A Level-Set Approach to 3D Reconstruction From Range Data

Ross T. Whitaker
School of Computing
University of Utah
Salt Lake City, UT 84112-9205
email: whitaker@cs.utah.edu

Abstract

This paper presents a method that uses the level sets of volumes to reconstruct the shapes of 3D objects from range data. The strategy is to formulate 3D reconstruction as a statistical problem: find that surface which is mostly likely, given the data and some prior knowledge about the application domain. The resulting optimization problem is solved by an incremental process of deformation. We represent a deformable surface as the level set of a discretely sampled scalar function of 3 dimensions, i.e. a volume. Such *level-set models* have been shown to mimic conventional deformable surface models by encoding surface movements as changes in the greyscale values of the volume. The result is a voxel-based modeling technology that offers several advantages over conventional parametric models, including flexible topology, no need for reparameterization, concise descriptions of differential structure, and a natural scale space for hierarchical representations. This paper builds on previous work in both 3D reconstruction and level-set modeling. It presents a fundamental result in surface estimation from range data: an analytical characterization of the surface that maximizes the posterior probability. It also presents a novel computational technique for level-set modeling, called the sparse-field algorithm, which combines the advantages of a level-set approach with the computational efficiency and accuracy of a parametric representation. The sparse-field algorithm is more efficient than other approaches, and because it assigns the level set to a specific set of grid points, it positions the level-set model more accurately than the grid itself. These properties, computational efficiency and sub-cell accuracy, are essential when trying to reconstruct the shapes of 3D objects. Results are shown for the reconstruction objects from sets of noisy and overlapping range maps.

Appears in: *The International Journal of Computer Vision* , 29(3), October, 1998, pp. 203–231.

1 Introduction

There are two aspects to this research. The first aspect is the application, which is the development of 3D models from range data. That problem is formulated using a statistical approach, which maximizes the posterior probability of a surface. The result is a surface that is described by a nonlinear objective function. That optimization is solved using a variational approach and a gradient-descent method. This optimization produces an evolution equation for deforming a surface so that it matches a given set of data. The work in this paper uses the level sets of volumes as a means of representing and manipulating object shapes. The second aspect of the research is the enhancement of the underlying level-set technology in order to make it suitable for problems in 3D reconstruction. This section introduces the issues surrounding those two aspects of the work, and lays out the overall structure of the paper.

1.1 3D Reconstruction

When investigating 3D reconstruction, it is necessary to describe the kind of data being considered. This work deals primarily with data that is from range or distance sensors that have some kind of "scanning capability". Three different properties of this data are important. First, the data is range or distance (i.e., direct 3D) compared to intensity or luminance, as one would expect from an X-ray or a video camera. Second, the data is relatively dense, rather than a sparse scattering of 3D points. This density is obtained by rotating or translating a device that takes a discrete set of 3D measurements. From a single point of view the range finder sweeps out a volume, and there is a 2D topology on these range measurements that is induced by the motion of the scanning device. The third important aspect of this data is that it is noisy. Thus, the 3D reconstruction problem is not strictly geometric; it is also statistical. For this work we assume that algorithms for calibrating the range sensor and registering the multiple views are taken from the variety of automated and semiautomated techniques that are in the literature — e.g. (Besl and McKay 1992, Chen and Medioni 1992, Zhang 1994).

Under the circumstances described above, approaches to 3D reconstruction can be distinguished based on the "level" of the models that one uses. High-level models have been used extensively in computer vision — some example are given in (Jain and Jain 1990, Jain and Flynn 1993). High-level models are those that represent specific objects or classes of objects. Such approaches can work with relatively little information (such as sparse sets of features), and when they work they can lead directly to higher level processes such as recognition. High-level approaches usually work best in situations where the domain is restricted and well characterized, and where the distinguishing shape characteristics of a particular object can be captured by relatively few parameters. Typically, such reconstructions do not capture those details that are unique to a particular object.

Another class of approaches could be characterized as "mid level". Mid-level approaches to reconstruction are characterized by the use of geometric primitives that can be combined in various ways to form more complicated objects. The assumption is that the objects in the domain can be decomposed into finite set of such primitives. This strategy can work quite well on certain domains, such as man-made objects—many of which were designed using such primitives. The fitting of primitives can also provide important structural information such as part/sub-part decomposition. The results of such systems are generally some mixture of the data and the models, but they tend to strongly reflect the shapes of the models or primitives that are chosen.

Finally, one could characterize the class of "low level" approaches to 3D reconstruction. These approaches use a few general assumptions about the domain; assumptions are often at the level of basic surface geometry, e.g. continuity. Low-level approaches can capture more detail but often provide very little higher level information. For this reason such approaches are well suited for applications that are geared to visualization by a human, rather than recognition by some automated system. These approaches form surfaces from sets of 3D data by performing a fusion of different scans, but they typically impose very little structure on the data. The performance of low-level approaches tends to degrade gradually as the assumptions about the environment and the sensor are violated. However, low-level approaches typically require more data in order to give useful results. The voxel-based approach in this paper is a low-level approach and is therefore best suited to applications that provide a relatively large amount of data that must be fused to produce a result with a great amount of detail.

1.2 Level-set models

The strategy taken within this research is to pose the reconstruction problem as the process of finding the surface or set of surfaces that are most likely to have given rise to the data. Thus, it is a Maximum A Posteriori (MAP) strategy which can combine the data with known properties or tendencies of the surfaces being measured. Finding those surfaces with the highest likelihood is generally a nonlinear optimization problem.

A common and often effective strategy for solving such optimization problems is to use a variational approach. That is, start with an initial solution and perturb or deform that solution so that it improves the overall likelihood. Thus, the variational approach requires a modeling technology that can undergo such goal-driven deformations. Such models, often called *deformable models*, are used extensively in the computer graphics and computer vision literature, e.g. (Kass, Witkin and Terzopoulos 1987, Staib and Duncan 1992, Miller, Breen, Lorensen, O'Bara and Wozny 1991, Terzopoulos, Witkin and Kass 1988). These models typically undergo evolution processes which implement some type of hill-climbing strategy on the objective function.

Conventional geometric models are parametric. That is, they are mappings from one space, which coincides with the dimensionality of the model, to another, which is the range. Typically, these parameterizations rely on a set of basis functions that span some finite subspace of all possible shapes. Thus when considering deformations, parametric models have several drawbacks which make them inadequate for certain kinds of applications. The dependency on the parameterization and an associated basis is critical; it limits the kinds of shapes a model can represent. Models typically do not deform far from their initial conditions without some sort of reparameterization. Such reparameterizations are often inefficient and developed on the basis of heuristics rather than a consistent mathematical foundation. The parameterization can also make it difficult to measure the intrinsic geometry of the model, as with polygonal meshes for instance.

An alternative to a parametric model is an implicit model, i.e., specifying a model as a level set of a scalar function, ϕ. This scalar function can be sampled on a discrete rectilinear grid. Such a *level-set* representation (Sethian 1996) has a number of practical and theoretical advantages over conventional parametric models, especially in the context of deformation and reconstruction. First, level-set models are topologically flexible, that is, the models can "split" into pieces to form multiple objects (Malladi, Sethian and Vemuri 1995, Whitaker and Chen 1994). Second the evolution of the embedding ϕ is a differential expression that is invariant to orthogonal group transformations (rotations and translations). The shapes formed by the level sets of ϕ are restricted only by the resolution of the discrete sampling used to represent ϕ. Finally, the representation of deformable models in terms of a *multidimensional image* provides a method for multi-scale or multi-grid solutions. For instance solutions can start on a relatively coarse grid and then proceed to progressively finer grids as the energy reaches a minimum (Whitaker 1995). Such a multigrid strategy reduces computation time, controls the relative importance of large and small-scale structures in the input image, and helps to simplify the objective function.

Despite these advantages, level-set models suffer from several drawbacks compared with parametric models. One disadvantage is the large number of computations needed to solve these equations over the entire range. Surface deformations in 3D, for example, require solutions to a 3D, nonlinear, partial differential equation: a significant computational task. Another drawback of level-set models is the absence of any direct, efficient representation of the surface during deformation. Such direct representations are useful when incorporating forces that depend on the multi-local or global information. A third drawback is the finite resolution that is imposed by the discrete approximation to the scalar field. Level-set models are limited by resolution rather than shape.

These problems are significant in the context of 3D reconstruction. The sparse-field algorithm described in this paper addresses these issues by representing a surface as both a discretely-sampled 3D field and a subset of that field which consists of a set of grid points (or voxels), called *active points*, through which the model passes. At each time step in the deformation process only a thin layer of voxels near the active points are visited and updated. In this way the computational complexity of the algorithm grows with the dimensionality of the surface, rather than the dimensionality of the volume. The set of active points are kept in a linked list which enables quick and efficient access to a set of points near or on the model.

The remainder of this paper proceeds as follows. After a brief review in Section 2 of other relevant work, Sec-

tion 3 presents the formulation of 3D reconstruction as a process of finding the most likely surface given a set of range data and some general knowledge about the relative likelihoods of different surface properties. Solving that problem requires the ability to manipulate or deform surface shape in a systematic manner. Section 4 shows how this deformation can be achieved via an implicit representation, using the level sets of a discretely-sampled scalar function of 3D. That section discusses the numerical methods that are used to solve the equations of motion. Section 5 describes an efficient computational technique, the sparse-field algorithm, and presents the results of an empirical analysis which shows that the sparse-field algorithm gives solutions that compare favorably to previously proposed algorithms. Additionally, the sparse-field algorithm is shown to overcome the aliasing artifacts associated with other level-set approaches and thereby achieves greater accuracy compared to those approaches. Section 6 shows how the level-set technology can be applied to the problem of 3D reconstruction. It shows how using level-set models within the MAP reconstruction framework provides new capabilities for surface reconstruction.

2 Related work

Several low- and mid-level reconstruction approaches described in the literature are worth noting. Turk and Levoy (1994) describe a "zippering" algorithm that combines triangle meshes, each representing a range map of the same object from a different point of view. Chen and Médioni (1994) use a triangle mesh which expands inside a sequence of range maps. This strategy is similar to that of Miller et al. (1991) which was demonstrated on 3D medical data.

Several low-level volumetric approaches have been proposed. Chien and Aggarwal (1989) have used binary volumes with the aid of oct-trees to reconstruct objects from multiple silhouettes. The focus is on using silhouettes and object features to do recognition. Muraki (1991) uses implicit or blobby models to reconstruct objects from range data. The individual blobs are spherically symmetric 3D potentials that are combined linearly so that they blend together. The global nature of the potentials makes updates somewhat expensive, thus limiting the number of primitives that can be efficiently computed.

Hoppe, DeRose, Duchamp, McDonald and Stuetzle (1992) proposed an implicit strategy which constructs a 3D field based on the signed distance transform of tangent planes generated from unordered 3D data. Regions are always associated with the nearest points, and therefore there is little or no averaging. Curless and Levoy (1996) describe a volume-based technique for combining range data. They use the signed distance transform to encode volume elements with data that represent the averages (with some allowance for outliers) of multiple measurements. Surfaces of objects are the level sets of volumes. They show that the resolution of the scanner can be overcome by such an averaging technique. Hilton, Stoddart, Illingworth and Windeatt (1996) describe a similar algorithm which is an extension to in (Hoppe et al. 1992) that accounts for interference at occluding contours and thin objects.

Another volumetric approach, which is more of a "mid-level" strategy, is that of (DeCarlo and Metaxas 1995), in which they use a combination of primitives that can deform, split, and even change topology in order to match the input data. The computer vision literature shows numerous mid-level approaches that fit volumetric models to range data—see (Dickinson, Metaxas and Pentland 1997), for example.

In robotics several researchers (Elfes 1989, Moravec 1988) have investigated the use of *occupancy grids*. The sensor model and the methods for combining sensor measurements from different points of view follow from a Bayesian formulation. Strictly speaking an occupancy grid does not give a 3D reconstruction; occupancies do not give the most likely surfaces.

With regard to deformable models, there are several parametric approaches that seek to overcome the usual topological limitations of parametric models. DeCarlo and Metaxas (1995) allow the primitives associated with an object to make discrete changes as a result of a set of rules and thresholds. Szeliski, Tonnesen and Terzopoulos (1993) propose systems of oriented particles that join together, disconnect, and rejoin in order to change topology. McInerny and Terzopoulos (1995) propose parametric models that are reparameterized based on a local grid structure that allows for changes in topology. Despite these advances, for 3D reconstruction the level-set approach offers several practical advantages, which are discussed in more detail in the sections that follow.

With regard to level-set models, this work draws on a number of recent developments in computational physics

and computer vision. Osher and Sethian (1988) have proposed the embedding of wavefront propagations that model physical systems, and they have developed numerical schemes, called *up-wind* schemes, to solve the resulting equations. In image processing Alvarez, Guichard, Lions and Morel (1992) have proposed a morphological scale space for planar curves which relies on geometric invariant curve evolutions. In computer vision Kimia, Tannenbaum and Zucker (1992) have proposed a reaction-diffusion space in which singularities represent basic properties of 2D shape. They use the same strategy as (Osher and Sethian 1988) of embedding planar curves in order to create geometric flows that are independent of any particular parameterization.

Malladi et al. (1995) have proposed a segmentation scheme based on a wavefront propagation that allows seed points to contract or expand. The author (Whitaker and Chen 1994) has shown that image "forces" (the term used in the sense of first-order physics) can drive level sets toward interesting features in images or volumes, as with conventional deformable models, while geometric flows can be used to enforce smoothness and continuity. Caselles, Kimmel and Sapiro (1995) have reached a similar formulation using a conformal mapping and have shown that the equations resulting from moving level sets in this manner are well posed. Adalstein and Sethian (1995) have proposed a narrow-band scheme, with a finite band of 6-12 grid points on either side of the level set, for reducing the computations associated with level-set models. Results in successive sections of this paper will show the ability to reduce the width of this narrow band to a single grid point will improve computational performance and accuracy.

This paper makes several contributions to previous work in this field. One is a statistical formulation of the 3D reconstruction problem that gives an optimal surface estimate while making very few assumptions about the scene. This formulation incorporates a noise model and may include some prior knowledge about the relative likelihoods of different kinds of surfaces. The result is quite general, but it is particularly well suited for level-set models. This paper presents examples that show effectiveness of the level-set approach for this application. Another contribution is the numerical scheme for implementing the level-set models. This new scheme compares favorably with previously proposed methods both in terms of computational efficiency and accuracy.

3 A MAP Reconstruction Strategy

A range finder is a device that produces a distance measurement along a particular line of site. The distance measurement and, to a lesser degree, the direction of the line of sight are corrupted by noise. For the purposes of this work we assume that the noise is additive and Gaussian. When the range finder is combined with a scanning mechanism, it produces a 2D array of data, and the noise is often assumed to be independent from one element of the array to another. In the event that the noise is non-stationary, we assume that the variability of the noise process is known (e.g. as a function of object distance or scanner orientation). The scanning mechanism is typically calibrated, enabling one to calculate the direction of the line of sight associated with each measurement in the 2D array. In the event that the scanning mechanism does not produce a 2D array, but rather an unordered list of range measurements taken in different directions, one can impose a 2D topology on the data by the use of nearest-neighbor relationships such as Delaunay triangulation. A single 2D array of range data taken from a single location of the scanner is called a *range map*. One of the goals of this work is to combine the information from a collection of range maps.

The above sensor model has some significant shortcomings. First, the noise is rarely additive Gaussian. Outliers, for instance, are a problem. The noise can often be related to surface characteristics, and therefore could be correlated. In the case of structured light sensors, such as the one used for some of the results in this paper, the data contains occluded regions for which there is no data at all. As a theoretical consideration, Gaussian noise allows finite probabilities for measurements behind the scanner. In cases where the noise level is some appreciable fraction of the the distance from the scanner (very rare in practice), a log-normal distribution for the noise might be more appropriate.

In this work, we start with the simple sensor model, i.e. independent, additive, Gaussian noise, to generate the basic algorithm and then modify that algorithm to account for some of these other complicating factors. The outliers, for instance, are handled within the minimization process rather than the underlying model. Areas where the sensor failed (either by lack of signal or occlusion) are handled by giving such points low confidence, i.e. large variance, so that they have little effect on the final results. The formulation that results from these assumptions is optimal for

independent, additive, Gaussian noise, but later sections will show that it is useful when the noise is more complex (such as in the case of outliers).

The strategy in this work is to use a maximum a posteriori (MAP) approach to construct an expression for the surface likelihood conditional on the measured data. The total error associated with a model converts into a volume integral, and the Euler-Lagrange of that volume integral defines the motion of a deformable model that seeks to maximize this likelihood.

Let $\mathcal{S}$, a compact subset of 3D, be the surface that encloses the object Ω, i.e., $\mathcal{S} = \partial\Omega$. Let $\{R^{(1)}, \ldots, R^{(n)}\}$ be a set of range maps, each of which consists of set of 3D points $R^{(i)} = \{\mathbf{r}_1^{(i)}, \ldots, \mathbf{r}_m^{(i)}\}$, which can arranged in a 2D grid, with coordinates u, v, defined by the scanning mechanism. Associated with each grid point is a ray, called the line of sight, along which the range measurement is taken. Assume that the rays within a single range map do not intersect, as is the case with the spherical, cylindrical, or linear projective scanning mechanisms that are used on most range finders.

The posterior probability of $\mathcal{S}$ is given by Bayes rule:

$$P(\mathcal{S}|R^{(1)}, \ldots, R^{(n)}) = \frac{P(R^{(1)}, \ldots, R^{(n)}|\mathcal{S})P(\mathcal{S})}{P(R^{(1)}, \ldots, R^{(n)})}. \tag{1}$$

The goal is to find the most likely surface model $\mathcal{S}$ to give rise to a particular collection of range data,

$$\sup_{\mathcal{S}} \left[P(\mathcal{S}|R^{(1)}, \ldots, R^{(n)})\right] = \sup_{\mathcal{S}} \left[P(R^{(1)}, \ldots, R^{(n)}|\mathcal{S})P(\mathcal{S})\right] \tag{2}$$

where the denominator $P(R^{(1)}, \ldots, R^{(n)})$ is removed because it is a normalization factor that does not depend on $\mathcal{S}$. The term $P(\mathcal{S})$ is the prior, which reflects the fact that within the set of possible surfaces, some are more likely than others, independent of the data.

The range maps, conditional on the surface position, are independent random variables (this ignores the effects of surface properties on error, as discussed above), and thus,

$$P(R^{(1)}, \ldots, R^{(n)}|\mathcal{S}) = P(R^{(1)}|\mathcal{S}) \ldots P(R^{(n)}|\mathcal{S}) = \prod_i P(R^{(i)}|\mathcal{S}). \tag{3}$$

Each $R^{(i)}$ consists of an array of 3D range data that can be represented as $\mathbf{r}_{u,v}^{(i)}$. Within a given range map each range measurement is a independent random variable, and therefore

$$P(R^{(1)}, \ldots, R^{(n)}|\mathcal{S}) = \prod_i \prod_{u,v} P(\mathbf{r}_{u,v}^{(i)}|\mathcal{S}). \tag{4}$$

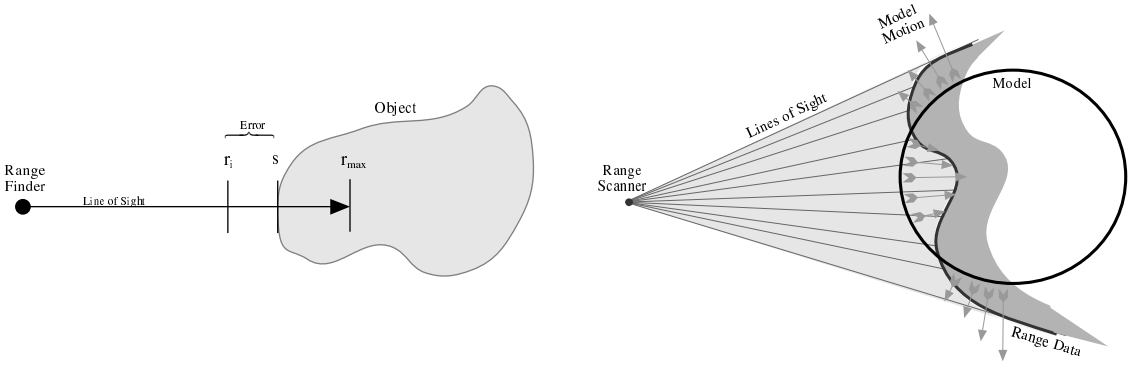
As is typical with such MAP algorithms, the maximization is performed as a minimization on the negative logarithm of the probability:

$$\sup_{\mathcal{S}} \left\{\ln\left(P(\mathcal{S}|R^{(1)}, \ldots, R^{(n)})\right)\right\} = \inf_{\mathcal{S}} \left\{-\sum_i \sum_{u,v} \ln P(\mathbf{r}_{u,v}^{(i)}|\mathcal{S}) - \ln P(\mathcal{S})\right\}. \tag{5}$$

The right-hand side of equation (5) reflects the combination of terms that is typical with MAP reconstructions. The terms are of two different types: those terms that enforce the solution to look like the data and those terms that enforce the prior, i.e., that encourage the solutions to conform to those expectations that are independent of the data.

Each individual range measurement is dependent not on the surface $\mathcal{S}$ as a whole, but on that particular point on the surface that is first intersected along the line of sight from the scanner location. Consider a single range reading r_i, represented as a distance along its associated line of sight. Suppose that the distance to the surface along that line of sight is given by s_i. The Gaussian noise model gives the following probability density function for individual range readings,

$$-\ln P(r_i|\mathcal{S}) = \frac{1}{\sigma_i^2}(s_i - r_i)^2 + \frac{1}{\sqrt{2\Pi\sigma}} = c_i(s_i - r_i)^2 + k(c_i), \tag{6}$$



(a) (b)

Figure 1: The MAP reconstruction strategy: (a) The squared error along the line of sight can be written as a an integral that includes a binary membership function for the object. (b) A single range scan, consisting of a number of measurements along lines of sight emanating from the point at the left, creates surface in 3D (shown here as a curve in the plane). A circular model deforms to fit the data (motion indicated by arrows).

where the variance σ_i^2 associated with a range reading r_i is represented as a confidence measure $c_i = 1/\sigma_i^2$. The term $k(c_i)$ can be ignored, because it does not depend on surface position and will not affect the outcome of the optimization. The squared distance of the surface to the range measurement can be represented as an integral

$$(s_i - r_i)^2 = \int_0^{r_{\max}} (\alpha - r_i)\, I_s(\alpha) \mathrm{d}\alpha - \int_{r_i}^{r_{\max}} (\alpha - r_i)\, \mathrm{d}\alpha \tag{7}$$

where $I_s(\alpha)$ is a function that is unity inside the object and zero outside, and $r_{\max}$ indicates the maximum effective range of the range finder. The term $r_{\max}$ serves as a bound on the error if the line of sight fails to intersect the surface, and it also serves to limit the range of influence of a particular range reading. That is, surface points that lie beyond $r_{\max}$ have no impact on the conditional likelihood associated with that range measurement.

Strictly speaking, this formulation is valid only if the surface intersects the line of sight at most once between the scanner and $r_{\max}$, as shown in figure 1(a). Thus, the segment along the line of sight with length $r_{\max}$ should be long enough to penetrate the model (as it deforms) but not so long that it emerges from the other side or enters the object again, as could easily happen with self occlusions. Typically, $r_{\max}$ should depend on r_i, e.g. $r_{\max} = r_i + \varepsilon$. The use of $r_{\max}$ and ε serves to limit the effects of the range data and thereby simplifies the mathematical model by ignoring self occlusions of the object. Unfortunately, in order for this to work properly, ε must be chosen so that scans to do not interfere with each other in areas where the object self occludes. This is not always possible, but with modifications to the minimization algorithm, discussed in successive sections, this problem can be all but eliminated.

Instead of putting bound $r_{\max}$ on the integral, we can use a *windowing function*, ω, that nullifies the effects of surface that lie beyond $r_{\max}$, i.e.,

$$\omega(r_{\max}, \alpha) = \begin{cases} 1 & \text{if } r_{\max} - \alpha \geq 0 \\ 0 & \text{if } r_{\max} - \alpha < 0 \end{cases} \tag{8}$$

The error becomes

$$(s_i - r_i)^2 = \int_0^{\infty} (\alpha - r_i)\, I_s(\alpha) \omega(r_i + \varepsilon, \alpha) \mathrm{d}\alpha - \int_{r_i}^{\infty} (\alpha - r_i)\, \omega(r_i + \varepsilon, \alpha) \mathrm{d}\alpha. \tag{9}$$

Given this formulation, the windowing function $\omega(\cdot)$ need not be binary; it could implement a fuzzy cutoff of the influence of the range data. Results in later sections use a Gaussian centered at the range reading with standard deviation ϵ.

To extend this formulation to include all of the samples in a single scan, R, we represent the distance along the line of sight in 3D. Let $\mathbf{n}_i$ be the unit vector along the line of sight and $\mathbf{r}_i = r_i \mathbf{n}_i$ be the 3D location of the ith range reading. Then

$$\begin{aligned} -\ln\left(P(R|\mathcal{S})\right) &= -\ln\left(P(\mathbf{r}_1, \ldots, \mathbf{r}_n|\mathcal{S})\right) \\ &= \sum_i c_i \int_0^\infty \left((\alpha\mathbf{n}_i - \mathbf{r}_i)\cdot\mathbf{n}_i\right) \omega\left((\alpha\mathbf{n}_i - \mathbf{r}_i)\cdot\mathbf{n}_i\right)\ I_{\mathcal{S}}(\alpha\mathbf{n}_i)\mathrm{d}\alpha - k \\ &= \sum_{u,v} c_{u,v} \int_0^\infty \left((\alpha\mathbf{n}_{u,v} - \mathbf{r}_{u,v})\cdot\mathbf{n}_{u,v}\right) \omega\left((\alpha\mathbf{n}_{u,v} - \mathbf{r}_{u,v})\cdot\mathbf{n}_{u,v}\right)\ I_{\mathcal{S}}(\alpha\mathbf{n}_{u,v})\mathrm{d}\alpha - k \end{aligned} \tag{10}$$

where $I_{\mathcal{S}} : \mathbb{R}^3 \mapsto \mathbb{R}$ is unity inside the object and zero otherwise, and

$$k = \sum_{u,v} c_{u,v} \int_{r_{u,v}}^\infty \omega\left((\alpha\mathbf{n}_{u,v} - \mathbf{r}_{u,v})\cdot\mathbf{n}_{u,v}\right)\ (\alpha\mathbf{n}_{u,v} - \mathbf{r}_{u,v})\cdot\mathbf{n}_{u,v}\ \mathrm{d}\alpha, \tag{11}$$

which does not depend on the surface. The notation u, v refers to the fact that the measurements from a single range map are arranged in a 2D grid. If we assume that this grid is relatively dense we can approximate the likelihood of the jth scan as an integral:

$$\begin{aligned} -\ln\left(P(R^{(j)}|\mathcal{S})\right) \approx \int\int\int & c(\varphi,\theta)\left((\alpha\mathbf{n}(\varphi,\theta) - \mathbf{r}(\varphi,\theta))\cdot\mathbf{n}(\varphi,\theta)\right) \\ & \omega\left((\alpha\mathbf{n}(\varphi,\theta) - \mathbf{r}(\varphi,\theta))\cdot\mathbf{n}(\varphi,\theta)\right)\ I_{\mathcal{S}}(\alpha\mathbf{n}(\varphi,\theta))\ \mathrm{d}\alpha\ \mathrm{d}\varphi\ \mathrm{d}\theta - k, \end{aligned} \tag{12}$$

where φ and θ are the continuous versions of u and v, respectively, and $\mathbf{n}(\cdot)$, $\mathbf{r}(\cdot)$, $c(\cdot)$, are continuous functions derived from some suitable interpolation (e.g., bilinear) of their discrete counterparts.

Because the rays that define the lines of sight associated with a single scan do not cross, there is unique mapping from each point within the 3D subspace swept out by the range finder to the point (φ, θ) within the range map that has a line of sight passing through that 3D point. Therefore, the volume integral of equation (13) can be reformulated in Cartesian coordinates:

$$-\ln\left(P(R^{(j)}|\mathcal{S})\right) \approx \int_{\mathcal{R}} c(\mathbf{x})\ D(\mathbf{x})\ \omega\left(D(\mathbf{x})\right)\ I_{\mathcal{S}}(\mathbf{x})\gamma(\mathbf{x})\ \mathrm{d}\mathbf{x}\ - k, \tag{13}$$

where $\gamma(\mathbf{x})$ is an integration factor, such that $\gamma(\mathbf{x})\mathrm{d}\mathbf{x} = \mathrm{d}\varphi\mathrm{d}\theta\mathrm{d}\alpha$. The term $D(\mathbf{x}) = \left((\mathbf{x} - \mathbf{r}(\mathbf{x}))\cdot\mathbf{n}(\mathbf{x})\right)$ is the signed distance from the surface position to the range measurement, and $\mathcal{R}$ is the volume swept out by the range finder. If we define the confidence $c(\mathbf{x})$ to be zero for any point outside the space swept out by the scanner, then one can extend bounds of the integral to $\mathbb{R}^3$. Because, $I_{\mathcal{S}}$ is a binary function that is one only within the object, the integral can be re-expressed over the object itself

$$-\ln\left(P(R^{(j)}|\mathcal{S})\right) \approx \int_{\Omega} c(\mathbf{x})\ D(\mathbf{x})\ \omega\left(D(\mathbf{x})\right)\ \gamma(\mathbf{x})\ \mathrm{d}\mathbf{x}\ - k. \tag{14}$$

The gradient descent on the posterior for a single range map is therefore

$$\frac{\partial\mathcal{S}}{\partial t} = -c(\mathcal{S})\ D(\mathcal{S})\ \omega\left(D(\mathcal{S})\right)\ \gamma(\mathcal{S})\ \mathcal{N}, \tag{15}$$

where $\mathcal{N}$ is the surface normal.

Equation (15) describes the motion of a model as it seeks to maximize its likelihood of giving rise to a single range map R, which is set of range measurements. A single range map creates a surface in 3D (shown in figure 1(b) as a curve in the plane), and the model expands or contracts, with a magnitude proportional to the distance along the line of sight to the range reading, so that it exactly fills the volume behind that surface. The windowing function $\omega(\cdot)$ controls the depth of the region behind the range surface over which this expanding or contracting action occurs. The

function $\gamma(\cdot)$ describes the unit-volume relationship between the scanner coordinates and cartesian coordinates. For the results in this paper we assume that the rays (lines of sight) emanating from the scanner are nearly parallel and regularly sampled, so that $\gamma(\cdot)$ is a constant.

The effect of multiple range maps is additive, and the Euler-Lagrange of the combined posterior is

$$\frac{\partial \mathcal{S}}{\partial t} = -g(\mathcal{S})\mathcal{N} + \rho(\mathcal{S}), \tag{16}$$

where

$$g(\mathbf{x}) = \sum_j c^{(j)}(\mathbf{x})\ D^{(j)}(\mathbf{x})\ \omega\left(D^{(i)}(\mathbf{x})\right)\ \gamma^{(j)}(\mathbf{x}), \tag{17}$$

and the superscripts indicate the range, line-of-sight, and confidence functions associated with a particular range map. The term $\rho(\mathcal{S}) = -\delta \ln P(\mathcal{S})$ is the Euler-Lagrange of the prior. In the absence of any prior (i.e., uniform prior), $\rho(\mathcal{S}) = 0$, and maximizing the combined effects of a sequence of range maps is linear; the solution is given by the zero crossings of the 3D function defined in equation (17). Indeed, the algorithms proposed by (Curless and Levoy 1996), as well as (Hoppe et al. 1992) and (Hilton et al. 1996), can be formulated as a special cases of this MAP approach, with a particular choice of $\omega(\cdot)$ and $c(\cdot)$, and with a uniform prior.

There are several reasons for going to an iterative scheme for finding optimal solutions. First is the use of a prior. In surface reconstruction, even a very low level of noise can degrade the quality of the rendered surfaces in the final result, and in such cases better reconstructions can be obtained by introducing a prior. Second is aliasing. Discretizing $g(\mathbf{x})$ and finding the zero crossings will cause aliasing in those places where the transition from positive to negative is particularly steep. A deformable model can place the surface much more precisely. The third reason for going to an iterative scheme is that despite the windowing function $\omega(\mathbf{x})$ there is interference between different range maps at places of high curvature. This problem is addressed by introducing a nonlinearity which is solved in an iterative scheme as described in Section 6.1. In this work, solutions of the linear problem will serve as the initial conditions for the nonlinear, iterative optimization strategy that results from the inclusion of a prior and a nonlinear term that compensates for lack of any explicit model of self occlusions.

3.1 The Prior

There are numerous possibilities for selecting priors that are consistent with the MAP formulation of the previous section. These possibilities range from high-level priors that bias the solution to look like certain shapes (Johnson 1993) to low-level priors that enforce some very general properties on those shapes. The goal of this work is to obtain a somewhat general reconstruction algorithm that can be subsequently tuned to specific applications. Therefore, we use a low-level prior that biases solutions toward smooth, continuous surfaces, like those associated with many man-made objects or anatomical structures. Of course, the selection of a prior will depend strongly on the application; the choice of prior for the reconstruction of injection-molded, hand-held objects should probably differ from the prior needed to reconstruct the shapes of highly irregular objects, such as the human brain with its many folds and protrusions.

For many applications, a natural choice of prior is to penalize the integral of the normalized first derivatives on the object surface, i.e., penalize surface area. This choice is based on the heuristic that many physical processes, both synthetic and natural, tend to conserve surface area and produce objects that reflect, at some level of detail, this tendency to minimize area. Alternatively, one could say that given a set of surfaces that are near the data, the algorithm should choose a surface that has less area. Often, but not always, this will be the smoother surface. A probability distribution for the prior that reflects this principle is

$$P(\mathcal{S}) = \left(\frac{\beta}{\Pi}\right)^{\frac{1}{2}} \exp\left(-\beta \int_{\mathcal{S}} \mathcal{N} \cdot \mathcal{N} \mathrm{d}\mathbf{x}\right) \tag{18}$$

After the logarithm, the Euler-Lagrange of this quantity is the mean curvature of the surface:

$$\rho(\mathcal{S}(r,s,t)) = \beta H(\mathcal{S}) = \beta \left(\frac{\partial^2 \mathcal{S}}{\partial r^2} + \frac{\partial^2 \mathcal{S}}{\partial s^2} \right) \tag{19}$$

where r, s is a local, orthonormal parameterization.

The term β controls the probability distribution of surfaces in the prior. A larger β means that the prior favors more strongly surfaces that are smoother. The parameter β could be set in any number of different ways. One way would be to "calibrate" the system by scanning a number of known objects and determine which values of β give the most accurate/reliable reconstructions. Another method would be to take a collection of objects that represent the application domain, model them, and generate a probability density function for surface area, possibly at different scales or resolutions. For this work, we treat β as a free parameter that must be tuned by the person using the algorithm.

The existence of a free parameter β represents a typical property of signal processing systems that must deal with noise in the absence of specific information about the signal. Results in the following sections will show that the algorithm is somewhat robust with respect to this free parameter, β, and that the algorithms fails gradually as this parameter is set too high or low. This paper will also show that very small values of β can reduce the effects of uncorrelated noise without distorting the overall shapes of the objects being reconstructed.

Despite the usefulness of this second-order smoothing term, it is somewhat limited in its effectiveness because it can interpolate only position and tends to create straight lines, flat surfaces, or singularities where there is little data. The topic of developing more effective low-level priors (Whitaker 1995), as well as incorporating high-level priors, is an area of ongoing investigation. Fourth-order terms, for example, could create structures with smoothly varying normals and allow the MAP approach to operate with less data (Sethian 1996). This paper will show that despite the limitations of the second-order smoothing given in equation (19), that prior is an improvement over none at all, especially in cases of noisy surface data.

4 Level-Set Models

The hill-climbing optimization strategy described in the previous section requires a modeling technology that is capable of accommodating incremental changes surface shape in an efficient manner. The equation of motion given by (16) makes no assumptions about the type surface model used to achieve the reconstruction. The MAP formulation could be adapted to any number of conventional, parametric surface models by expressing changes in surface position in terms of the parameters that control surface shape.

However, there are some drawbacks to using parametric deformable models. For instance, as models evolve and undergo large deviations from their original shapes, surface parameterizations often introduce de facto constraints. The expansion of polygonal models can create a kind of coarseness which prevents the model from capturing smaller structures; thus the evolution of polygonal models requires the creation and deletion of polygons (Miller et al. 1991, MacDonald, Avis and Evans 1994, Chen and Médioni 1994), or alternatively, a reparameterization (McInerny and Terzopoulos 1995, DeCarlo and Metaxas 1995). Also the choice a surface model with relatively few degrees of freedom, such as superquadric or superelipsoid, restricts solutions to the relatively small space of shapes that are captured by those modeling technologies.

An alternative to a parametric model, is a level-set model (Sethian 1996), i.e., a model that treats a 3D surface as the level-set of a discretely-sampled volume. Previous work has shown promising results with this modeling technology for 3D reconstruction, primarily in the context of 3D medical data (Malladi et al. 1995, Whitaker and Chen 1994, Whitaker 1994). This section gives the formulation for this modeling strategy, starting with an equation of motion for a deformable surface (Whitaker and Chen 1994). This formulation differs somewhat from that of (Caselles et al. 1995), which they develop by applying a conformal mapping to the energy function defined over the range.

The strategy is to rely on the properties of regular surfaces, which have local parameterizations that can be ex-

pressed in terms of intrinsic quantities such as arc length, curvature, etc. If one starts with a parameterized surface and the associated equations of motion, and then removes the parameterization from the deformable model, all that remains is intrinsic geometry, which is expressed in terms of the embedding, ϕ. The strategy for embedding active surfaces consists of four steps:

1. Express the equations of motion for a deformable model in terms of some unspecified parameterization (as was done in Section 3).
2. Describe the parameterization in terms of the differential structure of the model.
3. Assume the model is the level set of a function ϕ.
4. Express the geometry of the level set in terms of the differential structure of ϕ and create an evolution equation for ϕ.

In order to apply this strategy to a deformable surface $\mathcal{S}$, represent $\mathcal{S}$ as a level set of an 3D scalar function, $\phi : \mathbb{R}^3 \times \mathbb{R}^+ \mapsto \mathbb{R}$, which evolves over time. The evolution equations of the individual level surfaces imply corresponding evolution equations for the scalar function $\phi(\mathbf{x}, t)$, where $\mathbf{x} \in \mathbb{R}^3$.

A parametric deformable model, $\mathcal{S}(r, s, t)$ where $r, s \in U \subset \mathbb{R}$, can be represented

$$\mathcal{S} = \{\mathbf{x} | \phi(\mathbf{x}, t) = k\}. \tag{20}$$

The surface $\mathcal{S}$ remains a level set of ϕ over time, and therefore the time derivative is zero:

$$\frac{\partial \phi(\mathcal{S}, t)}{\partial t} + \nabla \phi(\mathcal{S}, t) \cdot \frac{\partial \mathcal{S}}{\partial t} = 0, \tag{21}$$

where

$$\nabla \phi(\mathbf{x}) \triangleq \left(\frac{\partial \phi}{\partial x}, \frac{\partial \phi}{\partial y}, \frac{\partial \phi}{\partial z} \right). \tag{22}$$

Thus,

$$\frac{\partial \phi}{\partial t} = -\nabla \phi \cdot \frac{\partial \mathcal{S}}{\partial t} = |\nabla \phi| \frac{\partial \mathcal{S}}{\partial t} \cdot \mathcal{N}, \tag{23}$$

where $\mathcal{N} = -\nabla \phi / |\nabla \phi|$ is the surface normal.

The data term for reconstruction from equation (16) takes the form

$$\frac{\partial \mathcal{S}}{\partial t} = -g(\mathbf{x}) \mathcal{N} \tag{24}$$

where $g(\cdot)$ is the cumulative effect from all of the individual range maps as in equation (17). The level-set formulation, without the prior, becomes

$$\frac{\partial \phi}{\partial t} = g(\mathbf{x}) |\nabla \phi| \tag{25}$$

The mean curvature, used for the prior, from equation (19) is computed directly from the first- and second-order structure of ϕ,

$$\begin{aligned} |\nabla \phi| \ (\mathcal{S}_{rr} + \mathcal{S}_{ss}) \cdot \mathcal{N} \quad = \quad & \left(\phi_x^2 + \phi_y^2 + \phi_z^2\right)^{-\frac{1}{2}} \left[\left(\phi_y^2 + \phi_z^2\right) \phi_{xx} + \left(\phi_z^2 + \phi_x^2\right) \phi_{yy} \right. \\ & \left. + \left(\phi_x^2 + \phi_y^2\right) \phi_{zz} - 2\phi_x \phi_y \phi_{xy} - 2\phi_y \phi_z \phi_{yz} - 2\phi_z \phi_x \phi_{zx} \right]. \end{aligned} \tag{26}$$

In the numerical implementation, the derivatives are calculated using centralized differences (Sethian 1996).

Combining the data term and the prior gives the following level-set formulation:

$$\begin{aligned} \frac{\partial \phi}{\partial t} \quad = \quad & g(\mathbf{x}) |\nabla \phi| + \beta \left(\phi_x^2 + \phi_y^2 + \phi_z^2\right)^{-\frac{1}{2}} \left[\left(\phi_y^2 + \phi_z^2\right) \phi_{xx} + \left(\phi_z^2 + \phi_x^2\right) \phi_{yy} \right. \\ & \left. + \left(\phi_x^2 + \phi_y^2\right) \phi_{zz} - 2\phi_x \phi_y \phi_{xy} - 2\phi_y \phi_z \phi_{yz} - 2\phi_z \phi_x \phi_{zx} \right] \end{aligned} \tag{27}$$

Equation (27) is invariant under certain kinds of geometric transformations. First, it is invariant to orthogonal group transformations on $\mathbf{x}$. Thus the position and orientation of the model or the data has no impact (to within the

error introduced by the representation of ϕ) on the solution. Equation (27) is also invariant to monotonic transformations on ϕ; that is, equation (27) acts only on level sets and treats each level set of ϕ as an individual surface evolving under the same set of set of priors and forces as all other level sets of ϕ.

4.1 Numerical Algorithms

The differential equation described by equation (27) has two parts. The first part is a first-order term that has the form $g(\mathbf{x})|\nabla\phi(\mathbf{x})|$. It is a moving wave front with a velocity that depends on position. This expression cannot be solved with a simple forward finite difference scheme. Such schemes tend to overshoot, and they are unstable. To solve this problem Osher and Sethian (1988) propose an *up-wind* scheme. The up-wind method uses a one-sided derivative that looks in the up-wind direction of the moving wavefront, and thereby avoids the overshooting associated with forward finite differences.

The one-sided derivatives are denoted by $\mathrm{d}^{(+)}$ and $\mathrm{d}^{(-)}$. Let $u_{x,y,z}$ with domain X be an approximation to ϕ defined on a discrete rectilinear grid with spacing h. The one-sided derivatives are

$$\mathrm{d}_x^{(+)} u_{x,y,z} \triangleq (u_{x+h,y,z} - u_{x,y,z})/h, \tag{28}$$

$$\mathrm{d}_x^{(-)} u_{x,y,z} \triangleq (u_{x,y,z} - u_{x-h,y,z})/h. \tag{29}$$

For the first term of equation (27) the up-wind scheme has the following form (Osher and Sethian 1988):

$$\begin{aligned} g(x,y,z)|\nabla u| = \Big(& \left[\max\left(g(x,y,z),0\right)\ \mathrm{d}_x^{(+)}u + \min\left(g(x,y,z),0\right)\ \mathrm{d}_x^{(-)}u\right]^2 \\ & + \left[\max\left(g(x,y,z),0\right)\ \mathrm{d}_y^{(+)}u + \min\left(g(x,y,z),0\right)\ \mathrm{d}_y^{(-)}u\right]^2 \\ & + \left[\max\left(g(x,y,z),0\right)\ \mathrm{d}_z^{(+)}u + \min\left(g(x,y,z),0\right)\ \mathrm{d}_z^{(-)}u\right]^2 \Big)^{\frac{1}{2}}, \end{aligned} \tag{30}$$

where the time steps are limited by the speed of the fastest moving wavefront,

$$\Delta t \leq \frac{1}{\sup_{x,y,z\in X}\{|\nabla g(x,y,z,t)|\}} \tag{31}$$

The second term in equation (27) is a diffusion term that can be solved using a finite forward difference scheme and does not require the up-wind method.

4.2 Narrow-Band Methods

If one is interested in only a single level set, the formulation described previously is not computationally efficient. This is because solutions are usually computed over the entire domain of u. The solutions, $u_{x,y,z}(t)$ describe the evolution of an embedded family of contours. While this dense family of solutions might be advantageous for certain applications, there are other applications that require only a single surface model. In such applications the calculation of solutions over a dense field is an unnecessary computational burden, and the presence of contour families can be a nuisance because further processing might be required to extract the level set that is of interest.

When solving for only a single level set, $\phi(\mathbf{x},t) = k$, the evolution of ϕ is important only in the vicinity of that level set. The evolution of the implicit models is such that the level sets evolve independently (to within the error introduced by the discrete grid). Thus, one should perform calculations for the evolution of ϕ only in a neighborhood of the surface $\mathcal{S} = \{\mathbf{x}|\phi(\mathbf{x}) = k\}$. In the discrete setting, there is a particular subset of grid points whose values control a particular level set (see figure 2).

Adalstein and Sethian (1995) propose a *narrow-band* approach that takes advantage of the fact that the movement of a particular level set is a local phenomenon. The narrow-band technique constructs an embedding of the evolving curve or surface via a signed distance transform. The distance transform is computed over a finite width of only m points, and the remaining points are set to constant values to indicate that they do not lie within the narrow band, or *tube* as they call it. The evolution of the surface (they demonstrate it for curves in the plane) is computed by

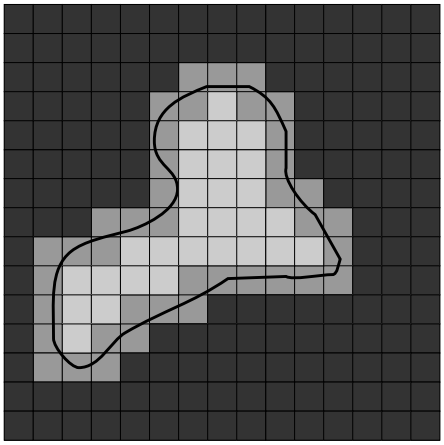


Figure 2: A level curve of a 2D scalar field passes through a finite set of cells. Only those grid points nearest to the level curve are relevant to the evolution of that curve.

calculating the evolution of u only on the set of grid points that are within a fixed distance to the initial level set, i.e. within the narrow band. When the evolving level set approaches the edge of the band, they calculate a new distance transform and a new embedding, and they repeat the process. This algorithm relies on the fact that the embedding is not a critical aspect of the evolution of the level set. That is, the embedding can be transformed or recomputed at any point in time, so long as such a transformation does not change the position of the kth level set, and the evolution will be unaffected by this change in the embedding.

Despite the improvements in computation time, the narrow-band approach is not optimal for several reasons. First it requires a band of significant width ($m = 12$ in the examples of Adalstein and Sethian (1995)) where one would like to have a band that is only as wide as necessary to calculate the derivatives of u near the level set (e.g. $m = 2$). The wider band is necessary because the narrow-band algorithm trades off two competing computational costs. One is the cost of stopping the evolution and computing the position of the curve and distance transform (to sub-cell accuracy) and determining the domain of the band. The other is the cost of computing the evolution process over the entire band. The narrow-band method also requires additional techniques, such as smoothing, to maintain the stability at the boundaries of the band, where some grid points are undergoing the evolution and nearby neighbors are stationary.

5 Sparse-Field Solutions

The narrow-band algorithm reduces computation time by restricting the updates to a *band* of grid points that lie near the level set. However, that strategy is based on the assumption that computing the distance transform is so costly that it cannot be done at every iteration of the evolution process — the band of computation must be wide enough to justify this costly update of the embedding.

The sparse-field algorithm proposed in this section uses an approximation to the distance transform that makes it feasible to recompute the neighborhood of the level-set model at each time step. Thus, it takes the narrow-band strategy to the extreme; it computes updates on a band of grid points that is only one point wide. The values of the points in the active set can be updated using the up-wind scheme and the mean-curvature flow described in the previous sections. When computing updates on so few points, however, one must be careful to maintain a neighborhood around those points so that the derivatives that control the process can be computed with sufficient accuracy. The strategy is to extend the embedding from the active points outward in layers to create a neighborhood around those points that is precisely the width needed to calculate the derivatives for the next time step.

This approach has several advantages. The algorithm does precisely the number of calculations needed to compute

the next position of the level curve. It does not require explicitly recalculating the positions of level sets and their distance transforms. For large 3D data sets, the very process of incrementing a counter and checking the status of all of the grid points is prohibitive. In the sparse-field algorithm the number of points being computed is so small, it is feasible to use a linked-list to keep track of them. Thus, at each iteration the algorithm *visits* only those points adjacent to the k-level curve. Also, the sparse-field approach identifies a single level set with a specific set of points whose values control the position of that level set. This allows one to compute external forces to an accuracy that is better than the grid spacing of the model, resulting in a modeling system that is more accurate for 3D reconstruction.

The k-level surface, S, of a function u defined on a discrete grid has a set of cells through which it passes, as shown in figure 2. The set of grid points adjacent to the level set is called the *active set*, and the individual elements of this set are called *active points*. As the model deforms the active will change. All of the derivatives (up to second order) required to calculate the update of u are computed using nearest neighbor differences. Therefore, only the active points and their neighbors are relevant to the evolution of the level-set at any particular time in the evolution process.

One important aspect of the sparse-field algorithm is the mechanism to control membership in the active set. In order to maintain stability, one must update the active set and neighboring points in a way that allows grid points to enter and leave the active set without those changes in status affecting their values. This mechanism can be understood as follows. Active points must be adjacent to the level-set model. Therefore their positions lie within a fixed distance to the model. Because the embedding is a distance transform, the values of u for locations in the active set must lie within a certain range of values. When active-point values move out of this *active range* they are no longer adjacent to the model. They must be removed from the set and other grid points, those whose values are moving into the active range, must be added to take their place. The precise ordering and execution of these operations is critical to the proper operation of the algorithm.

If we assume that the embedding u is a discrete approximation to the distance transform of the model, then the distance of a particular grid point, x_j, to the level set is given by the value of u at that grid point. If the distance between grid points is defined to be unity, then we should remove a point from the active set when the value of u at that point no longer lies in the interval $[-\frac{1}{2}, \frac{1}{2}]$ (see figure 3). If the neighbors of that point maintain their distance of 1, then those neighbors will move into the active range just as x_j is ready to be removed.

There are two operations that are significant to the evolution of the active set. First, the values of u at active points change from one iteration to the next. Second, as the values of active points move out of the active range they are removed from the active set and other, neighboring grid points are added to the active set to take their place. The appendix of this paper gives some formal definitions of active sets and the operations that affect them, and it shows that active sets will always form a boundary between positive and negative regions in the discrete sampling u, even as control of the level set passes from one set of active points to another.

Because grid points that are near the active set are kept at a fixed value difference from the active points, active points serve to control changes in the nonactive grid points to which they are adjacent. The neighborhoods of the active set are defined in *layers*, $L_{+1}, \ldots L_{+N}$ and $L_{-1}, \ldots L_{-N}$, where the i indicates the city-block distance from the nearest active grid point, and negative numbers are used for the outside layers. For notational convenience the active set is denoted L_0.

The number of layers should coincide with the size of the footprint or neighborhood used to calculate derivatives. In this way, the inside and outside grid points undergo no changes in their values that affect or distort the evolution of the zero set. The work in this paper uses first- and second-order derivatives computed on a a $3 \times 3 \times 3$ kernel (city-block distance 2 to the corners). Therefore only five layers are necessary: 2 inside layers, 2 outside layers, and the active set. These layers are denoted L_1, L_2, L_{-1}, L_{-2}, and L_0, respectively.

The active set has grid point values in the range $[-\frac{1}{2}, \frac{1}{2}]$. The values of the grid points in each neighborhood layer are kept 1 unit from the next layer closest to the active set (as in figure 3). Thus the values of layer L_i fall in the interval $[i - \frac{1}{2}, i + \frac{1}{2}]$. For $2N + 1$ layers, the values of the grid points that are not in any of the layers are either inside all of the layers, with a value of $N + \frac{1}{2}$, or outside all of the layers, with a value of $-N - \frac{1}{2}$. The procedure for updating the image and the active set based on surface movements is as follows:

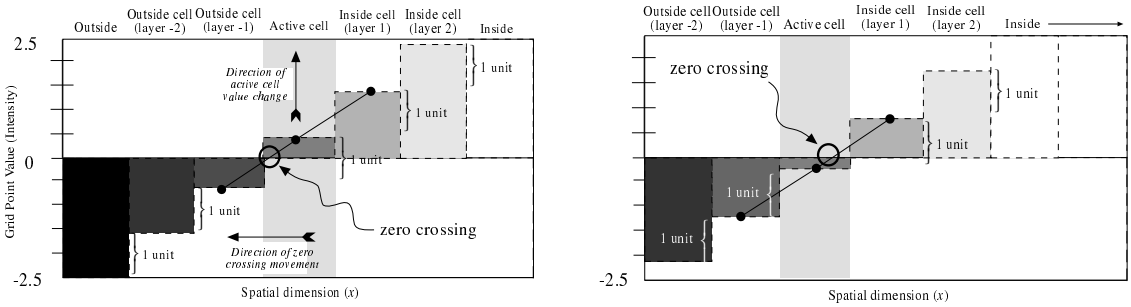


Figure 3: The status of grid points and their values at two different points in time show that as the zero crossing moves, *activity* is passed from one grid point to another.

1. For each active grid point, x_j, do the following:
   (a) Calculate the local geometry of the level set.
   (b) Compute the net change of u_{x_j}, based on the internal and external forces, using some stable (e.g., upwind) numerical scheme where necessary.
2. For each active grid point x_j add the change to the grid point value and decide if the new value $u_{x_j}(t + \Delta t)$ falls outside the $[-\frac{1}{2}, \frac{1}{2}]$ interval. If so, put x_j on lists of grid points that are changing status, called the *status list*; S_1 or S_{-1}, for $u_{x_j}(t + \Delta t) > \frac{1}{2}$ or $u_{x_j}(t + \Delta t) < -\frac{1}{2}$, respectively.
3. Visit the grid points in the layers L_i in the order $i = \pm 1, \ldots \pm N$, and update the grid point values based on the values (by adding or subtracting one unit) of the next inner layer, $L_{i \mp 1}$. If more than one $L_{i \mp 1}$ neighbor exists then use the neighbor that indicates a level curve closest to that grid point, i.e., use the maximum for the outside layers and minimum for the inside layers. If a grid point in layer L_i has no $L_{i \mp 1}$ neighbors, then it gets demoted to $L_{i \pm 1}$, the next level away from the active set.
4. For each status list $S_{\pm 1}, S_{\pm 2}, \ldots, S_{\pm N}$ do the following:
   (a) For each element x_j on the status list S_i, remove x_j from the list $L_{i \mp 1}$, and add it to the L_i list, or, in the case of $i = \pm(N + 1)$, remove it from all lists.
   (b) Add all $L_{i \mp 1}$ neighbors to the $S_{i \pm 1}$ list.

This algorithm can be implemented efficiently using linked-list data structures combined with arrays to store the values of the grid points and their states as shown in figure 4. This requires only those grid points whose values are changing, the active points and their neighbors, to be visited at each time step. Therefore computation time grows as m^{n-1}, where m is the number of grid points along one dimension of u. Computation time for dense-field approach increases as m^n. The m^{n-1} growth in computation time for the sparse-field models is consistent with conventional (parameterized) models, for which computation times increase with the resolution of the domain, rather than the range.

Another important aspect of the performance of the sparse-field algorithm is the larger time steps that are possible. In the numerical schemes for updating level-set models, the time steps are limited by the speed of the "fastest" moving level curve, i.e., the maximum of the force function. Because the sparse-field method calculates the movement of level sets over a subset of the image, time steps are bounded from below by those of the dense-field case, i.e.,

$$\sup_{x \in \mathcal{A} \subset X} (g(x)) \leq \sup_{x \in X} (g(x)), \tag{32}$$

where $g(x)$ is the space varying speed function and $\mathcal{A}$ is the active set.

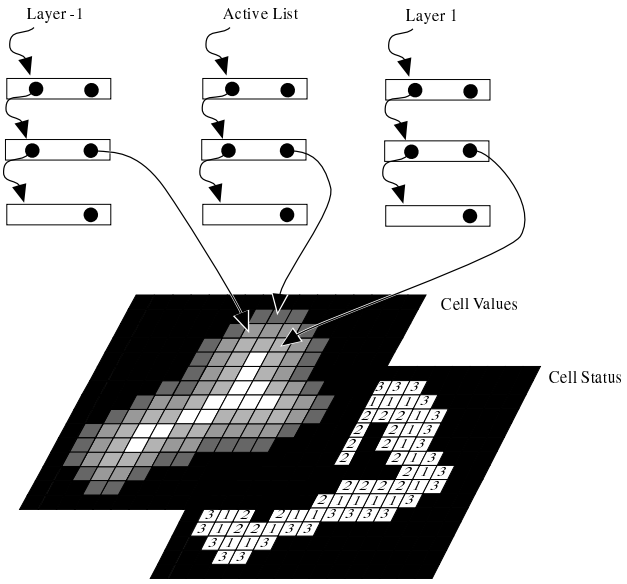


Figure 4: Linked-list data structures provide efficient access to those grid points with values and status that must be updated.

5.1 Empirical results of The Sparse-Field Algorithm

5.1.1 Error analysis

The sparse-field algorithm described above is based on an important approximation: grid points adjacent to the active points are assumed to undergo the same change in value as their nearby active-set neighbors. The dense-field algorithms treat each level set (and each grid point) separately. Because proximate level sets can have different shapes and undergo different forces, nearby grid points can undergo different changes in value. The question is how the sparse-field approximation affects the evolution of the zero-level set. The results in this section show that the evolution of the zero-level set in the sparse-field algorithm introduces an error that is consistent with that of the dense-field approach and that both errors are significantly smaller than the grid spacing h.

In order to measure the effects of these approximations we compare the movement of the level sets computed by the both the dense-field and sparse-field methods to the deformation of a circle, which can be computed analytically (Adalstein and Sethian 1995). The total error of a model is computed from the set of zero crossings along the lines connecting the grid. These points are found by using a linear interpolation between adjacent points that lie on either side of a zero crossing. The total error is the average squared distance of these zero crossings from the ideal.

Figure 5a shows the error of the dense- and sparse-field methods in 2D for a circle moving under its own curvature. Figure 5b shows the same results for the a circle moving in the direction of the inward normal at a uniform speed of 1. The 2D grid is 100×100, and the grid spacing is unity. The circle begins with a radius of 30 units. These results show that, on the whole, the sparse-field method and dense-field methods give comparable errors. The magnitude of these errors, when scaled appropriately for differences in time constants and grid spacing, are consistent with those documented in (Adalstein and Sethian 1995) even though the error metric is slightly different. Notice that in the case of constant inward velocity, both discrete level-set methods have high errors as the circle radius becomes quite small (it should be zero at $t = 30$). This is to be expected and is a inevitable consequence of using discrete methods to represent objects with sizes that are comparable to the grid spacing.

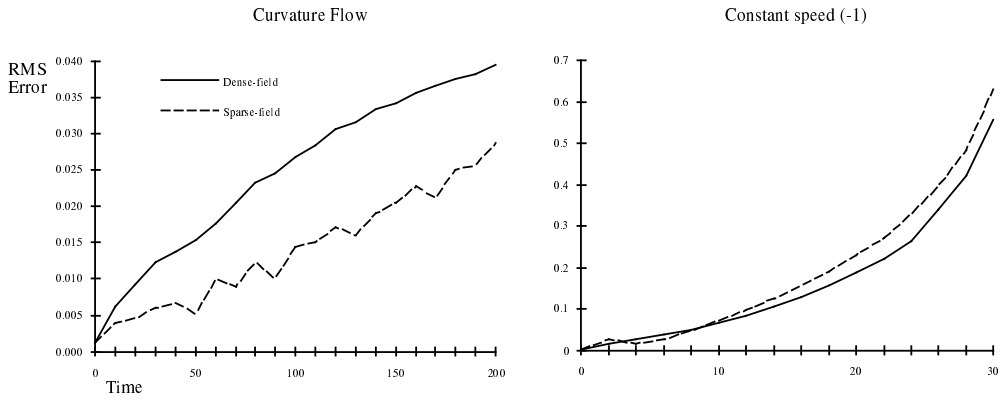


Figure 5: The rms errors, relative to ground truth, for a circle evolving under constant speed and curvature suggest that the errors associated with the sparse-field algorithm are no worse than those introduced by the discretization and level-set approximation.

5.1.2 Timing

Measurements of the timing bear out the expected improvements in performance with the sparse-field method. A rough calculation of the expected improvements is as follows. In 2D, an $n \times n$ image requires kn^2 calculations per update, where k is the number of calculations per grid point. The updates for the sparse field should be $k'n$, where k' includes the extra costs associated with updating and maintaining the neighborhood. Let $R = k/k'$, be the efficiency of the improvement. Of course, we expect R is less than one because of the extra overhead of maintaining the active set and the neighborhood around it. For a circle of radius $n/3$, the cardinality of the active set is approximately $2n$. The neighborhood is another 4 layers, each requiring 1 pass for an update. Let the cost of visiting maintaining the list and updating the neighborhood be denoted α. he Then the ratio of calculation times, R, for the two methods should be approximately

$$R = \frac{k}{k'} = \frac{k}{(4\alpha + 1)2k} = \frac{1}{8\alpha + 2}, \tag{33}$$

If α is of the same order magnitude as k (because it requires several floating point calculations involving nearest neighbors), then equation 33 gives some rough bounds on the efficiency: $0.012 \leq R \leq 0.36$.

Of course these numbers are estimates and depend quite heavily on the implementation. For the experiments of this section we have used a rather high-level, object-oriented, *C++* image processing library (developed in house), which uses in-line functions where practical as well as templated images and generic data structures. This library emphasizes ease of implementation rather than performance. The "inner loops" of the methods compared, i.e. those loops that do the calculations and updates at each pixel, are written with identical code where possible. All results are computed on a Sun Sparc 10.

Table 1 gives the average time per iteration, averaged over 25 iterations, for a circle of radius $n/3$ moving with first curvature and then a constant inward speed. The execution times and the efficiency factors confirm the expected improvements in performance from the sparse-field method.

Making direct comparisons of these computation times with the results of other researchers is difficult because of differences in implementations and hardware. However, the ratio of the dense-field computation time to that of the sparse-field algorithm gives a *performance ratio*. We have computed the same performance ratio for the narrow-band method from the data in (Adalstein and Sethian 1995). Figure 6 shows a graph of that ratio for the sparse-field approach and the narrow-band method for two distinct band widths, 6 and 12. These results show that the sparse-field method is somewhat faster, and the improvements in computation time grow as the domain size increases. For larger models, the difference in computation time between these methods is even more extreme. The models in these experiments are relatively small and the time difference should be more dramatic as one goes to 3D. For instance,

Image Width	Sparse-Field	Dense-Field	Efficiency Factor R
Circle moving under curvature.			
75	0.02	0.12	0.080
150	0.03	0.49	0.109
300	0.07	1.96	0.093
600	0.14	7.87	0.094
Circle moving at constant speed -1.			
75	0.02	0.11	0.073
150	0.05	0.44	0.059
300	0.10	1.74	0.058
600	0.20	6.97	0.058

Table 1: A comparison of execution times (seconds/iteration) for computing the evolution a circle

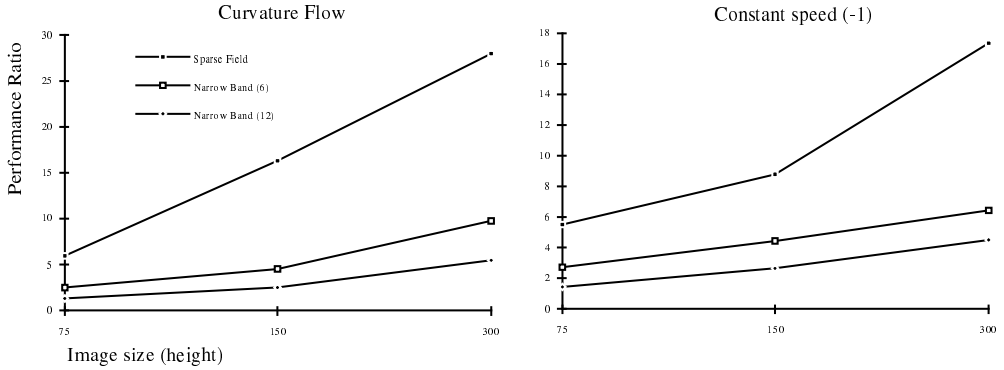


Figure 6: A comparison of the sparse-field method with the narrow-band results from (Adalstein and Sethian 1995) bears out the advantage of the sparse field method. The advantage is modest for small models but grows as the models get larger — an important trend when considering large, volumetric models.

(a) (b) (c)

Figure 7: The deformation of a cube moves downhill on the distance transform of a torus. In (a) the steady state of the dense-field approach shows that the torus suffers from aliasing due to the shocks which form near the boundary of the torus. In (b) the sparse-field algorithm bounces back and forth between grid points that have forces in opposite directions. In (c) the modified sparse-field algorithm uses a first-order approximation to position the zero-level set to subcell accuracy.

the circle in the 300×300 image contains approximately 600 pixels along its border. The surface of a sphere in a $300 \times 300 \times 300$ volume would contain about 40,000 voxels.

5.2 Subcell accuracy

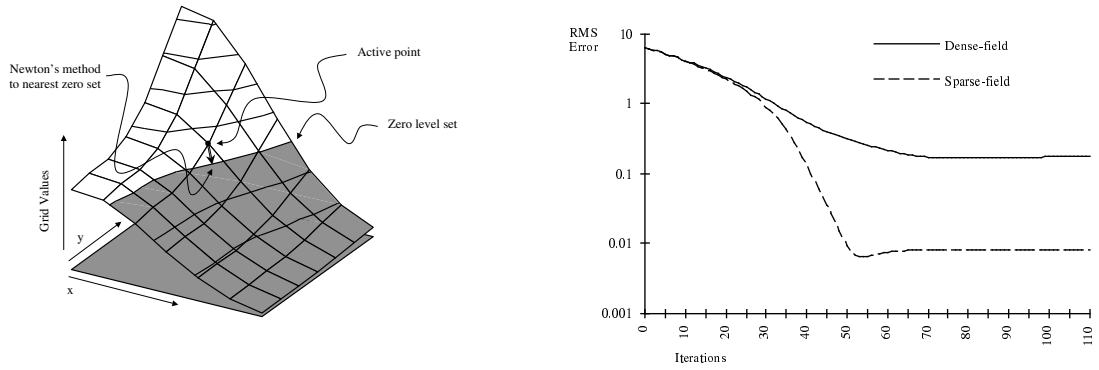
Results of the previous section demonstrate the accuracy and computational efficiency of the sparse-field algorithm. However, the level-set approach is still not appropriate for 3D reconstruction because of its limited spatial resolution. This section shows how the sparse-field algorithm can be modified to improve the overall accuracy of the level-set method.

It is well known that front propagations of the form of equation (25), without any smoothing term, form shocks where fronts moving in opposite directions meet. In the discrete domain, when using an upwind scheme, these shocks take the form of high contrast regions that form along those grid points that lie near the zero crossings of $g(\cdot)$. Unfortunately these shocks have an adverse side effect; they constrain the positions of level-set solutions to fall on the grid lines, half way between grid points. The high contrast regions associated with shock formation cause *aliasing* in the final results of level-set models.

This aliasing is not an inherent property of the implicit representation. Indeed, grid-point values can be manipulated to position zero crossings anywhere on the grid lines, and linear or higher order interpolation techniques can be used to construct parametric representations from level sets to within subcell accuracy. The aliasing associated with the moving wavefronts follows from the fact that the numerical schemes for propagating fronts sample the force function $g(\cdot)$ only at grid point locations; it is an inherent problem in the level-set numerical schemes that have been proposed to date. Any iterative deformation scheme that samples the forcing function at only a discrete set of grid locations will be limited in its ability to accurately locate the solution.

The problems associated with the discrete sampling of the forcing function are particularly troublesome when considering 3D reconstruction. Figure 7(a) shows the result of allowing a cube (sampled on a $50 \times 50 \times 50$ grid) to move on the distance transform of a torus. The distance transform of the torus (which serves as the $g(\mathbf{x})$ for the reconstruction) is computed analytically and then sampled on the same grid as the cube. The level-set model forms patterns that reflect the underlying grid structure because of the shocks that form between grid points that are inside and outside of the torus boundary. The problems of shock creation and aliasing are even more serious in cases where the forces, represented by $g(\cdot)$, have a greater resolution than the model — as in the case of recovering 3D models from high-resolution range maps. In such cases limiting the model position to the resolution of the grid is far from optimal; it introduces unnecessary artifacts.

The sparse-field algorithm provides a mechanism for positioning level sets to subcell accuracy. In the sparse-field algorithm the active grid points can be thought of as control points for a nearby zero-level set. The level set does not necessarily pass through the center of the grid point (except in the special case where the grid point has a value zero).



(a) (b)

Figure 8: The modified sparse-field algorithm: (a) For a curve in a plane, the position of the level set near to a grid point is found using Newton's method. (b) For a square being fit to a circle, the sparse-field method obtains subcell accuracy and almost perfect fit (less than a hundredth of a voxel error), while the dense-field scheme forms shocks and stabilizes at an error of 0.20 cells.

Newton's method gives a first-order approximation to the position of the nearby zero-level set, as in figure 8(a).

Let $\mathbf{x} \in \mathrm{I}^D$ be the position of a grid point, $\nabla u_{\mathbf{x}}$ be the vector of first derivatives of the model at some time t. The position of the closest zero-level set to the grid point $\hat{\mathbf{x}}$ is given by

$$\hat{\mathbf{x}} = \mathbf{x} - u_{\mathbf{x}} \frac{\nabla u_{\mathbf{x}}}{\nabla u_{\mathbf{x}} \cdot \nabla u_{\mathbf{x}}}, \tag{34}$$

except when $\nabla u_{\mathbf{x}} = 0$, which must be handled as a special case. Computing the forces on level-set locations away from the grid points is similar in philosophy to the method of *extending* the velocity fields of level sets described by Sethian (1996).

The numerical computation of $\nabla u_{\mathbf{x}}$ in equation (34) can proceed in one of several different ways. Although centralized differences are possible, the closest zero crossing can be found by finding the *steepest* one-sided derivative in each dimension. Define the function

$$\mathrm{MaxAbs}(a, b) \triangleq \begin{cases} a & |a| > |b| \\ b & |a| < |b| \\ \frac{a+b}{2} & |a| = |b| \end{cases} . \tag{35}$$

The direction of nearest zero crossing can be calculating by using

$$\nabla u_{\mathbf{x}} \triangleq \mathrm{MaxAbs}(\mathrm{d}_{x_1}^{(+)} u_{\mathbf{x}}, \mathrm{d}_{x_1}^{(+)} u_{\mathbf{x}}), \ldots, \mathrm{MaxAbs}(\mathrm{d}_{x_D}^{(+)} u_{\mathbf{x}}, \mathrm{d}_{x_D}^{(+)} u_{\mathbf{x}}). \tag{36}$$

Figure 7(c) shows that better results are obtained from the modified sparse-field method (first-order approximation to the level set location) than methods which sample force-field values only at grid point locations. The first-order approximation allows grid points to achieve grey-level values that reflect their distance from nearby features. This first-order improvement is essential in using the level-set paradigm for modeling 3D objects. The rendering of 3D models makes use of first-order derivatives of the volume data, which are sensitive to aliasing artifacts.

Figure 8(b) shows the error, using the same error measure described in section 5.1.1, for a square moving downhill on the distance transform of a circle. The error begins at about 7 units and slowly decreases as the model moves toward the circle. The difference in the sparse-field and dense-field methods demonstrates the subcell accuracy that is obtained with the first-order modifications. The dense-field method forms shocks, and the error stabilizes at about 0.20 cells. The dense-field scheme produces a binary image which is the inside-outside function for the circle. This result is representative of other numerical algorithms for level sets, such as that of (Adalstein and Sethian 1995),

which do not attempt to position level sets to sub-cell accuracy. The sparse-field method positions the zero-level set to subcell accuracy and eventually obtains very small errors in fitting the level set. The small dip in error at about 50 iterations is even more dramatic in the dense-field approach but is not easily visible on the logarithmic scale given in figure 8(b). This overshoot indicates that the model moves through the solution and slightly beyond before reaching a steady state.

This strategy of approximating the level-set position to sub-cell accuracy can generalized to higher orders. For instance, in two dimensions one can fit a second-order function to u in the neighborhood of the active grid point. Such higher order schemes are not pursued in this paper for two reasons. First, they would add considerably to the computational burden of the method. Second, the embedding of the level set is the distance transform implies that $|\phi| = 1$ almost everywhere, and therefore second derivative in the gradient direction is virtually 0 except at singularities (Bruce and Giblin 1986).

6 Level-Set Models for 3D Reconstruction

This section combines the level-set modeling technology from Sections 4 and 5 with the MAP reconstruction formulation of Section 3 to generate 3D reconstructions of objects from multiple range maps. The strategy is as follows. Construct a rather coarse volume that is the solution to the linear problem, i.e. the zero-level sets of $g(\cdot)$, without the prior. This volume serves as initialization for a level-set model which moves toward the data given by the range maps while undergoing a second-order flow to enforce the prior. After the rate of deformation slows to below some threshold, the resolution is increased, the volume resampled, and the process repeated. The coarse-to-fine strategy is intended to be a continuation method (as described by (Nielson 1997, Snyder, Han, Bilbro, Whitaker and Pizer 1995)) for both reducing computation and preventing the algorithm from converging on local minima.

There are several additional considerations which affect the performance of this algorithm. The first consideration is that the MAP formulation in Section 3 suffers from a problem; it ignores the nonlinearity resulting from the fact that the range scanner gives the depth reading for the *single closest surface point* along the line of sight. Therefore the solution given by the zero sets of $g(\cdot)$ could contain artifacts that result from surfaces interacting at occlusion boundaries. The use of $r_{\max}$ or the windowing function $\omega(\cdot)$ help alleviate this problem, but it is virtually impossible to stop this interaction in places where the object has high curvature near its occluding contour. An iterative scheme can eliminate interactions of surfaces near occluding contours by taking advantage of the knowledge that the objects are solids. That is, a surface cannot return a range reading if it is facing away from the scanner. If the surface normal is taken to be outward, then the dot product of the surface normal and the line of sight must be negative. The evolution equation, modified to include only those surface that face the scanner associated with a particular range map, is

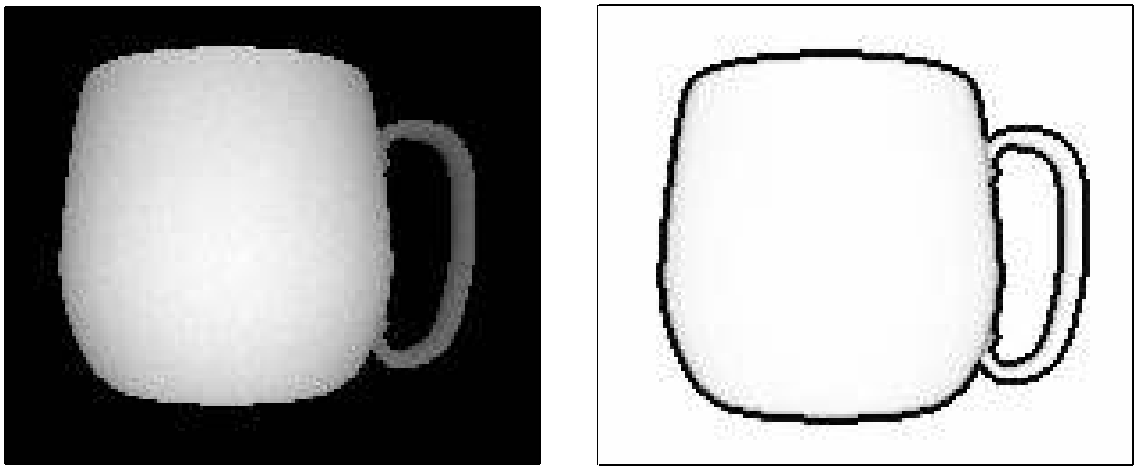
$$\frac{\partial \phi(\mathbf{x},t)}{\partial t} = |\nabla \phi(\mathbf{x})| \sum_j D^{(j)}(\hat{\mathbf{x}})\, \omega\left(D^{(i)}(\hat{\mathbf{x}})\right)\, \gamma^{(j)}(\hat{\mathbf{x}})\, c^{(j)}(\hat{\mathbf{x}}) \frac{\left(\nabla\phi \cdot \mathbf{n}^{(j)}(\hat{\mathbf{x}})\right)^+}{\nabla\phi \cdot \mathbf{n}^{(j)}(\hat{\mathbf{x}})} + \tag{37}$$

$$\beta\left(\phi_x^2+\phi_y^2+\phi_z^2\right)^{-\frac{1}{2}}\left[\left(\phi_y^2+\phi_z^2\right)\phi_{xx} + \left(\phi_z^2+\phi_x^2\right)\phi_{yy} + \left(\phi_x^2+\phi_y^2\right)\phi_{zz} - 2\left(\phi_x\phi_y\phi_{xy} + \phi_y\phi_z\phi_{yz} + \phi_z\phi_x\phi_{zx}\right)\right], \tag{38}$$

where $\mathbf{n}^{(j)}(\mathbf{x})$ is the line of sight from a range finder to a 3D point, $\mathbf{x}$.

A second practical consideration is the confidence associated with the range data. In Section 3 the confidence measure $c^{(j)}(\mathbf{x})$ was expressed as the inverse of the variance of the sensor noise associated with the range measurement along the line of sight $\mathbf{n}(\hat{\mathbf{x}})$. Other factors also affect this confidence metric. One such factor is the uncertainty in the position of the model that results from the discretization of the embedding ϕ. More specifically, each grid point being updated in the volume u, controls the movement a of a level surface nearby, whose position is known to only finite accuracy. Thus there is cloud of uncertainty around each grid point. The first-order approximation to the level-set position described in Section 5.2 improves matters considerably, but the 3D positional error cannot be discounted entirely. For this work we assume that uncertainty is isotropic with variance denoted $\sigma_{\mathbf{x}}^2$.

The positional error associated with the discrete sampling of a volume gives rise to an uncertainty about the line of



(a) (b)

Figure 9: The range map of a mug, viewed as an image (a), gives rise to a confidence measure (b) that combines both the noise of the scanner with the spatial uncertainty of the model.

sight and thereby uncertainty about the value of the associated range measurement. Using a first-order approximation to calculate this error we have

$$c^{(j)}(\mathbf{x}) = \frac{1}{\left(\sigma^{(j)}(\mathbf{x})\right)^2 + \left|\nabla D^{(j)}(\mathbf{x})\right|^2 \sigma_{\mathbf{x}}^2(\mathbf{x})}, \tag{39}$$

where $\sigma^{(j)}(\mathbf{x})^2$ is the variance associated with the range measurement along the line of sight that passes through $\mathbf{x}$ and $D^{(j)}(\mathbf{x})$, as in equation (13), is the signed distance along the line of sight between the range measurement and $\mathbf{x} \in \mathbb{R}^3$. Derivatives of $D^{(j)}(\mathbf{x})$ combine the geometry of the scanner with derivatives of the range map. That is,

$$\nabla D^{(j)}(\mathbf{x}) = \nabla \mathbf{x} - \nabla r^{(j)}(\mathbf{x}) = \nabla \mathbf{x} - \frac{\partial r^{(j)}(\theta,\varphi)}{\partial \theta}\nabla\theta(\mathbf{x}) - \frac{\partial r^{(j)}(\theta,\varphi)}{\partial \varphi}\nabla\varphi(\mathbf{x}). \tag{40}$$

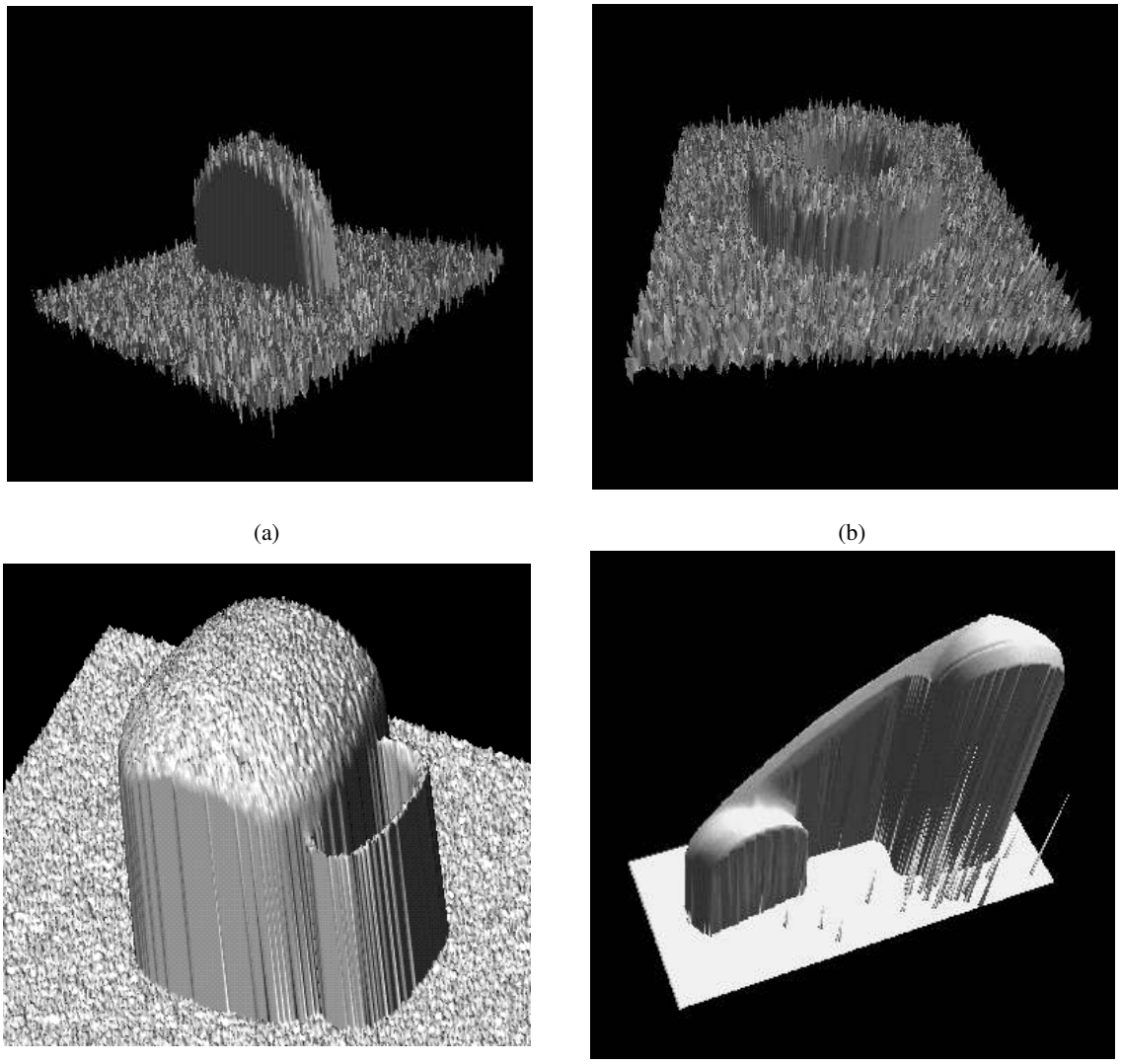
Thus, this formulation has the intuitive result of lowering the confidence near step edges in the range map. Some simple reasoning gives approximate values for $\sigma_{\mathbf{x}}$. If the surface position is known to within one half of a voxel (and voxels have unit length), then one should choose $\sigma_{\mathbf{x}}^2 = 1/12$. If the first-order approximation reduces that by an order of magnitude we have $\sigma_{\mathbf{x}}^2 = 1/120$. Figure 9a shows a noisy range map generated from the CAD model of a mug, and figure 9b shows the resulting confidence map with $\sigma_{\mathbf{x}}^2 = 1/120$ and a constant noise value for all range measurements of 1 unit.

6.1 Results

This section presents 3D reconstructions from several different sets of data. The first is synthetic data of a torus, shown as a surface in Figures 10(a) and 10(b). These 200×200 pixel depth maps are computed analytically and corrupted with 20% uncorrelated Gaussian noise. Six such views of the torus are combined in the examples that follow.

The second set of data is ten synthetic views computed from a CAD model of a mug as shown in figure 10(c). This mug has some smaller features such as the handle (particularly where it attaches to the body) and the top of the rim. Results will be shown with and without 50% additive Gaussian noise.

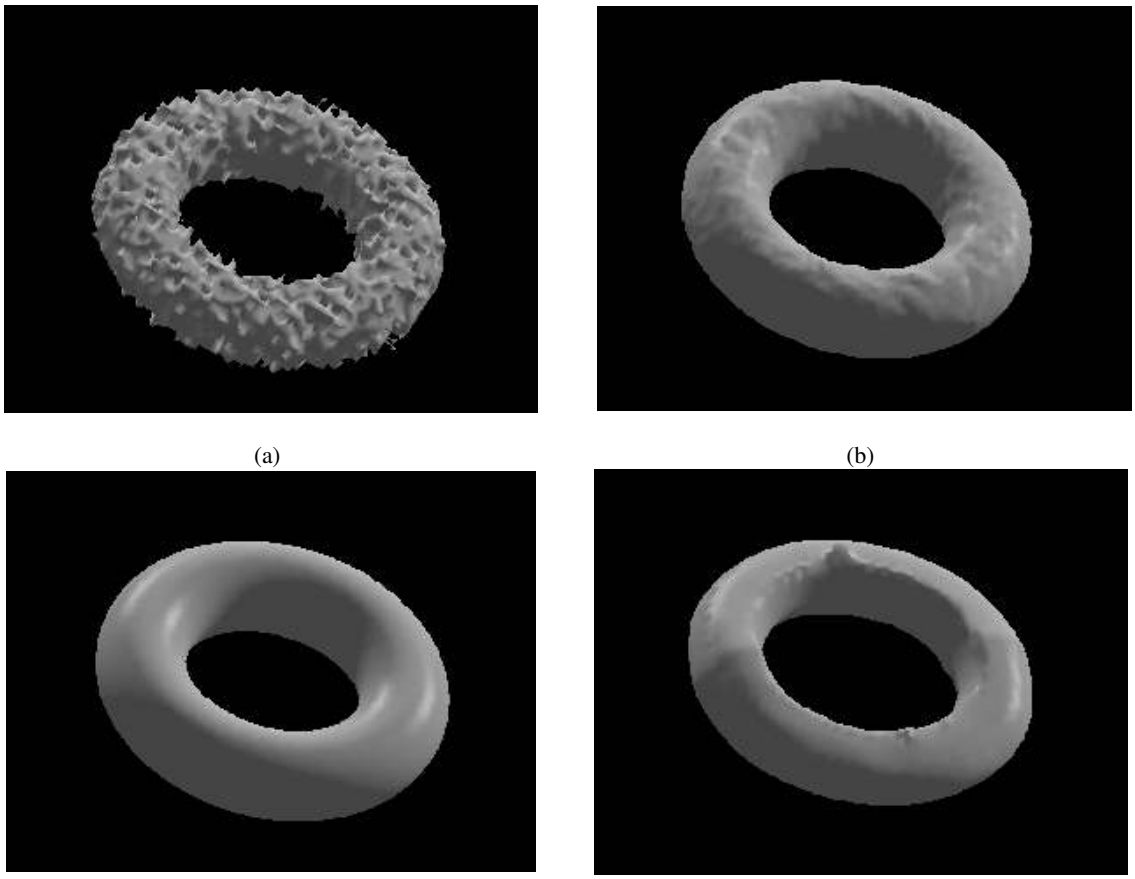
For real data, figure 10(d) shows a 400×500 pixel depth map taken from a telephone receiver using a triangulating laser range finder. Eight such views have been positioned relative to each other, initially by hand and then by an automated surface matching technique (Turk and Levoy 1994).



(a) (b)

(c) (d)

Figure 10: Range maps: Synthetic range data of a torus — 200×200 pixels with 20% Gaussian white noise (as a fraction of smaller diameter) taken of both end (a) and side (b). Synthetic range data of a mug (c) — 256×256 pixels with 50% Gaussian white noise (as a fraction of handle width). Triangulating laser range data of telephone (d).



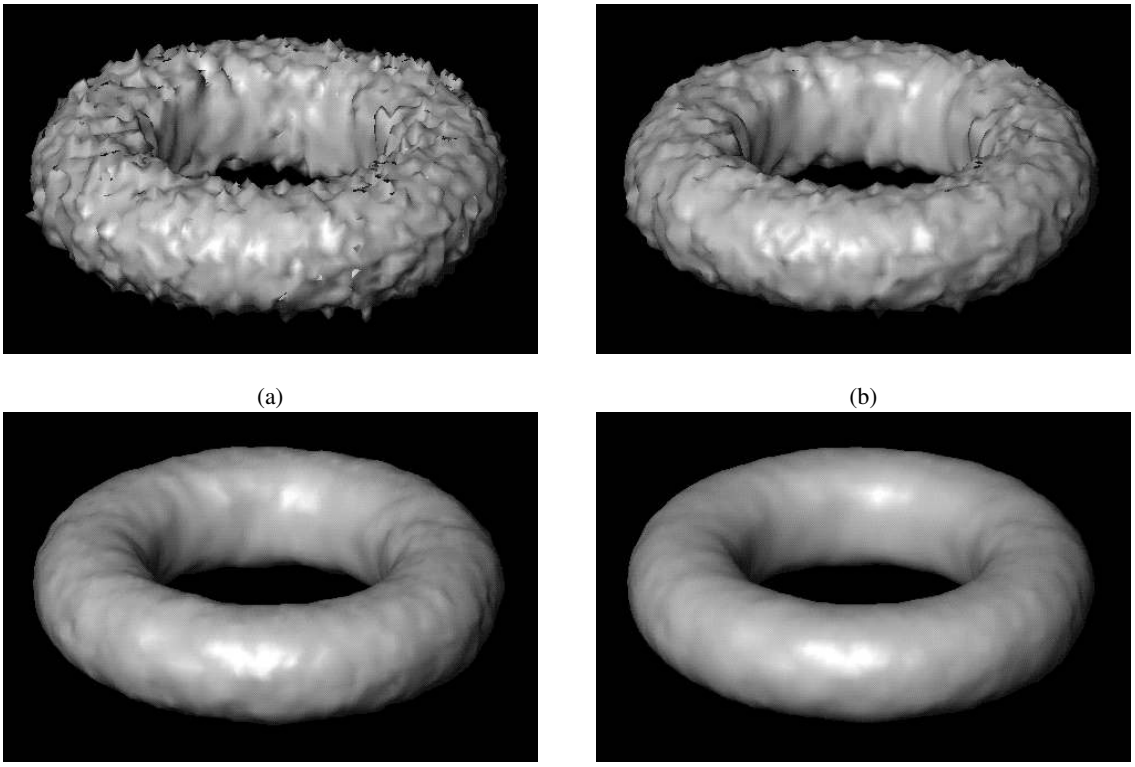
(a) (b)

(c) (d)

Figure 11: An initial model (a) is constructed by combining six points of view of a torus using the zero crossings of $g(\cdot)$. The model which is attracted to the range data but undergoes internal forces evolves and settles into a smoother steady state (b) which resembles the uncorrupted torus (c). Image-based smoothing produces view-dependent artifacts (d).

Figure 11(a) shows the initial model used for fitting the level-set model to the range data for the torus. The initial model is a $80\times80\times40$ voxels and is produced by finding the zero crossings of the data term, $g(\cdot)$. For all of the examples of this paper, 3D surface renderings of level sets are from polygonalized surfaces obtained from the volumes by the marching cubes method (Lorenson and Cline 1982). Figure 11(b) shows the result of the level-set model that uses 11(a) as an initial state and β value of 0.25. The resulting model, which combines the six points of view and the smoothing function, is a reasonable reconstruction of the original object (figure 11(c)). Figure 11(d) shows the result of the combined data term $g(\cdot)$ created by the six noisy torus range maps that have been smoothed with Gaussian blurring prior to their combination. Such an image-space blurring distorts the geometric structure of the range maps so that they no longer fit together; this results in view-dependent artifacts. The object-based smoothing, achieved by incorporating the prior of equation (19) into the level-set approach, does not produce such artifacts.

Figure 12 shows the effects of the parameter β on the final result. The parameter is not a threshold that must be set precisely; results change gradually as β varies over an order of magnitude. Figure 13 shows, for different values of β, the rms error of the reconstruction of the torus as a percentage of the magnitude of the noise added to the synthetic range maps. Even without the prior, the averaging obtained from the MAP method overcomes much of the initial noise in the data. The quality of the reconstruction does depend on the choice of β, but the use of the prior



(a) (b)

(c) (d)

Figure 12: Solutions to reconstruction described in figure 11 with values of β at (a) 0, (b) 0.04, (c) 0.32, and (d) 0.64.

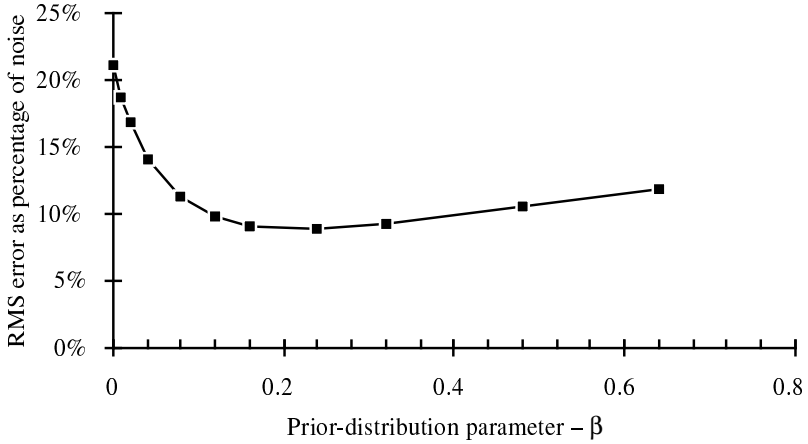


Figure 13: The rms error of the reconstruction of a torus, given as a percentage of the magnitude of the noise, for different values of the prior, β.

provides significant improvements in the reconstruction for any choice of beta that is within an order of magnitude of the optimum.

A 3D model of a mug, shown in figure 14(a), is used to generate 10 range maps. These ten views are then combined to reconstruct the 3D model (figure 14(b)). The reconstruction resembles the original in figure 14(c) to within the accuracy afforded by the discretization of the volume and some small shadowing artifacts around the handle, where it is difficult to completely map because of self occlusions. The final model is $140 \times 140 \times 140$ voxels. At this resolution the handle is less than two voxels thick, but one can still recover the fine details where the handle attaches to the main part of the mug. The facets are in the original, polygonal model. Such large volumes (almost 3 million voxels) necessitate the use of the sparse-field approach, which visits the entire data set only during the initialization, and visits only those voxels that lie near the surface thereafter. With no priors (i.e. $\beta = 0$) and range images that are corrupted by additive Gaussian noise, the surface takes on an rough appearance. Figure 14(d) shows how a small influence of the prior, $\beta = 0.5$, produces a smoother result. The result that includes the smoothing prior is also quantitatively better as determined by the rms error from the original mug model. The $140 \times 140 \times 140$ grid used for the mug required 16 iterations. The entire reconstruction process lasted about 20 minutes on a Sparc 10. Most of that time was spent on the initialization and resampling which requires visiting the entire volume.

For the workstation used in this work, this large model is at the limit of what can be stored in random access memory. As a result, models larger than this typically introduce thrashing and significantly longer computation times. Methods for efficiently representing sparse volumes are well documented in the literature, but access time penalties associated with current technologies (e.g. oct-trees) would make such methods inefficient for the iterative procedures used in this work. Reducing these memory requirements while maintaining run-time efficiency is an area of ongoing investigation.

Figure 15 shows how the algorithm is able to handle outliers. Figure 15(a) shows a single scan from the same mug data set corrupted with 1% replacement noise (i.e. outliers) instead of additive Gaussian noise. The resulting initialization shown in 15(b) is quite poor; it contains hundreds of holes. The combination of smoothing and sub-voxel accuracy obtained from the level-set models pulls the model away from the outliers and gives a convincing reconstruction.

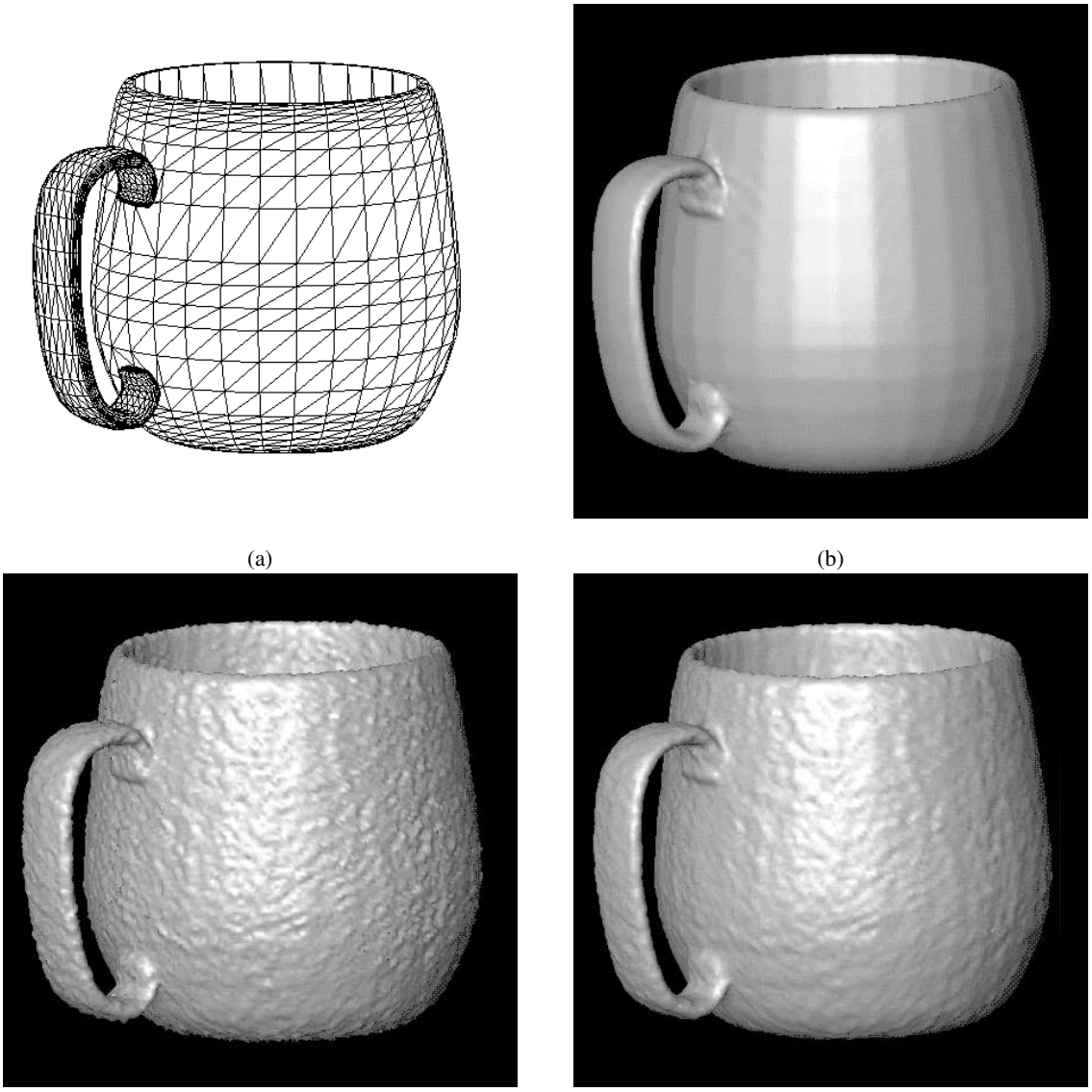
In figure 16 eight range maps of a telephone (taken with a triangulating laser range finder (Curless and Levoy 1995)) are combined to create a 3D reconstruction. Figure 16(a) shows the $40 \times 20 \times 20$ initial model used for fitting level-set models to the range data. That model serves as the initial conditions for the evolution given by equation (39). After the model settles down (change from one iteration to the next drops below a threshold), the volume is resampled onto a finer grid, and the process is repeated. The scanner images have artifacts which affect the result of the modeling as shown in figure 17(a). These artifacts result from false surfaces in the data, as well as misalignments in the range maps. The MAP reconstruction algorithm as it stands cannot realign the data, but to the extent these artifacts have a high-frequency character to them, they can be controlled by adjusting the smoothing parameter, β.

7 Conclusions

This paper has presented a strategy for reconstructing 3D objects by combining range maps taken from different points of view. The strategy is to compute the surface which is mostly likely to have given rise to the data generated by the range scanner; it maximizes the posterior probability of the surface.

By using the independence of the sensor noise and assuming that the collection of data in any one map is dense relative to the object structure, the MAP formulation converts the error from individual range readings into a volume integral. The Euler-Lagrange of that volume integral gives the equation of motion for a surface that seeks to minimize its likelihood conditional on the data. The introduction of a prior leads to a full MAP formulation. This is a fundamental result in 3D reconstruction which poses surface reconstruction as an evolutionary process based on first principles.

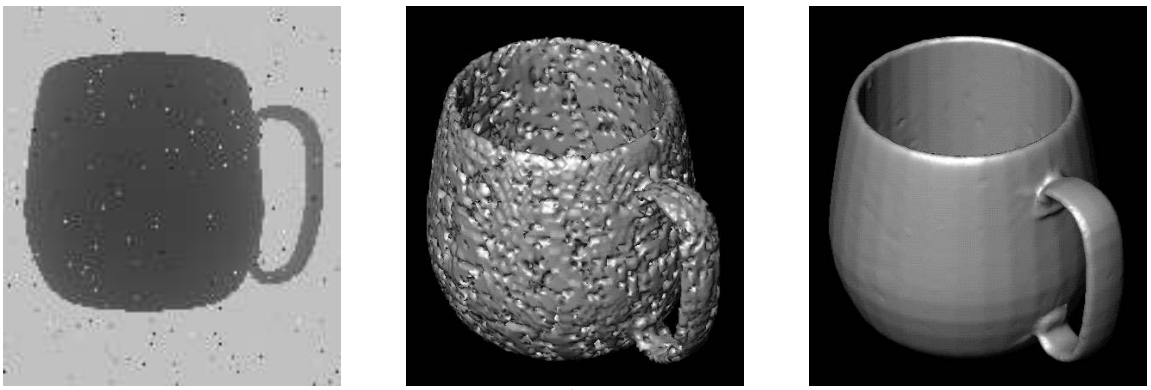
This paper has also proposed level-set models for computing the deformations associated with MAP estimation. The level-set modeling approach is well suited for the MAP reconstruction formulation because it offers flexible



(a) (b)

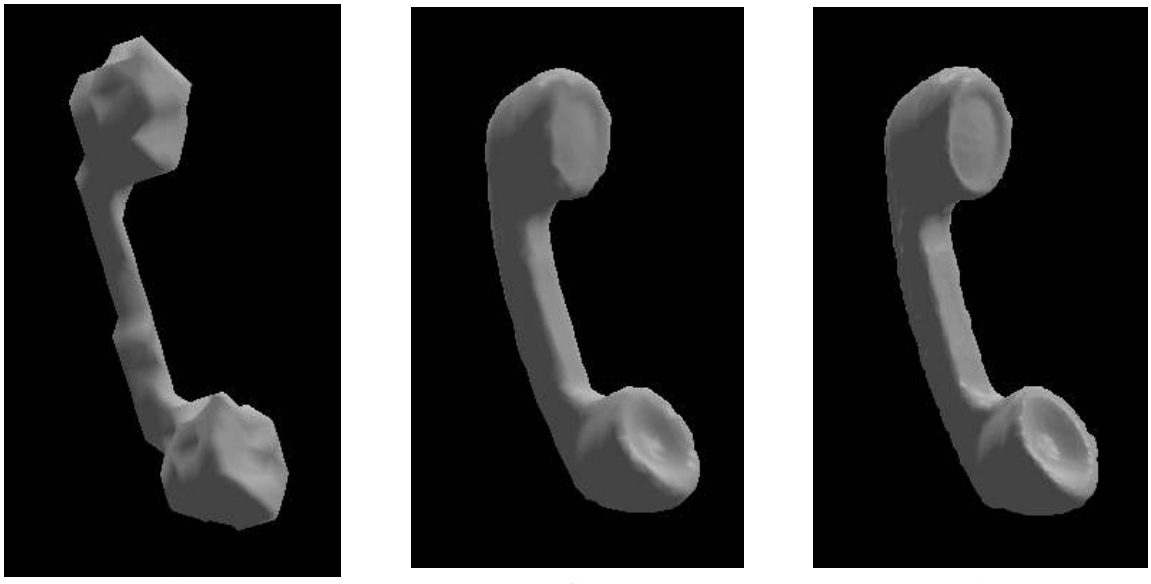
(c) (d)

Figure 14: Level-set MAP reconstruction of a mug using synthetic data generated from a 3D model (a). Without noise the reconstruction (b) is limited only by the resolution of the model ($140 \times 140 \times 140$). With noise, the surface appears rough (c). Including a prior of $\beta = 0.25$ improves the appearance of the reconstruction (d) and the rms error from the original.



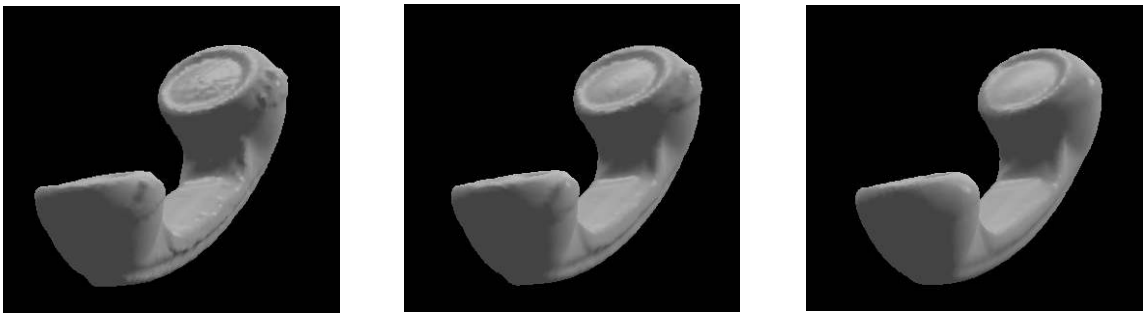
(a) (b) (c)

Figure 15: The effects of outliers: (a) The original mug data set corrupted by replacement noise. (b) The poor initialization that results from that data. (c) The results of the iterative, level-set formulation.



(a) (b) (c)

Figure 16: The results of the level-set MAP reconstruction approach: An initial model (a) is constructed by combining eight points of view of a telephone into a discrete occupancy grid of $40\times20\times20$ grid points. Results from the course model serve as initial conditions for successively finer models in (b) and (c).



(a) (b) (c)

Figure 17: Artifacts that produce discontinuities can be controlled by tuning the weight β of the smoothing term. In (a), (b), and (c), steady-state solutions with β = 0.1, 1.0, and 5.0, respectively.

topologies, multiscale/multigrid representations, many degrees of freedom, and efficient methods for calculating geometric surface properties. Despite these advantages, the level-set approach as described in the literature suffers from high computational costs and limited accuracy.

This paper has proposed an enhancement to level-set modeling technology in the form of a new algorithm, a sparse-field method. The sparse-field method performs calculations and updates only at those grid point locations that lie near the zero-level set (or any level set of interest). As a result this algorithm is computable in a fraction of the time required for other level-set approaches. This paper has shown that this new method introduces errors that are no worse than previous algorithms. The sparse-field algorithm also enables one to use a first-order approximation to the level-set position, and therefore it is actually more accurate under circumstances where the forcing function is defined to greater accuracy than the grid structure, as is the case with the proposed MAP formulation for 3D reconstruction.

Results on both real and synthetic data confirm the effectiveness of combining the MAP strategy with the level-set modeling method. The use of a very simple prior, which biases solutions toward those that have less surface area, can improve the reconstruction in the presence of uncorrelated noise and high-frequency artifacts.

There are, however, several areas that warrant further development. One area is the problem of large data sets. While the sparse-field method reduces the number of computations required for deforming, it does not alleviate the need to store data for the volume that covers the entire domain. Another area of ongoing investigation is the prior. The second-order smoothing used in this paper tends to distort the shapes of objects when applied too aggressively. This is particularly true when the objects have high curvature, such as the handle of the mug in figure 14. Priors, such as the bending energy, that rely on higher order geometry or priors that incorporate higher level knowledge about the objects could improve the quality of the results.

Acknowledgments

Thanks go to David Elsner for supplying the “mug” data set. The laser range data for the telephone was provided by the Stanford University Computer Graphics Laboratory. This work is supported by the Office of Naval Research grant N00014-97-0227 and the University Research Program in Robotics under the Department of Energy DOE-DE-FG02-86NE37968.

References

Adalstein, D. and Sethian, J. A.: 1995, A Fast Level Set Method for Propogating Interfaces, *Journal of Computational Physics* pp. 269–277.

Alvarez, L., Guichard, F., Lions, P.-L. and Morel, J.-M.: 1992, Axioms and fundamental equations of image processing, *Technical Report 9231*, Ceremade, Universite Paris-Dauphine, Place de Lattre de Tassigny, 75775 Paris Cedex 16 France.

Besl, P. J. and McKay, N. D.: 1992, A method for registration of 3-D shapes, *IEEE Trans. Pattern Anal. Mach. Intelligence* **14**, 239–256.

Bruce, J. and Giblin, P.: 1986, Growth, motion, and 1-parameter families of symmetry sets, *Proc. Roy. Soc. Edinburgh* **104A**, 179–204.

Caselles, V., Kimmel, R. and Sapiro, G.: 1995, Geodesic active contours, *in* ICC (1995), pp. 694–699.

Chen, Y. and Medioni, G.: 1992, Object modeling by registration of multiple range images, *Int. J. Image and Vision Computing* **10**, 145–155.

Chen, Y. and Médioni, G.: 1994, Fitting a surface to 3-d points using an inflating ballon model, *in* A. Kak and K. Ikeuchi (eds), *Second CAD-Based Vision Workshop*, Vol. 13, IEEE, pp. 266–273.

Chien, C.-H. and Aggarwal, J. K.: 1989, Model construction and shape recognition from occluding contours, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **11**(4), 372–390.

Curless, B. and Levoy, M.: 1995, Better optical triangulation through spacetime analysis, *in* ICC (1995), pp. 987–994.

Curless, B. and Levoy, M.: 1996, A volumetric method for building complex models from range images, *Computer Graphics (SIGGRAPH '96 Proceedings)* .

DeCarlo, D. and Metaxas, D.: 1995, Adaptive shape evolution using blending, *in* ICC (1995), pp. 834–839.

Dickinson, S., Metaxas, D. and Pentland, A.: 1997, The role of model-based segmentation in the recovery of volumetric parts from range data, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **19**(3), 259–267.

Elfes, A.: 1989, Using Occupancy Grids for Mobile Robot Perception and Navigation, *Computer* **22**(6), 46–57.

Hilton, A., Stoddart, A. J., Illingworth, J. and Windeatt, T.: 1996, Reliable surface reconstruction from multiple range images, *Proceedings from the Sixth European Conference on Compter Vision*, Springer-Verlag.

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J. and Stuetzle, W.: 1992, Surface reconstruction from unorganized points, *Computer Graphics* **26**(2), 71–78.

ICC: 1995, *Fifth International Conference on Computer Vision*, IEEE Computer Society Press.

Jain, A. and Flynn, P. (eds): 1993, *Three-Dimensional Object Recognition Systems*, Elsevier Science Publishers, The Netherlands.

Jain, R. and Jain, A. (eds): 1990, *Analysis and Interpretation of Range Images*, Springer-Verlag.

Johnson, V. E.: 1993, A framework for incorporating structural prior information into the estimation of medical images, *in* H. H. Barrett and A. F. Gmitro (eds), *Information Processing in Medical Imaging (IPMI'93)*, number 687 in *Lecture Notes in Computer Science*, Springer-Verlag, pp. 307–321.

Kass, M., Witkin, A. and Terzopoulos, D.: 1987, Snakes: Active contour models, *International Journal of Computer Vision* **1**, 321–323.

Kimia, B. J., Tannenbaum, A. R. and Zucker, S. W.: 1992, Shapes, shocks, and deformations I: The components of shape and the reaction-diffusion space, *Technical Report LEMS-105*, Division of Engineering, Brown University.

Lorenson, W. and Cline, H.: 1982, Marching cubes: A high resolution 3d surface construction algorithm, *Computer Graphics* **21**(4), 163–169.

MacDonald, D., Avis, D. and Evans, A.: 1994, Volumetric deformable models: Active blobs, *in* R. A. Robb (ed.), *Visualization In Biomedical Computing 1994*, SPIE—The International Society for Optical Engineering, Mayo Clinic, Rochester, Minnesota, pp. 160–169.

Malladi, R., Sethian, J. A. and Vemuri, B. C.: 1995, Shape modeling with front propogation: A level set approach, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **17**(2), 158–175.

McInerny, T. and Terzopoulos, D.: 1995, Toplogically adaptable snakes, *in* ICC (1995), pp. 840–845.

Miller, J., Breen, D., Lorensen, W., O'Bara, R. and Wozny, M.: 1991, Geometrically deformed models: A method for extracting closed geometric models from volume data, *Computer Graphics (SIGGRAPH '91 Proceedings)* **25**(4), 217–226.

Moravec, H. P.: 1988, Sensor Fusion in Certainty Grids for Mobile Robots, *AI Magazine* **9**(2), 61–77.

Muraki, S.: 1991, Volumetric shape description of range data using "blobby model", *Computer Graphics (SIGGRAPH '91 Proceedings)* **25**(4), 227–235.

Nielson, M.: 1997, Graduated nonconvexity by functional focusing, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **19**(5), 521–525.

Osher, S. and Sethian, J.: 1988, Fronts propogating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations, *Journal of Computational Physics* **79**, 12–49.

Sethian, J. A.: 1996, *Level Set Methods: Evolving Interfaces in Gometry, Fluid Mechanics, Computer Vision, and Material Sciences*, Cambridge University Press.

Snyder, W., Han, Y.-S., Bilbro, G., Whitaker, R. and Pizer, S.: 1995, Image relaxation: restoration and feature extraction, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **17**(6), 620–624.

Staib, L. and Duncan, J.: 1992, Boundary finding with parametrically deformable models, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **14**, 1061–1075.

Szeliski, R., Tonnesen, D. and Terzopoulos, D.: 1993, Modeling surfaces of arbitrary topology with dynamic particles, *Proceedings Fourth International Conference on Computer Vision (ICCV'93)*, IEEE Computer Society Press, Berlin, Germany, pp. 82–87.

Terzopoulos, D., Witkin, A. and Kass, M.: 1988, Constraints on deformable models: Recovering 3d shape and nonrigid motion, *Artificial Intellegence* **36**(1), 91–123.

Turk, G. and Levoy, M.: 1994, Zippered polygon meshes from range images, *SIGGRAPH '94 Proceedings*, pp. 311–318.

Whitaker, R. T.: 1994, Volumetric deformable models: Active blobs, *in* R. A. Robb (ed.), *Visualization In Biomedical Computing 1994*, SPIE, Mayo Clinic, Rochester, Minnesota, pp. 122–134.

Whitaker, R. T.: 1995, Algorithms for implicit deformable models, *in* ICC (1995).

Whitaker, R. T. and Chen, D. T.: 1994, Embedded active surfaces for volume visualization, *SPIE Medical Imaging 1994*, Newport Beach, California.

Zhang, Z.: 1994, Iterative point matching for registration of free-form curves and surfaces, *Int. J. Computer Vision* **13**, 119–152.

Appendix

A formal definition of active sets and the operations performed on them gives some general properties regarding their behavior during the sparse-field algorithm. Let u be a discrete approximation to ϕ on a rectangular grid. On each of the grid points (or voxels in 3D), u has a value equal to that of ϕ. An active set is a set of grid points that are adjacent to the level set. For notational convenience the zero-level set is used, and thus active sets consist of grid points that lie near zero crossings. As the level set moves, two things happen to active sets. First, the values of active set and nearby points change. Second, points are added and removed from the active set, i.e., activity is *passed* on from one grid point to another. The remainder of this section serves to formalize these ideas and show that active sets maintain their important properties as the sparse-field algorithm progresses. In order to maintain generality, the discussion proceeds in n dimensions, where $n = 3$ for surface reconstruction problem at hand.

A volume $I = \{X, u\}$ of dimension n is a set of grid points, X, arranged in a rectilinear grid and a discrete mapping $u : X \mapsto \mathbb{R}$. A grid point $x_j \in X$ has an associated index $i(x_j) \in \mathrm{I}^n$ (i.e., $i = i_1, \ldots, i_n$ and $i_j \in \mathrm{I}$). This index has a basis, $e_1, \ldots, e_n$, which consists of unit vectors along the grid lines, i.e., $e_1 = (1, 0, \cdots), e_2 = (0, 1, 0, \cdots), \cdots, e_n = (0, \cdots, 1)$.

Each grid point has a set of $2n$ connected neighbors which are given by $C(x_j) = \{x_k | i(x_k) = i(x_j) \pm e_m, 1 <= m <= n\}$. That is, the $2d$ neighbors of x_j, given by $C(x_j)$, are those points that lie adjacent to x_j in one of the grid directions.

One minor point is the handling of boundary conditions on the domain. For this work we assume that every grid point has $2d$ adjacent grid points, meaning either a toroidal topology (wrap around) or an infinite array of grid points. All of the results described in this section can be extended to other topologies or boundary conditions.

The *adjacent* operator, $A(x_i, x_j)$ for $x_i, x_j \in X$, indicates whether or not two or more grid points are adjacent, i.e.,

$$A(x_i, x_j) \stackrel{\triangle}{=} x_j \in C(x_i). \tag{41}$$

The sign operator $\mathrm{O}(x_i, x_j, u)$ indicates whether or not the values of u associated with two grid points have values of opposite sign, i.e.,

$$\mathrm{O}(x_i, x_j, u) \stackrel{\triangle}{=} \left(u_{x_i} \geq 0 \text{ and } u_{x_j} 0\right) \text{ or } \left(u_{x_i} < 0 \text{ and } u_{x_j} \geq 0\right). \tag{42}$$

Note that both $\mathrm{O}(x_i, x_j)$ and $A(x_i, x_j, u)$ are commutative. An adjacent grid point pair with values that have opposite signs are said to lie on a *zero crossing*.

An *active set* $\mathcal{A}$ is a subset of X such that for every adjacent pair with opposite sign, at least one of the grid points of that pair is in the active set. That is, $\mathcal{A} \subset X$ is an active set if and only if $\forall x_i, x_j \in X$, $O(x_i, x_j, u)$ and $A(x_i, x_j) \implies \{x_i, x_j\} \cap \mathcal{A} \neq \emptyset$. This definition means that an active set could include grid points that are not necessary to satisfy the condition. An active set is said to be *efficient* if for all members of the active set have a neighbor of opposite sign, i.e., $\mathcal{A}$ is efficient if and only if $\forall x_j \in \mathcal{A}\ \exists\ x_i$ s.t. $O(x_i, x_j, u)$ and $A(x_i, x_j)$. Efficient active sets consist entirely of grid points that lie on zero crossings.

There are several properties of active sets that are important. First, an efficient active set can be constructed from a volume (if it is finite) by testing the values of each grid point and its neighbors and constructing a union of all of those grid points that lie on zero crossings. Second, an efficient active set can be constructed from any (finite) active set by successively removing those grid points that do not lie on zero crossings. Third, active sets divide images into enclosed positive and negative regions. This is true because, by the definition of an active set, there is no connected path (a sequence of adjacent grid points) from a grid point with positive value to a grid point with negative value that does not pass through an active grid point.

The sparse field algorithm works with active sets and performs two distinct actions on these sets; it moves the activity of grid points to adjacent grid points, and it changes the values of grid points. The following results show that sparse-field algorithm maintains the properties of active sets (they continue to divide positive and negative regions).

A *movement* of an active set $\mathcal{A}$ is a procedure for constructing a new set of grid points $\mathcal{A}' = M(\mathcal{A}, I)$. This procedure is to remove a grid point $x_j \in \mathcal{A}$ from $\mathcal{A}$ and add to $\mathcal{A}$ all of x_j's neighbors that have opposite sign. That

is,

$$M(\mathcal{A}, I) \stackrel{\triangle}{=} \{\mathcal{A} - \{x_j\}\} \cup \{x_i | A(x_i, x_j) \text{ and } O(x_i, x_j, u)\}\,. \tag{43}$$

Proposition 1 *A set of grid points $\mathcal{A}'$ constructed by a movement procedure, $\mathcal{A}' = M(\mathcal{A}, I)$, is an active set.*

Proof Proposition 1 Let x_j denote the grid point that is removed. Assume that $\mathcal{A}'$ is not an active set. Then there exists a pair of grid points $\{x_i, x_k\}$ such that $O(x_i, x_j, u)$ and $A(x_i, x_k)$ and $\{x_i, x_k\} \cap \mathcal{A}' = \emptyset$. There are two cases to consider:

$x_j \notin \{x_i, x_k\}$ In this case neither of the grid points $\{x_i, x_k\} \cap \mathcal{A} = \emptyset$. Thus $\mathcal{A}$ is not an active set, which violates one of the assumptions.

$x_j \in \{x_i, x_k\}$ This also violates the assumption because all opposite-sign neighbors of x_j are in $\mathcal{A}'$.

□

Proposition 2 *If $\mathcal{A}$ is an efficient active set, then the set of grid points $\mathcal{A}'$ constructed by a movement procedure $\mathcal{A}' = M(\mathcal{A}, I)$, is also an efficient active set.*

Proof Proposition 2 Let x_j denote the grid point that is removed. $\mathcal{A}'$ is an active set by Proposition 1. Thus only the proof of efficiency is necessary. Assume that $\mathcal{A}'$ is not efficient, then there exists an active grid point x_i with a neighborhood $N = \{x_k | A(x_i, x_k\}$ such that $\{x_k | O(x_i, x_k, u)\} \cap N = \emptyset$. There are two cases to consider:

$x_j \in \mathcal{A}$: This violates the assumption because $\mathcal{A}$ is assumed to be efficient.

$x_j \notin \mathcal{A}'$: In this case x_j is one of the grid points that is added to $\mathcal{A}'$ by the movement procedure. However, these grid points are added because they have a neighbor of opposite sign, namely x_j.

□

An *update* of an active set is the construction of a new image which has a new value for one of its active points, i.e., $I' = \{X, u'\}$ where $u' = U(u, \mathcal{A})$.

Proposition 3 *An update of I to I' with $I' = \{X, u'\}$ and $u' = U(u, \mathcal{A})$ preserves the active set $\mathcal{A}$, i.e., $\mathcal{A}$ is still an active set of I'.*

Proof Proposition 3 Let x_j be the grid point with a value that is changed. Assume that $\mathcal{A}$ is not an active set of I'. Then there exists a pair of grid points $\{x_i, x_k\}$ such that $O(x_i, x_j, u')$ and $A(x_i, x_k)$ and $\{x_i, x_k\} \cap \mathcal{A} = \emptyset$. There are two cases to consider:

$x_j \notin \{x_i, x_k\}$ In this case neither of the grid points $\{x_i, x_k\} \cap \mathcal{A} = \emptyset$, which means that $\mathcal{A}$ is not an active set of X, u (since u and u' differ only in the value at x_j), which violates one of the assumptions.

$x_j \in \{x_i, x_k\}$ This also violates the assumption because x_j is by definition in the active set $\mathcal{A}$.

□

It is provable that the update procedure does not preserve the efficiency of an active set in all cases. Certain combinations of grid points can be shown to produce inefficient active sets after an update procedure. Pathological cases can produce dense sets of active points that fill whole regions and render the sparse-field method inefficient.

One solution to the problem of inefficiency is a *pruning* process which systematically removes unnecessary active points. A pruning procedure $P(u, \mathcal{A})$ is a procedure that produces a new active set $\mathcal{A}' = \mathcal{A} - \{x_j\}$ if the neighborhood N of x_j is of the same sign, i.e., $O(x_i, x_j, u) \forall x_i \in N$, and $\mathcal{A}' = \mathcal{A}$ otherwise. The pruning procedure can be repeated for each $x_j \in \mathcal{A}$ after an update step and can be shown to produce an efficient active set. The author has found that in practice such a pruning procedure is unnecessary, and that under normal situations active sets remain efficient except at singularities, i.e., where level sets break, join, or disappear.