on Advances in Visual Computing* (2009), Springer, pp. 678–689. 2

[BLN*11] BERGER M., LEVINE J., NONATO L., TAUBIN G., SILVA C.: *An End-to-End Framework for Evaluating Surface Reconstruction*. Sci technical report, University of Utah, Jan. 2011. http://www.cs.utah.edu/~bergerm/recon_bench/. 8

[BMR*99] BERNARDINI F., MITTLEMAN J., RUSHMEIER H., SILVA C., TAUBIN G.: The Ball-Pivoting Algorithm for Surface Reconstruction. *IEEE Transactions on Visualization and Computer Graphics 5*, 4 (1999), 349–359. 2

[BV91] BOLLE R., VEMURI B.: On Three-Dimensional Surface Reconstruction Methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (1991), 1–13. 1

[CBC*01] CARR J., BEATSON R., CHERRIE J., MITCHELL T., FRIGHT W., MCCALLUM B., EVANS T.: Reconstruction and Representation of 3D Objects with Radial Basis Functions. In *Proceedings of ACM Siggraph* (2001), ACM, pp. 67–76. 2, 4

[CL96] CURLESS B., LEVOY M.: A Volumetric Method for Building Complex Models from Range Images. In *Proceedings of ACM Siggraph* (1996), ACM, pp. 303–312. 2

[CLS*06] CRISPELL D., LANMAN D., SIBLEY P., ZHAO Y., TAUBIN G.: Beyond silhouettes: Surface reconstruction using multi-flash photography. In *Third International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT'2006)* (2006), IEEE, pp. 405–412. 7

[CRS96] CIGNONI P., ROCCHINI C., SCOPIGNO R.: *Metro: measuring error on simplified surfaces*. Tech. rep., Paris, France, France, 1996. 7

[CT11] CALAKLI F., TAUBIN G.: Smooth signed distance surface reconstruction. http://mesh.brown.edu/ssd, 2011. Software and data download. 9

[Dey07] DEY T.: *Curve and Surface Reconstruction: Algorithms with Mathematical Analysis*. Cambridge Univ Pr, 2007. 2

[FCOS05] FLEISHMAN S., COHEN-OR D., SILVA C.: Robust Moving Least-Squares Fitting With Sharp Features. *ACM Transactions on Graphics (TOG) 24*, 3 (2005), 544–552. 2

[HDD*92] HOPPE H., DEROSE T., DUCHAMP T., MCDONALD J., STUETZLE W.: Surface Reconstruction from Unorganized Points. In *Proceedings of ACM Siggraph* (1992), vol. 26, pp. 71–71. 2

[JF02] JÜTTLER B., FELIS A.: Least-Squares Fitting of Algebraic Spline Surfaces. *Advances in Computational Mathematics 17*, 1 (2002), 135–152. 2, 4

[Kaz05] KAZHDAN M.: Reconstruction of Solid Models from Oriented Point Sets. In *Proceedings of the 3rd. Eurographics symposium on Geometry processing* (2005), Eurographics Association, p. 73. 2, 4, 8

[KB04] KOBBELT L., BOTSCH M.: A Survey of Point-Based Techniques in Computer Graphics. *Computers & Graphics 28*, 6 (2004), 801–814. 1

[KBH06] KAZHDAN M., BOLITHO M., HOPPE H.: Poisson surface reconstruction. In *Proceedings of the fourth Eurographics Symposium on Geometry Processing* (2006), Eurographics Association, pp. 61–70. 2, 4, 5, 6, 7, 8

[KTFC01] KANG K., TAREL J.-P., FISHMAN R., COOPER D.: A linear dual-space approach to 3d surface reconstruction from occluding contours using algebraic surfaces. In *In International Conference on Computer Vision* (2001), pp. 198–204. 4

[LC87] LORENSEN W. E., CLINE H. E.: Marching cubes: A high resolution 3d surface construction algorithm. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques* (1987), SIGGRAPH '87, pp. 163–169. 2, 5, 6

[MPS08] MANSON J., PETROVA G., SCHAEFER S.: Streaming Surface Reconstruction Using Wavelets. *Computer Graphics Forum 27*, 5 (2008), 1411–1420. 2, 4, 5, 6, 7

[MW90] MOORE D., WARREN J.: Approximation of dense scattered data using algebraic surfaces. In *Proceedings of the Twenty-Fourth Annual Hawaii International Conference on on System Sciences* (1990), vol. 1, IEEE, pp. 681–690. 2

[OBA*03] OHTAKE Y., BELYAEV A., ALEXA M., TURK G., SEIDEL H.: Multi-Level Partition of Unity Implicits. *ACM Transactions on Graphics (TOG) 22*, 3 (2003), 463–470. 2, 4, 5, 6

[OBS03] OHTAKE Y., BELYAEV A., SEIDEL H.: A Multi-Scale Approach to 3D Scattered Data Interpolation with Compactly Supported Basis Functions. In *International Conference on Shape Modeling and Applications* (2003), pp. 153–161. 2, 4, 5

[Pra87] PRATT V.: Direct Least-Squares Fitting of Algebraic Surfaces. In *Proceedings of ACM Siggraph* (1987), pp. 145–152. 2, 4

[SOS04] SHEN C., O'BRIEN J., SHEWCHUK J.: Interpolating and Approximating Implicit Surfaces from Polygon Soup. In *Proceedings of ACM Siggraph* (2004), pp. 896–904. 2

[SS05] SCHALL O., SAMOZINO M.: Surface from scattered points: A brief survey of recent developments. In *1st International Workshop on Semantic Virtual Environments* (Villars, Switzerland, 2005), Falcidieno B., Magnenat-Thalmann N., (Eds.), MIRALab, pp. 138–147. 1

[ST05] SIBLEY P., TAUBIN G.: Vectorfield Isosurface-Based Reconstruction from Oriented Points. In *ACM Siggraph 2005 Sketches* (2005), ACM, p. 29. 8, 9

[SW05] SCHAEFER S., WARREN J.: Dual marching cubes: Primal contouring of dual grids. *Computer Graphics Forum 24*, 2 (2005), 195–201. 6

[Tau91] TAUBIN G.: Estimation of Planar Curves, Surfaces, and Nonplanar Space Curves Defined by Implicit Equations with Applications to Edge and Range Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (1991), 1115–1138. 2, 4

[ZGHG11] ZHOU K., GONG M., HUANG X., GUO B.: Data-Parallel Octrees for Surface Reconstruction. *IEEE Transactions on Visualization and Computer Graphics 17*, 5 (May 2011), 669–681. 2

[ZOMK00] ZHAO H., OSHER S., MERRIMAN B., KANG M.: Implicit and Nonparametric Shape Reconstruction from Unorganized Data Using a Variational Level Set Method. *Computer Vision and Image Understanding 80*, 3 (2000), 295–314. 2