SCI INSTITUTE
TECHNICAL REPORT

An End-to-End Framework for Evaluating Surface Reconstruction

Matt Berger[*], *Josh Levine*[†], *Luis Gustavo Nonato*[‡], *Gabriel Taubin*[§], *Claudio T. Silva*[†]
[*]University of Utah and Air Force Research Laboratory, USA
[†]University of Utah, USA
[‡]Universidade de São Paulo, Brazil
[§]Brown University, USA

UUSCI-2011-001

Scientific Computing and Imaging Institute
University of Utah
Salt Lake City, UT 84112 USA

January 26, 2011

Abstract:

We present a benchmark for the evaluation and comparison of algorithms which reconstruct a surface from point cloud data. Although a substantial amount of effort has been dedicated to the problem of surface reconstruction, a comprehensive means of evaluating this class of algorithms is noticeably absent. We propose a simple pipeline for measuring surface reconstruction algorithms, consisting of three main phases: surface modeling, sampling, and evaluation.We employ implicit surfaces for modeling shapes which are expressive enough to contain details of varying size, in addition to preserving sharp features. From these implicit surfaces, we produce point clouds by synthetically generating range scans which resemble realistic scan data. We validate our synthetic sampling scheme by comparing against scan data produced via a commercial optical laser scanner, wherein we scan a 3D-printed version of the original implicit surface. Last, we perform evaluation by comparing the output reconstructed surface to a dense uniformly-distributed sampling of the implicit surface. We decompose our benchmark into two distinct sets of experiments. The first set of experiments measures reconstruction against point clouds of complex shapes sampled under a wide variety of conditions. Although these experiments are quite useful for the comparison of surface reconstruction algorithms, they lack a fine-grain analysis. Hence to complement this, the second set of experiments are designed to measure specific properties of surface reconstruction, both from a sampling and surface modeling viewpoint. Together, these experiments depict a detailed examination of the state of surface reconstruction algorithms.

THE UNIVERSITY OF UTAH

An End-to-End Framework for Evaluating Surface Reconstruction

MATT BERGER
University of Utah and Air Force Research Laboratory
JOSH LEVINE
University of Utah
LUIS GUSTAVO NONATO
Universidade de São Paulo
GABRIEL TAUBIN
Brown University
and
CLAUDIO T. SILVA
University of Utah

We present a benchmark for the evaluation and comparison of algorithms which reconstruct a surface from point cloud data. Although a substantial amount of effort has been dedicated to the problem of surface reconstruction, a comprehensive means of evaluating this class of algorithms is noticeably absent. We propose a simple pipeline for measuring surface reconstruction algorithms, consisting of three main phases: surface modeling, sampling, and evaluation. We employ implicit surfaces for modeling shapes which are expressive enough to contain details of varying size, in addition to preserving sharp features. From these implicit surfaces, we produce point clouds by synthetically generating range scans which resemble realistic scan data. We validate our synthetic sampling scheme by comparing against scan data produced via a commercial optical laser scanner, wherein we scan a 3D-printed version of the original implicit surface. Last, we perform evaluation by comparing the output reconstructed surface to a dense uniformly-distributed sampling of the implicit surface. We decompose our benchmark into two distinct sets of experiments. The first set of experiments measures reconstruction against point clouds of complex shapes sampled under a wide variety of conditions. Although these experiments are quite useful for the comparison of surface reconstruction algorithms, they lack a fine-grain analysis. Hence to complement this, the second set of experiments are designed to measure specific properties of surface reconstruction, both from a sampling and surface modeling viewpoint. Together, these experiments depict a detailed examination of the state of surface reconstruction algorithms.

1. INTRODUCTION

Over the past two decades there has been an immense amount of effort dedicated to the problem of surface reconstruction. The problem of surface reconstruction may be formulated as follows: given a sampling of points measured on a surface, recover the *original* surface from which those points came. This problem is motivated by a large number of applications. For instance, surface reconstruction is a crucial first step in the recovery of non-rigid motion of time-varying geometry [Sharf et al. 2008; Li et al. 2009], and used as "ground-truth" data for multi-view stereo reconstruction evaluation [Seitz et al. 2006].

The generality of the problem has given rise to a wide variety of surface reconstruction algorithms. The distinctions in the various reconstruction algorithms hinge on the expected form of the input point data and output reconstructed surface. The input may be a single depth image, a registered point cloud, or a registered point cloud equipped with normals. Moreover, the modality of the point data plays a major role in reconstruction, where various modalities from the 3D vision literature include optical laser scanners, structured lighting, structure from motion, and photometric stereo.

The form of the output can be broken down into two main components: surface representation and the dependency on the input data. The surface representation may be a parametric surface, an implicit surface, and a triangulated surface mesh. The dependency on the input data can range from interpolating all of the input data, interpolating only a subset of the input, or simply approximating the input.

The focus of this work is on the evaluation and comparison of surface reconstruction algorithms which take as input a registered point cloud equipped with normals and output a triangulated surface mesh which approximates the input data. More specifically, we focus on input data acquired via triangulation-based scanning, wherein normals are absent and must be computed from the points themselves. This class of input is extremely broad, and quite common in point cloud data due to the rising ubiquity of triangulation-based scanners such as optical laser scanners. This class of output is very flexible for surface reconstruction, in that triangle meshes are capable of representing surfaces of arbitrary detail, while the approximation requirement allows for much freedom in reconstruction from point clouds containing large imperfections.

Despite the vast amount of work in this class of algorithms, to date there has been an insufficient means of evaluation. These algorithms typically operate on acquired scan data, where there exists the lack of a computational representation of the surface from which the scanned points were measured. Hence it is not possible to compare the reconstructed surface to the original surface, and it is quite common for such approaches to instead provide a visual comparison. Quantitative measures are typically done using synthetically generated data, but existing quantitative evaluation approaches contain a number of shortcomings, ranging from the representation of the reference shape, to how sampling is performed.

Our benchmark for surface reconstruction rectifies these deficiencies in evaluation, providing the following contributions:

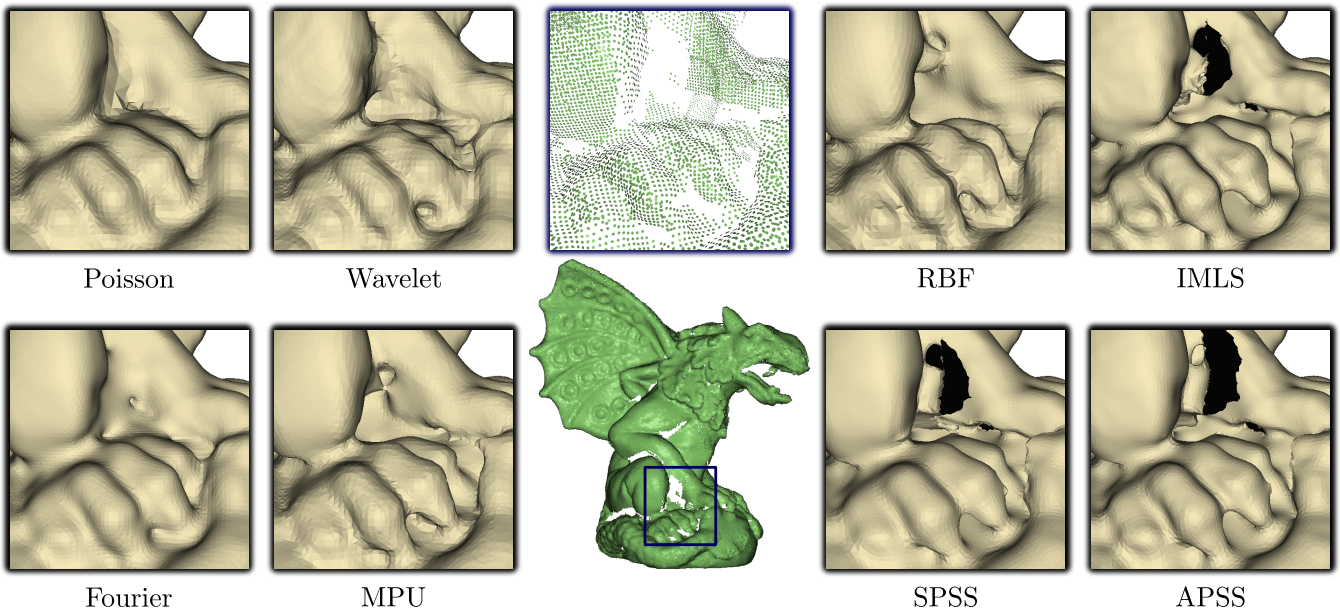


Fig. 1. Here we have synthetically sampled the Gargoyle model, and ran eight separate reconstruction algorithms on this point cloud. Note the differences between the algorithms on the claw, where some algorithms oversmooth the data, while others result in spurious holes being produced. Our benchmark aims to generate such imperfect point cloud data and study these various forms of error.

—**Realistic data.** We utilize a collection of both simple and complex shapes, where an implicit surface is used as the computational representation. We then synthetically scan the implicit surface to provide a collection of point clouds, where our scanning simulation is validated against real data.

—**Accuracy.** By employing implicit surfaces we have a precise means of performing evaluation, in both positional and differential measures. We utilize particle systems to uniformly sample both the implicit surface and the reconstructed surface mesh, thereby minimizing any potential bias of measure from the corresponding triangulation.

—**Comprehensiveness.** The set of experiments depicts a broad range of behavior across surface reconstruction algorithms.

Part of the difficulty in establishing a comprehensive set of experiments is the large variability in point clouds. Under triangulation-based scanning, a surface may be sampled under a wide variety of conditions, producing point clouds containing many different characteristics such as noise, outliers, nonuniform sampling, and missing data. This variablity is only further enhanced when scan data must be processed to produce an oriented point cloud, where registration and normal orientation must be performed. With all of these factors considered, it is difficult to determine the effectiveness of a surface reconstruction algorithm operating on an arbitrary point cloud; see Figure 1 for an illustration. In light of this, our experiments are broken down into two sets, one giving a macro view of reconstruction, and the other giving a micro view.

The first set of experiments samples a small number of complex shapes under a large variety of scanner settings. A point cloud for a given a shape provides us with a number of evaluation measures, and we aggregate each measure over all point clouds to provide an *error distribution* for a given shape. This serves two purposes. First, it provides a more objective means of comparing algorithms by looking at their performance over a distribution, rather than a single point cloud which could easily bias a certain class of algorithms. Secondly, it illustrates just how effective an algorithm is at reconstructing a single shape as a whole, given that a shape may be sampled in an unbounded number of ways.

The second set of experiments complements the first, by measuring algorithmic performance in the presence of specific sampling and shape properties. In order to achieve this, we operate in a controlled setting, and as such complex shapes are inappropriate since they may obfuscate the specific properties we are trying to measure. Hence we use a set of simple shapes, some strictly smooth and others containing sharp features, each sampled in ways to elucidate specific properties for carefully examining surface reconstruction algorithms.

Lastly, we have made our dataset and benchmark code available to the public (at: `http://www.cs.utah.edu/~bergerm/recon_bench`). Our experiments should benefit the surface reconstruction research community twofold. The first set of experiments may be used to obtain an immediate comparison across reconstruction algorithms, while the second set of experiments should prove useful for one to observe specific algorithmic behavior. Combined, our benchmark provides a comprehensive insight into this class of surface reconstruction algorithms.

2. RELATED WORK

Surface Reconstruction: Broadly speaking, we may classify surface reconstruction algorithms by their expected input and the type of output they produce.

One class of algorithms takes as input an unoriented point cloud and produces an *interpolating surface* in the form of a triangulation that uses a subset of the input points as vertices. Often these "connect-the-dots" algorithms are filtration-based techniques; they first build a triangulation with more elements than needed

and then prune away triangles not near the surface. By using the Delaunay triangulation coupled with modeling the point cloud as an ε-sample [Amenta and Bern 1999], many of these algorithms come with provable guarantees regarding the quality of the reconstruction. Extensive research efforts have been devoted to this model, producing the Cocone [Amenta et al. 2002] and Power Crust [Amenta et al. 2001] algorithms. Many other extensions have been compiled in a recent survey [Cazals and Giesen 2006] and monograph [Dey 2007].

Restricting the reconstruction to have vertices only on the input point cloud can often be limiting when the data is non-uniform, incomplete, or noisy. Algorithms that build *approximating surfaces* give a flexible alternative in these situations. Here the output is often a triangulation of an isosurface of a best-fit implicit function of the input. Many of these algorithms [Hoppe et al. 1992; Boissonnat and Cazals 2002] compute a distance field by estimating the tangent plane at every point and computing closest distance using these tangent planes. However, normal estimation in the presence of imperfect data is a difficult problem, despite recent advances [Mitra and Nguyen 2003; Dey et al. 2005].

Surface approximation from oriented point sets has gained recent attention. Approaches range from computing an indicator function [Kazhdan 2005; Kazhdan et al. 2006; Alliez et al. 2007; Manson et al. 2008], to locally fitting functions and moving least squares methods [Alexa et al. 2001; Ohtake et al. 2003; Ohtake et al. 2005b; Fleishman et al. 2005]. These approaches are well-equipped at handling various imperfections in the data, and provide for an interesting class of algorithms to study for comparison and evaluation.

Reconstruction Evaluation: In the area of surface reconstruction evaluation, most of the above approaches employ qualitative methods for evaluating the effectiveness of their approach compared to other algorithms. This usually takes the form of a visual comparison. However, significantly less work has been devoted to obtaining *quantitative* measures. This is due to the common use of scan data, where there is no longer a computational representation of the shape. For synthetic data, the works of [Kazhdan 2005; Manson et al. 2008; Sussmuth et al. 2010] take a triangle mesh as ground truth, and randomly sample the triangles directly to obtain a point cloud. This form of sampling, however, does not reflect the type of data obtained from scanned data which is consequently organized into a point cloud. The works of [Hoppe et al. 1992; ter Haar et al. 2005] obtain synthetic scans of a triangle mesh from ray tracing or z-buffering the mesh. While these methods may produce realistic data under the assumption of completely clean data, in order to replicate common scan artifacts these approaches are insufficient. Our approach also generates synthetic range data, but is more complete since we simulate an optical triangulation-based scanner.

A drawback of all of these approaches is the use of a triangle mesh as ground truth. Sampling a triangle mesh, either directly or through synthetic scans, may produce "faceted scans", where multiple samples lie on a single triangle. This can be misleading for reconstruction algorithms, as the reconstruction may preserve these faceted portions. It is also problematic to use a triangle mesh as ground truth for comparing surfaces. METRO [Cignoni et al. 1998] has become quite common for comparing two triangulated surface meshes, however for surface reconstruction we are more interested in seeing how well a reconstructed surface compares to a real shape which is smooth, not necessarily a faceted approximation. Moreover, if we are interested in comparing differential qualities, we have an ill-posed definition of surface normals when using a triangle mesh as ground truth.

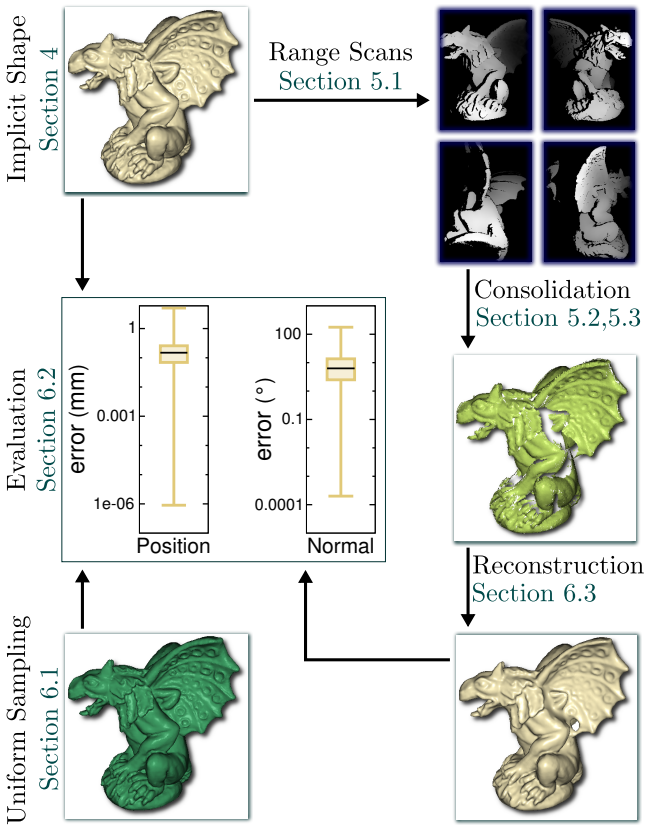


Fig. 2. Overview of our benchmark. First we create an implicit representation of a surface mesh. We then sample this implicit surface by synthetically scanning the shape to obtain individual range scans, and consolidate the scans into a single oriented point cloud via registration and normal estimation. We run a reconstruction algorithm on this oriented point cloud, and compare this output to the implicit model and a dense uniform sampling of the implicit shape to obtain quantitative results.

Similar Benchmarks: Finally, related benchmarks exist in the area of 3D stereo reconstruction, such as for binocular stereo [Scharstein and Szeliski 2002] and multiview stereo [Seitz et al. 2006]. Both use real-world data as input to the various acquisition methods. Multiview stereo, in particular, applies VRIP [Curless and Levoy 1996] to each range scanned surface, and uses the resulting triangle mesh as the gold standard for comparison. However, as pointed out for example by [Kazhdan et al. 2006], VRIP is certainly not free of errors, and arguably *every* surface reconstruction algorithm will contain errors in the presence of imperfections in scanned data. Thus, having a clear understanding of these types of errors is crucial when using a reconstructed surface as a gold standard for comparison.

3. OVERVIEW

Our benchmark may be broken up into three main phases: surface modeling, sampling, and evaluation. See Figure 2 for the full pipeline.

We first start off with an implicit surface. In order to minimize any potential bias inherit with our implicit surface representation, we use *integrated polgonal constraints*, and approximate an implicit surface from a triangle mesh, as detailed in Section 4.

We then sample this implicit surface to obtain an oriented point cloud. We simulate the process of an optical triangulation scanner in order to produce range scans. We then slightly overlap the range scans and register them via a rigid-body registration algorithm. From the registered point cloud, we then compute and orient normals for each point, producing an oriented point cloud suitable for the class of algorithms under consideration. These steps are described in more detail in Section 5.

From the oriented point cloud, we may now run a surface reconstruction algorithm on the input. This gives us a triangle mesh, for which we compare by using the implicit surface, as well as a dense uniformly sampled point cloud of the implicit surface. We may then construct positional and normal error metrics, demonstrated in Figure 2 as individual distributions of point-to-point correspondences. This is explained in more detail in Section 6.

4. SURFACE MODELING

We model the shapes in our benchmark using implicit surfaces. Our choice of implicit representation is driven by two basic requirements: expressivity and computational efficiency. It just so happens that most of the surface reconstruction algorithms under consideration satisfy these two requirements. However, modeling implicit surfaces under point/normal constraints may introduce bias for certain algorithms.

We instead model implicit surfaces under *integrated polygonal constraints*. Namely we create smooth and piecewise-smooth implicit surfaces by approximating triangulated surface meshes, or more generally polygon soup, through weight functions integrated over polygons. The advantages of using polygonal constraints over point constraints are twofold. First, approximation from a point cloud may produce specific forms of surface features in the presence of missing data; under polygon soup, we can ensure there is no such missing data. Secondly, identification and preservation of sharp features of a polygonal mesh is far more robust than a point cloud.

4.1 Polygonal MPU

Our implicit representation is a straightforward extension of Multi-level Partition of Unity (MPU) [Ohtake et al. 2003] applied to polygon soup, with the main distinction of integrating weight functions over polygons. We use the weight function of [Shen et al. 2004], defined for a given point $\mathbf{x} \in \mathbb{R}^3$ and for an arbitrary point on a triangle t, $\mathbf{p} \in t$:

$$w(\mathbf{x}, \mathbf{p}) = \frac{1}{\left(|\mathbf{x} - \mathbf{p}|^2 + \epsilon^2\right)^2} \tag{1}$$

Here, ϵ is a smoothing parameter. We may now integrate this weight function over the entire triangle t:

$$w(\mathbf{x}, t) = \int_{\mathbf{p} \in t} w(\mathbf{x}, \mathbf{p}) \mathrm{d}\mathbf{p} \tag{2}$$

For evaluating Equation 2, [Shen et al. 2004] propose a method for numerical integration. However, we derive a closed form solution for this expression. This prevents potential numerical inaccuracies caused by a quadrature scheme, which could be particularly detrimental to having a reliable benchmark. We outline the derivation in Appendix A.

Equipped with a mechanism for integrating weights over polygons, we proceed with MPU by fitting shape functions to a triangle soup $T = \{t_1, ..., t_n\}$. We adaptively build an octree over T, where for each octree cell we associate with it a sphere whose radius is the length of the diagonal of the cell. We then gather all triangles which are contained in, or overlap the sphere, and fit a shape function to those triangles.

In practice we use linear functions for our shape functions, where for each cell i we associate the function $g_i(\mathbf{x}) = \mathbf{x}^{\mathrm{T}}\mathbf{c}_i + b_i$. For all triangles which belong to the sphere of cell i, $T_i \subset T$, we fit the shape function as follows:

$$\mathbf{c}_i = \frac{\sum_{t \in T_i} \int_{\mathbf{p} \in t} \mathbf{p}\, w(\mathbf{s}_i, \mathbf{p})\, \mathrm{d}\mathbf{p}}{\sum_{t \in T_i} \int_{\mathbf{p} \in t} w(\mathbf{s}_i, \mathbf{p})\, \mathrm{d}\mathbf{p}} \tag{3}$$

$$b_i = -\left\langle \frac{\sum_{t \in T_i} \mathbf{n}_t \int_{\mathbf{p} \in t} w(\mathbf{s}_i, \mathbf{p})\, \mathrm{d}\mathbf{p}}{\sum_{t \in T_i} \int_{\mathbf{p} \in t} w(\mathbf{s}_i, \mathbf{p})\, \mathrm{d}\mathbf{p}}, \mathbf{c}_i \right\rangle \tag{4}$$

Where n_t is the triangle normal of t and $\mathbf{s}_i$ is the center of the sphere for cell i. Although one may use higher order shape functions under polygonal constraints, such as quadrics, we found the difference to be negligible, where the main difference is that for linear functions we require a larger number of shape functions to adequately approximate T.

The octree is built such that each cell is subdivided only if the zero set of its shape function deviates sufficiently from the sphere's triangles. If the octree cell's sphere is empty to start, then we grow the radius of the sphere out until we encompass a sufficient number of triangles, and terminate the subdivision with its shape function. Once the octree construction is complete, we have a spherical covering of the space. We may then evaluate the implicit function at a point by blending all shape functions whose spheres contain that point:

$$f(\mathbf{x}) = \frac{\sum_i q_i(\mathbf{x}) g_i(\mathbf{x})}{\sum_i q_i(\mathbf{x})} \tag{5}$$

Where q_i is a quadratic b-spline function centered at $\mathbf{s_i}$.

To preserve sharp features, we follow [Ohtake et al. 2003] in detecting sharp features within a leaf cell and consequently applying CSG operations for exact feature preservation. In these cases, rather than using polygon soup we instead use a manifold triangle mesh, so that sharp features can be easily identified by observing dihedral angles. We then apply union and intersection operations on overlapping shape functions to exactly preserve the sharp feature, where we support edges and corners containing a maximum degree of four.

4.2 Benchmark Shapes

We have modeled shapes specific for our two sets of experiments. Our first set of experiments consists of complex shapes, and so we have modeled five shapes containing different types of complexities. See Figure 3 for these shapes. The Gargoyle model contains details of various feature sizes, ranging from the bumps on the bottom to the ridges on its wings. The Dancing Children model is of nontrivial topology, in addition to having many varying features, such as the rim of the hat on the left child and wrinkles in the cloth. The Quasimoto model is representative of a shape containing articulated parts, such as arms, legs, and head. Lastly, the Daratech and Anchor models are shapes which contain sharp features and small topological features such as tunnels. These topological features pose a significant challenge for scanners to adequately sample.

The second set of experiments utilizes shapes which may be sampled in a controlled manner. See Figure 4 for these shapes. The Bumpy Sphere contains smooth features at varying scales. The Spiral shape is primarily composed of a thin cylindrical feature. Lastly,

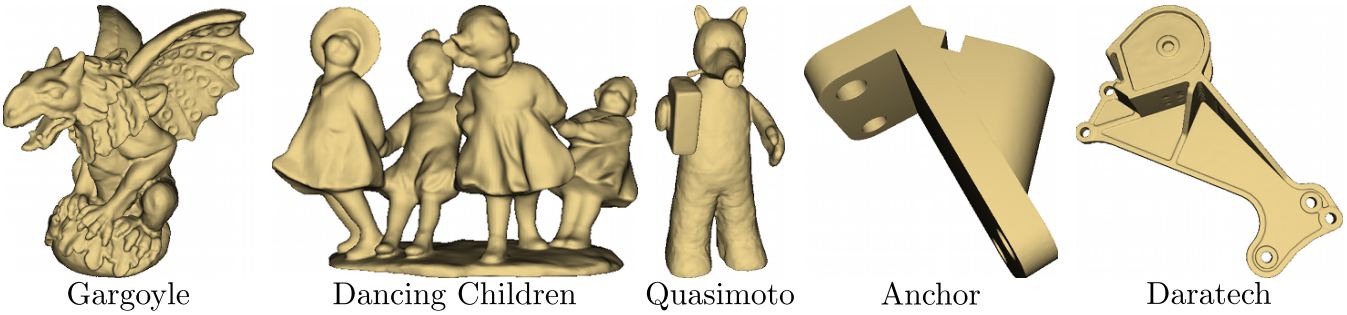


Fig. 3. Complex shapes created via our Polygonal MPU scheme. In our experiments these shapes are utilized by performing synthetic range scanning under a wide variety of typical use-case scan parameters. This class of shapes contains many interesting characteristics for scanning, such as multiple scales of detail, nontrivial topology, and sharp features.

the Mailbox consists of straight and curved sharp features alike, while also remaining simple enough to sample in a controlled setting.

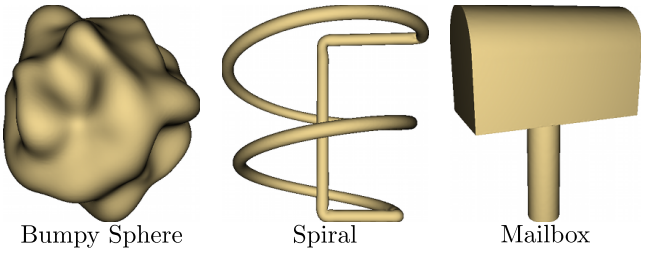


Fig. 4. Simple shapes created via our Polygonal MPU scheme. In our experiments these shapes are scanned in a precise a manner in order to replicate specific scanning difficulties, such as sparsity, missing data, and noise.

5. SAMPLING

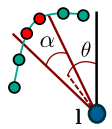
The intent of our sampling scheme is to best replicate the acquisition process of a triangulation-based scanner, in order to produce realistic point clouds. To this end, sampling is composed of three intermediate stages: synthetic range scans, registration, and orientation. Our sampling scheme aims to replicate the common properties found in scanned data, illustrated in Figure 5. See Figure 6 for an illustration of our synthetic scanner's capability in replicating such properties.

5.1 Synthetic Range Scans

We simulate the acquisition of range scans by modeling a basic optical laser-based triangulation scanning system. Such scanning systems suffer from *random error* and *systematic error*. Random error is due to physical constraints, such as noise in the laser, variations in the reflectance due to surface materials, and non-linear camera warping. Systematic error is the result of imprecise range measurement due to the peak detection algorithm. Our range scans are generated by synthesizing random error, while reproducing systematic error by performing standard peak detection.

Random Error Synthesis: We synthesize random errors by generating a series of *radiance images*, where each image is the result of a single laser stripe projection onto the implicit surface. To this end, given a pinhole camera at position $\mathbf{c}$ and a baseline configuration, we first generate the exact range data by ray tracing the implicit surface. We reject all points that are not visible from the laser position, which is a function of the baseline distance. This provides us with a set of pixels containing geometry $P = \{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_n\}$ and their corresponding points $X = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$.

We now project laser stripes onto the range geometry. We model each laser stripe projection according to a cylindrical projection, parameterized by laser position $\mathbf{l}$, field of view of the laser stripe α, and triangulation angle θ. The triangulation angle is defined with respect to an initial laser stripe plane. We may then define the laser stripe frustum as the volume enclosed by the two planes $\{\mathbf{l}, \theta - \frac{\alpha}{2}\}$ and $\{\mathbf{l}, \theta + \frac{\alpha}{2}\}$. A point is considered to be contained within the frustum if it is within positive distance to both planes. The inset depicts a 2D illustration of this configuration, where the red points of the green curve are considered to be within the laser's frustum.



For a single laser stripe, we gather all range geometry which is contained within the stripe. This consequently defines the set of "active" pixels which the laser stripe contributes to. We then determine the noise-free radiance at pixel $\mathbf{p}_i$ due to a laser stripe at triangulation angle θ by [Curless and Levoy 1995]:

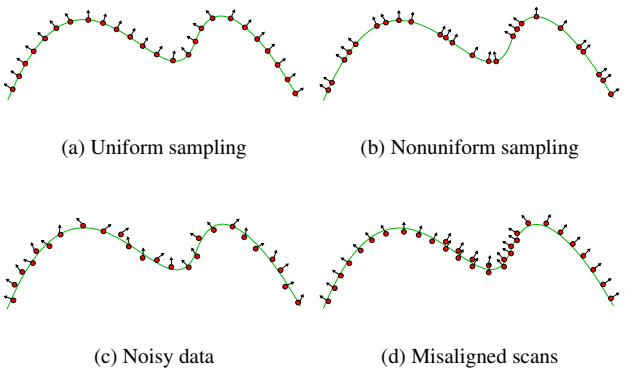
$$L_\theta(\mathbf{p}_i) = |\mathbf{n}_i \cdot \omega| e^{\frac{-2.0(d(\mathbf{x}_i))^2}{\beta^2}} \tag{6}$$

Where $\mathbf{n}_i$ is the normal of the implicit surface at $\mathbf{x}_i$, ω is the unit vector pointing towards the laser position from $\mathbf{x}_i$, $d : \mathbb{R}^3 \rightarrow \mathbb{R}$ is the closest distance to the laser frustum, and β is the width of the frustum at $\mathbf{x}_i$. Here we assume that the surface is purely diffuse, hence the BRDF is reduced to a constant factor which we omit.

In practice, diffuse surfaces suffer from noise in the form of laser speckle, where surface roughness contributes to variations in the reflectance [Baribeau and Rioux 1991]. We observe that this form of noise is more significant the further away from the center of the laser stripe frustum. We model this as additive noise sampled under a normal distribution, wherein the bandwidth is the distance away from the center of the laser stripe:

$$\tilde{L}_\theta(\mathbf{p}_i) = L_\theta(\mathbf{p}_i) + \eta\epsilon_\sigma(\mathbf{x}_i) \tag{7}$$

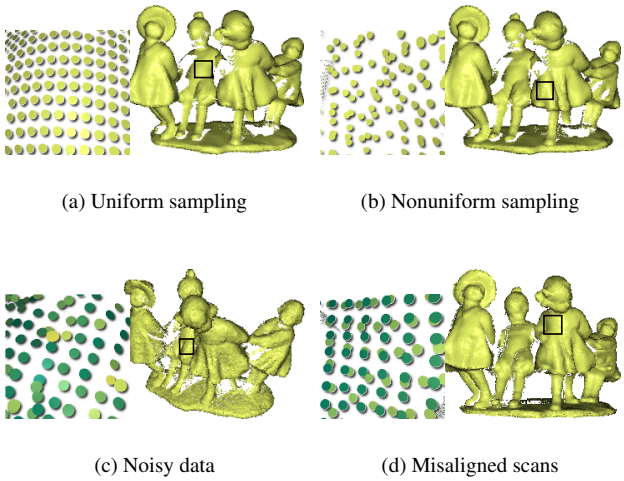
Here, η is a user-specified noise magnitude, while ϵ is a random variable normally distributed with bandwidth σ, the distance away from the center stripe. In addition, we also allow for smoothing of the noisy radiance image by convolving $\tilde{L}_\theta$ with a Gaussian kernel of a user-specified bandwidth.



(a) Uniform sampling (b) Nonuniform sampling

(c) Noisy data (d) Misaligned scans

Fig. 5. Common properties of scanned data on a sampled curve. The green curve is the true curve, while the red points are the oriented points sampled from the curve.



(a) Uniform sampling (b) Nonuniform sampling

(c) Noisy data (d) Misaligned scans

Fig. 6. Common Characteristics of 3D scans. These point clouds were generated using our synthetic scanner, illustrating our capability to replicate common scan properties. In the noise and misalignment insets we have color mapped the points by their distance away from the implicit shape, with yellow being far and green being close.

Systematic Error: For each corrupted radiance image $\tilde{L}_\theta$, we next perform peak detection in order to find each pixel's laser stripe plane. From the laser stripe plane, depth is obtained simply by triangulation. A common assumption in many peak detection algorithms is for the radiance profile, either over space or time (i.e. triangulation angle), to be Gaussian [Curless and Levoy 1995]. However in the presence of depth discontinuities, curved surfaces, and noise, this assumption is violated, resulting in range containing systematic errors.

To this end, we consider all radiance images $\tilde{L}_\theta$ defined for each triangulation angle $\theta \in \{\theta_1, \theta_2, ...\theta_m\}$, where m is the number of laser stripes. For each pixel, we consider its radiance profile as θ increases. We fit a Gaussian to this radiance profile via the Levenberg-Marquardt method. This Gaussian provides us with a mean, which determines the stripe plane, as well as a peak magnitude and variance, for which we use both as a means of rejecting unconfident range data.

Please see Appendix B for the full list of scanning parameters and common parameter settings.

5.2 Validation

It is important to verify that the range scans we are producing contain artifacts found in real scans. To this end, we validate the manner in which we generate range scans by comparing them to data acquired by commercial scanning systems. We illustrate our capability of replicating noise and missing data artifacts, which arguably have the greatest impact on surface reconstruction. We are not interested in exactly reproducing scans produced by commercial scanning systems. Most systems perform post-processing which is far beyond the scope of our scanning simulation. Instead, we show that our scanning simulation is expressive enough to generate a range of scan artifacts, while still capable of generating artifacts of a commercial scanner under proper scan parameters. To perform validation, we use the following pipeline: model implicit surface → 3D print surface → scan printed model → register scan to implicit surface → compare to our synthetic scan.

We have manufactured the Gargoyle model via 3D printing, through the company Shapeways. The minimum detail at which models may be manufactured through Shapeways is 0.2mm. From this physical model, we then scan it via an optical triangulation-based scanner, namely the NextEngine scanner. In its finest resolution mode, termed macro mode, the scanner has a maximum accuracy of 0.127mm. For shapes in which the distance from the camera is at a specified optimum, and whose normal is approximately aligned with the camera's optical axis, we found this to be true. However for a complex shape like the Gargoyle, as we will demonstrate, the accuracy can indeed vary and the noise becomes a greater magnitude then that of the shape's resolution.

To compare a real scan to a synthetic scan, we first register the real scan to the implicit surface. We perform ICP under a rigid-body deformation in order to best align the real scan to the implicit surface. As the NextEngine does not provide specifics on their CCD sensor, we take the depth image and utilize the camera calibration toolbox [Bouguet 2004] to obtain the intrinsic and extrinsic camera parameters. We feed these camera parameters in to our synthetic scanning system to obtain a comparable range scan. We note that a small non-rigid deformation might be preferable to a rigid-body deformation for registration due to small nonlinear camera deformation artifacts [Brown and Rusinkiewicz 2007]. However, this adversely impacts camera calibration and hence is unsuitable for our purposes.

Noise Validation: In our scanning simulation noise is strongly dependent on laser stripe resolution, laser stripe field of view, noise magnitude, and image smoothing bandwidth. As NextEngine does not provide these parameters for their system, to compare noise against the NextEngine scanner we have best estimated the stripe resolution, field of view, and smoothing bandwidth, while varying the noise magnitude. See Figure 7 for the comparison. Note that real scanner noise is in fact anisotropic - a function of the baseline [Abbasinejad et al. 2009]. Hence we see "bumps" which are slightly aligned with the direction of the laser projection in the NextEngine scan. Our synthetic scans demonstrate this anisotropy as well. We show that by simply tuning the noise magnitude, we are capable of producing a variety of noise profiles, wherein the NextEngine scanner is but a subset.

Missing Data Validation: Missing data in a range scan is typically the result of the rejection of unconfident range data. In our scanning simulation this is related to the peak intensity threshold,

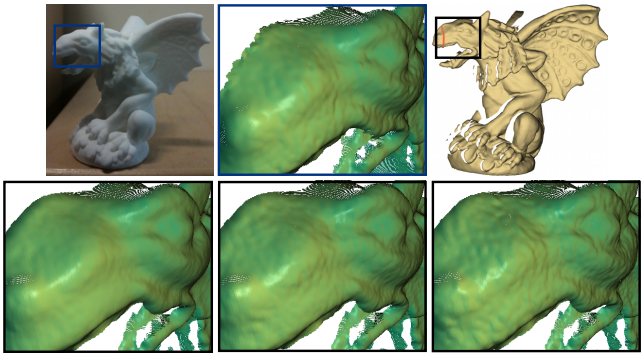


Fig. 7. Comparison of noise profiles between our scanning simulation in increasing noise magnitude (bottom), and a NextEngine scan (top-center). Note that real scanner noise takes the form of bumps aligned in the direction of the laser scan projection (top-right), and our synthetic noise is able to capture this anisotropic noise over varying noise magnitude.

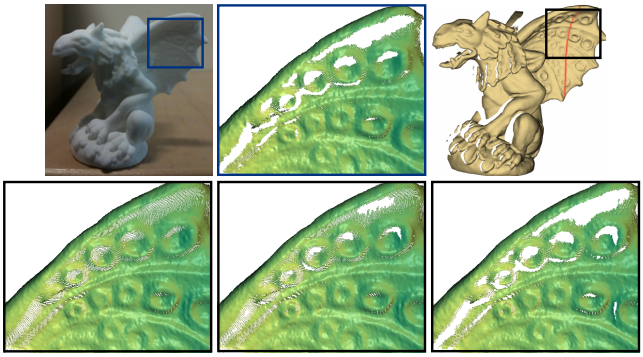


Fig. 8. A comparison of missing data between our scanning simulation in increasing peak threshold (bottom), and a NextEngine scan (top-center). Note the similarities in regions of missing data between our scan (bottom-right) and the NextEngine scan, chiefly due to the grazing angle at which laser strikes the surface, resulting in a low level of radiance.

where a small peak may indicate a poor Gaussian fit. Hence to compare missing data to the NextEngine scanner, we vary the peak intensity threshold and observe where regions of missing data exist, see Figure 8. As shown, the NextEngine scanner has a fixed threshold at which to reject unconfident range, while in our scanning system it is a tunable parameter, producing varying degrees of missing data.

5.3 Scanning and Registration

Given that we have a means of acquiring range scans, next we must determine where to scan, and register the scans. It is extremely difficult to automate the process of positioning/orienting a scanner, as this is inherently a manual process. We assume an ideal environment wherein we place the scanner at uniformly sampled positions over the bounding sphere of the object, such that the camera is oriented to look at the object's center of mass. Note that such acquisition systems are starting to gain popularity [Vlasic et al. 2009].

From these individual range scans, we next register them into a single coordinate system. First we overlap the scans by a parameterized amount. We then run the registration algorithm of [Brown and Rusinkiewicz 2007] to align the scans, which is a variant of ICP wherein a rigid-body transformation is assumed to be sufficient to align all scans. Note that the amount of overlap effectively determines the quality of the alignment less overlap means a poorer initialization, and the optimization process may hit an undesriable local minimum causing misalignment errors.

5.4 Orientation

From the registered point cloud, we must attach a normal to each point. One option is to simply use the analytical normal defined by the implicit function. However, for misaligned and noisy data, it becomes unclear what the normal should be from the implicit function. Instead, we also allow for normal orientation via the method of [Hoppe et al. 1992].

Under this method, at every point we estimate its local tangent plane via PCA, by gathering the k-nearest neighbors and extracting the eigenvectors of the covariance matrix. PCA, however, does not give orientation of the normals, and so we employ the minimum spanning tree approach of [Hoppe et al. 1992] to propagate normal directions.

We note that by using this method, we may end up with noisy tangent planes using PCA, due to a number of factors such as nonuniform sampling, noise, misalignment, and missing data. Moreover, normals may be oriented in the inverse direction due to these factors. However in certain scanning situations, we may have knowledge of the scanner positions, which can be used to properly orient the normals. Hence we allow for both options in our experiments.

6. EVALUATION

In order to evaluate the quality of a surface mesh output by a reconstruction algorithm against the input implicit surface, we take the view of *discrete differential geometry* for defining error measures. As illustrated in [Hildebrandt et al. 2006], pointwise plus normal convergence of a polyhedral surface to a smooth surface implies convergence in: the metric, surface area, and Laplace-Beltrami operator. In their context, pointwise convergence is measured in terms of Hausdorff distance and normal convergence is measured as the supremum of the infinity norm over all normals. We take their basic framework and expand it to include other error measures, in order to depict a more informative evaluation.

Following [Hildebrandt et al. 2006], for the implicit surface Ω and corresponding triangulated surface mesh M, we define the shortest distance map $\Phi\colon \Omega \to M$ for $\boldsymbol{\alpha} \in \Omega$ as:

$$\Phi(\boldsymbol{\alpha}) = \boldsymbol{\alpha} + \phi(\boldsymbol{\alpha})\mathbf{N}_\Omega(\boldsymbol{\alpha}) \tag{8}$$

Where $\mathbf{N}_\Omega$ is the normal field over Ω and $\phi\colon \Omega \to \mathbb{R}$ is the signed distance along the normal $\mathbf{N}_\Omega(\boldsymbol{\alpha})$. Hence, so long as the Hausdorff distance of Ω and M is bound by the *reach* of Ω, or the minimal radius of all medial balls. then $\boldsymbol{\alpha}$ is the point on Ω closest in distance to $\Phi(\boldsymbol{\alpha})$ [Federer 1959]. Slightly deviating from [Hildebrandt et al. 2006], we symmetrically define $\Psi\colon M \to \Omega$ for $\mathbf{x} \in M$ as:

$$\Psi(\mathbf{x}) = \mathbf{x} + \psi(\mathbf{x})\mathbf{N}_M(\mathbf{x}) \tag{9}$$

Where $\mathbf{N}_M$ is the normal field over M and $\psi\colon M \to \mathbb{R}$ is the signed distance along the normal $\mathbf{N}_M(\mathbf{x})$. Assuming Hausdorff distance is bounded by the reach of M, $\mathbf{x}$ is the point on M closest in distance to $\Psi(\mathbf{x})$. The mappings Φ and Ψ provide for *shortest distance correspondences* between points, providing a means for comparing Ω and M.

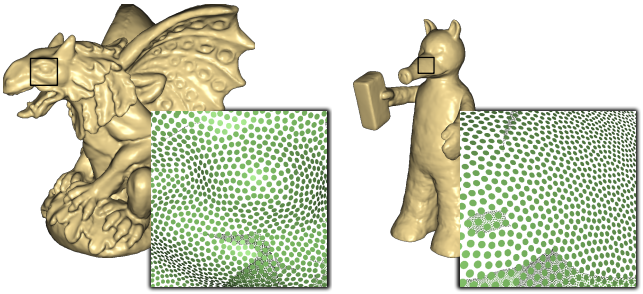
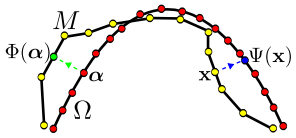


Fig. 9. Complex shapes sampled under particle systems. Note the high density and uniform distribution in the particles. Both of these properties are essential for obtaining precise error measures.



In the inset, we see that $(\Phi(\boldsymbol{\alpha}), \boldsymbol{\alpha})$ is a shortest distance correspondence, where for $\Phi(\boldsymbol{\alpha}) \in M$, its closest point on Ω is $\boldsymbol{\alpha}$. Likewise $(\Psi(\mathbf{x}), \mathbf{x})$ is a shortest distance correspondence, where for $\Psi(\mathbf{x}) \in \Omega$, its closest point on M is $\mathbf{x}$.

6.1 Sampling

In practice, we must sample Ω and M in order to obtain error measures, similar to METRO [Cignoni et al. 1998]. Ω and M must be sampled densely enough in order to have precise error measures. We argue that so long as each set of samples is an ϵ-sample for its respective surface, $\epsilon \ll 1$, then we are not missing any important features of the surface. An ϵ-sample is a function of the *local feature size* (lfs) of a surface [Amenta and Bern 1999], or the minimum distance to the medial axis for a given point on a surface. Lipschitz-continuity of the lfs ensures that for an ϵ-sample, the distance between any two points is bound by the lfs, in addition to bounding the variation in normals [Dey 2007].

Following this we sample Ω by using *particle systems* [Meyer et al. 2007], which minimize an energy functional based on inter-particle distances. We employ a uniform distribution of samples by prescribing a single inter-particle distance for all particles. This distance is specified such that it is less than the observed reach of Ω, hence providing a tight upper bound on the local feature size. See Figure 9 for uniform samplings of our complex shapes.

Similarly, we use particle systems to sample M, adapting the approach of [Meyer et al. 2007] to triangle meshes. Though one may randomly sample the mesh, or uniformly sample triangles individually as in METRO [Cignoni et al. 1998], we find that a uniform distribution is essential to achieve meshing-invariance in certain error measures, namely mean computation. Rather than specifying an inter-particle distance for M in the energy minimization, we specify the number of particles, since an output reconstructed mesh may be arbitrarily complicated.

If we denote P_Ω and P_M as the set of samples chosen from Ω and M, respectively, we build two sets of ordered pairs representing shortest distance correspondences:

$$C_\Omega = \{(\mathbf{x}, \boldsymbol{\alpha}) \mid \boldsymbol{\alpha} \in P_\Omega, \mathbf{x} = \Phi(\boldsymbol{\alpha})\} \tag{10}$$

$$C_M = \{(\boldsymbol{\alpha}, \mathbf{x}) \mid \mathbf{x} \in P_M, \boldsymbol{\alpha} = \Psi(\mathbf{x})\} \tag{11}$$

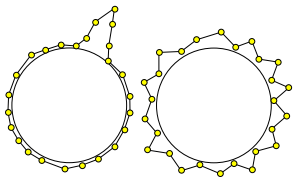
6.2 Discrete Error Measures

From here we may define a variety of discrete error measures between Ω and M. Denoting $|S| = |C_\Omega| + |C_M|$, Hausdorff distance is approximated by:

$$H(\Omega, M) = \max\left\{ \max_{(\mathbf{x},\boldsymbol{\alpha})\in C_\Omega} |\mathbf{x} - \boldsymbol{\alpha}|, \max_{(\boldsymbol{\alpha},\mathbf{x})\in C_M} |\boldsymbol{\alpha} - \mathbf{x}| \right\} \tag{12}$$

While mean distance is:

$$\mu(\Omega, M) = \frac{1}{|S|}\Big(\sum_{(\mathbf{x},\boldsymbol{\alpha})\in C_\Omega} |\mathbf{x} - \boldsymbol{\alpha}| + \sum_{(\boldsymbol{\alpha},\mathbf{x})\in C_M} |\boldsymbol{\alpha} - \mathbf{x}| \Big) \tag{13}$$



These measures depict error in very different ways, as the inset illustrates. Here the circle is the smooth shape, while the piecewise linear curve is the approximating mesh. Hausdorff distance will be large for the pair of shapes on the left, while mean distance will be rather small, whereas for the pair of shapes on the right, mean distance will be much larger than the pair of shapes on the left, while Hausdorff distance will be less.

From these shortest distance correspondences, we have a means of measuring higher-order geometric properties, by comparing differential properties at the correspondences. This is analogous to defining pullbacks on Φ and Ψ. We opt to measure normal angle deviations in a similar manner to distance measures. If we denote $\gamma(\boldsymbol{\alpha}, \mathbf{x}) = \angle(\mathbf{N}_\Omega(\boldsymbol{\alpha}), \mathbf{N}_M(\mathbf{x}))$, the maximum and mean angle deviation of point correspondences, respectively, are:

$$H_N(\Omega, M) = \max\left\{ \max_{(\mathbf{x},\boldsymbol{\alpha})\in C_\Omega} \gamma(\boldsymbol{\alpha}, \mathbf{x}), \max_{(\boldsymbol{\alpha},\mathbf{x})\in C_M} \gamma(\boldsymbol{\alpha}, \mathbf{x}) \right\} \tag{14}$$

$$\mu_N(\Omega, M) = \frac{1}{|S|}\Big(\sum_{(\mathbf{x},\boldsymbol{\alpha})\in C_\Omega} \gamma(\boldsymbol{\alpha}, \mathbf{x}) + \sum_{(\boldsymbol{\alpha},\mathbf{x})\in C_M} \gamma(\boldsymbol{\alpha}, \mathbf{x}) \Big) \tag{15}$$

In practice we take $\mathbf{N}_M$ to be triangle normals, as opposed to more sophisticated normal estimation methods [Meyer et al. 2002]. Such methods are sensitive to the triangulation and typically assume smoothness in the normal field, where in the presence of sharp features this can result in undesirable oversmoothing.

Note that when Ω and M are within a Hausdorff distance which exceeds the reach of Ω, then in certain regions Φ no longer accurately measures shortest distance (likewise for Ψ). To combat this, for $\boldsymbol{\alpha} \in \Omega$ and $\mathbf{x} = \Phi(\boldsymbol{\alpha})$, if there exists $\boldsymbol{\beta} \in P_\Omega$ which is closer in distance to $\mathbf{x}$ than $\boldsymbol{\alpha}$, we exclude the correspondence $(\mathbf{x}, \boldsymbol{\alpha})$ from C_Ω. This corresponds to the fact that Ω is more geometrically complex, locally, than M, so Φ may no longer be bijective. We would still like to associate a correspondence to $\boldsymbol{\alpha}$, and so we take the

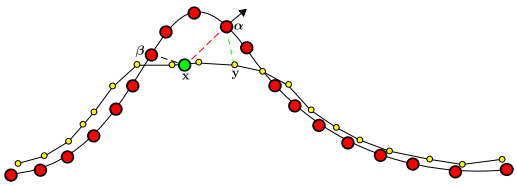


Fig. 10. A situation where the Φ mapping produces an incorrect shortest distance correspondence. The dashed red line indicates the normal line from $\boldsymbol{\alpha}$ to $\mathbf{x}$, giving us an inaccurate correspondence since $\boldsymbol{\beta}$ is closer to $\mathbf{x}$ than $\boldsymbol{\alpha}$. So we instead take $(\boldsymbol{\alpha}, \mathbf{y})$ as a correspondence.

closest point $\mathbf{y} \in P_M$ to $\boldsymbol{\alpha}$, and add $(\boldsymbol{\alpha}, \mathbf{y})$ as a correspondence to C_M. A similar situation exists for the Ψ mapping. See Figure 10 for an illustration of the case for Φ. In practice, we use exact distances whenever the above condition is not violated. We have found this violation to rarely occur, so long as the surfaces are not too far off in Hausdorff distance.

6.3 Algorithms

We have chosen a wide variety of publicly available surface reconstruction algorithms to test our benchmark against. For the sake of fair comparison, we have only used algorithms which take an oriented point cloud as input, and output an approximating surface. Here we provide a categorization and brief description of each algorithm, while also providing an abbreviation of each to help identify them in the experiments to follow.

Indicator Function: This class of algorithms reconstructs a three-dimensional solid O by finding the scalar function χ, known as the indicator function, defined in $\mathbb{R}^3$ such that:

$$\chi(\mathbf{x}) = \begin{cases} 1 & \mathbf{x} \in O \\ 0 & \mathbf{x} \notin O \end{cases} . \tag{16}$$

Where the surface Ω is then defined by ∂O. In practice, these approaches approximate χ by operating on a regular grid or an octree, and generate Ω by isosurfacing the grid.

Poisson surface reconstruction (abbr. **Poisson**) [Kazhdan et al. 2006] solves for χ by noticing that $\nabla\chi$ should agree with the normal field $\mathbf{N}$ at ∂O. This amounts to inverting the gradient operator, hence χ is found by solving the Poisson equation:

$$\nabla \cdot \nabla\chi = \nabla \cdot \mathbf{V} \tag{17}$$

Where $\mathbf{V}$ is the smoothed normal field defined throughout the volume. The Poisson equation is efficiently solved only near the surface by using an adaptive multigrid solver defined on the octree built on the point cloud. Note that use of an octree may result in limited resolution over regions of missing data.

An alternative method of constructing the indicator function is to solve for it indirectly by projecting χ onto a basis, and then performing an inverse transform to obtain χ. By invoking Stokes theorem, this projection need only be performed on ∂O:

$$\int_O \nabla \cdot \mathbf{F}(p) dp = \int_{\partial O} \langle \mathbf{F}(p), \mathbf{N}(p) \rangle \, dp \tag{18}$$

Where $\mathbf{F}$ is a vector-valued function whose divergence $\nabla \cdot \mathbf{F}$ defines the basis.

Fourier surface reconstruction (abbr. **Fourier**) [Kazhdan 2005] employs the Fourier basis as part of their solution. For efficiency they use the Fast Fourier transform (FFT), hence requiring a regular grid and the grid resolution being a power of two. However, use of a regular grid has its benefits when faced with missing data, as their is no loss of resolution.

Wavelet surface reconstruction (abbr. **Wavelet**) [Manson et al. 2008] employs a Wavelet basis for the solution of Equation 18. They show how one may use a Haar or a Daubechies (4-tap) basis. Due to the multiresolution structure of wavelets, they use an octree for the basis projection, hence similar to Poisson this method may result in limited resolution over regions of missing data.

Point Set Surfaces: Point set surfaces (PSS) are defined based on moving least squares (MLS), where a projection operator is used to define a surface by its collection of stationary points, or where the output point of the projection operator is its input point. Originally defined for unoriented points, its definition is greatly simplified when considering points equipped with normals, and may be used for surface reconstruction by considering its implicit surface definition, rather than its projection operator.

Basic PSS methods use a weighted combination of linear functions to locally define the surface at every point. Borrowing terminology from [Guennebaud and Gross 2007], we use two different definitions in our experiments: simple point set surfaces (abbr. **SPSS**) [Adamson and Alexa 2003] and implicit moving least squares (abbr. **IMLS**) [Kolluri 2005]. The implicit surface definition of SPSS is:

$$f(\mathbf{x}) = \mathbf{n}(\mathbf{x})^T (\mathbf{x} - \mathbf{c}(\mathbf{x})) \tag{19}$$

Where $\mathbf{n}$ is a weighted average of normals in a neighborhood of $\mathbf{x}$, and $\mathbf{c}$ is the weighted centroid in a neighborhood of $\mathbf{x}$. The weights used in computing the normal and the centroid are derived from a smooth, positive function $w_{\mathbf{x}}$ defined with respect to $\mathbf{x}$, which gives points closer to $\mathbf{x}$ larger influence. IMLS is defined as the implicit function:

$$f(\mathbf{x}) = \frac{\sum_i w_{\mathbf{x}}(\mathbf{p}_i)(\mathbf{x} - \mathbf{p}_i)^T \mathbf{n}_i}{\sum_i w_{\mathbf{x}}(\mathbf{p}_i)} \tag{20}$$

We note that IMLS is a weighted average of linear functions, whereas SPSS is a single linear function, whose centroid and normal is a weighted average of points and normals, respectively.

Algebraic point set surfaces (abbr. **APSS**) [Guennebaud and Gross 2007] uses spheres defined algebraically as the shape function. Rather than directly obtaining the implicit function at a point, APSS must fit a sphere to a neighborhood of points, requiring the solution of a linear least squares system for every point. By using a higher-order function, the method can be more robust to sparse data than SPSS and IMLS.

For our experiments, the software package kindly provided by Gael Guennebaud contains implementations of SPSS, IMLS, and APSS. Each PSS is evaluated over a regular grid, and the reconstructed surface is obtained by isosurfacing the zero level-set. In the software, neighborhoods used to locally fit functions are estimated at each point based on the density of the input point cloud. In the presence of missing data this method may produce an empty neighborhood, producing holes in the output. This has an impact on evaluation, which we further discuss throughout the experiments sections.

Multi-level Partition of Unity: In our own implicit surface definition we use a variant of Multi-Level Partition of Unity (MPU) applied to polygon soup, and so we refer to Section 4.1 for details about the overall approach, noting that the construction of MPU with points is quite similar to that of polygons. In our experiments we use three variants. First we use the original approach of [Ohtake et al. 2003] (abbr. **MPU**), where linear functions are used as low-order implicits. We also use the approach of [Nagai et al. 2009] (abbr. **MPUSm**), which defines differential operators directly on the MPU function, though restricted to linear functions. In doing so, diffusion of the MPU function becomes possible, resulting in a more robust reconstruction method. Lastly, we also use the method [Ohtake et al. 2005b] (abbr. **RBF**), which uses compactly-supported radial basis functions for locally-defined implicit functions in the MPU construction. For all MPU methods a surface mesh is generated by first evaluating the MPU function over a regular grid, and isosurfacing the zero level-set to obtain the surface.

Scattered Point Meshing: The method of [Ohtake et al. 2005a] (abbr. **Scattered**) is a departure from the above approaches. This method grows weighted spheres around points in order to determine the connectivity in the output triangle mesh. Quadric error functions [Garland and Heckbert 1997] are used to position points

in the output mesh, which can result in a small amount of simplification in the output. Similar to the PSS methods, regions in which data is absent may result in holes in the output.

6.4 Algorithm Parameters

We provide a brief description of the most relevant parameters for each algorithm.

Resolution: As all algorithms, except Scattered, contour a grid to obtain the surface they must contain sufficient resolution to adequately preserve all surface details. For each implicit surface we first determine the resolution which is necessary to easily extract the surface with minimal error. We find that across all shapes, a resolution of 350^3 provides for more than enough resolution to preserve surface details, hence for the PSS and MPU methods we have set their resolution to 350. For Fourier, the resolution at which to contour is also the resolution at which the FFT is applied. As it must be a power of two, we set it to 512 in order to reduce any smoothing resulting from the FFT. Poisson and Wavelet build an octree over the point cloud, and so we set their maximum depth to 10, giving them an effective resolution of 1024, depending on the input point cloud density.

Noise: Algorithms tend to handle noise according to their categorization. For indicator functions, noise may be combatted by setting a lower resolution at which to solve for the indicator function and consequently isosurface, though in practice we found them to perform best at their highest resolution. PSS methods all contain a bandwidth which determines the extent of neighborhood influence. A large bandwidth results in more points for consideration in shape fitting and hence larger data smoothing. MPU methods and Scattered all contain error thresholds for which to determine the quality of a shape fit. In the presence of noise the tolerance may simply be increased to avoid overfitting. MPUSm also provides parameters specific to their diffusion method, for which we use author-suggested settings.

Discussion: In practice we set an algorithm's parameters based on the characteristics of the input point cloud, namely the noise level. As the point clouds of experiments 7.1-7.3 contain a constant level of noise, we have kept all algorithm parameters fixed throughout these experiments. Though one may fine-tune an algorithm's parameters to improve its performance with respect to a particular error metric, parameter insensitivity is an important indication of algorithmic robustness. Only in experiment 7.4, where noise varies, do we set algorithm parameters in accordance with the noise level.

7. RESULTS

Our results are broken down into two main sets of experiments: one in which complex shapes are sampled under a variety of sampling settings, and one in which simple shapes are sampled under specific sampling settings.

We have not used the maximum angle deviation as an error measure in our experiments. By using triangle normals as the normal field over a surface mesh, this measure can be quite high even when the mesh contains low error in all other measures. As a result, in comparing algorithms we found this error measure to be rather indistinguishable, hence we have omitted it.

Note that it is possible for these algorithms to produce surfaces containing multiple connected components. We have decided to extract the largest connected component, in terms of surface area, as the surface for evaluation rather than all components. Unfortunately, this biases algorithms in which connected components are created far from the ground truth surface over algorithms which create additional components near the surface. Hence in addition to the error metrics, we have provided additional information on the algorithms including the number of connected components, as well as the length of the boundary components, whether or not the surface is manifold, deviation from the true genus, and computation time.

7.1 Error Distributions

Our first set of experiments focuses on the performance of surface reconstruction algorithms restricted to a single shape. For a single shape we sample it across a variety of scanner parameter settings, run all reconstruction algorithms across all point clouds, and compute error metrics for each point cloud. For each algorithm we then aggregate the error metrics across all point clouds to obtain what we term *error distributions*.

We argue that error distributions are more effective for benchmarking reconstruction algorithms, rather than comparing algorithms with respect to a single point cloud. Each algorithm has its strengths and flaws for particular forms of data, and to sample a shape in such a way that it caters towards the strengths of certain algorithms provides an incomplete picture in the comparison of reconstruction algorithms.

To this end we generate samples by varying scanning parameters across typical use-case settings. Namely, we vary: sampling resolution, the number of range scans, the distance the camera resides from the object, peak threshold, and variance threshold. To reproduce small imperfections commonly found in range data, we introduce a constant, modest amount of noise into the laser signal. We also slightly overlap the scans and register them, causing small misalignment errors. For each point cloud we randomly distribute camera positions uniformly on the bounding sphere of the object, rather than keeping their positions fixed.

See Figure 11 for the results of this experiment across all shapes, wherein the distributions take the form of box plots. The three error measures, mean distance, Hausdorff distance, and mean angle deviation, illuminate the various strengths and weaknesses of the algorithms.

Smooth Surfaces: The Gargoyle, Dancing Children, and Quasimoto shapes represent our class of shapes containing entirely smooth surface features. We find that the algorithms generally perform quite well on these shapes, however the different error metrics point to subtle differences in performance. For instance, Wavelet tends to produce nonsmooth, rather bumpy surfaces, yet the surface tends to stay close to the surface, which is likely due to the use of wavelet bases in the presence of nonuniform or missing data. This nonsmoothness is depicted in the mean distance and angle deviation plots, yet its Hausdorff distance performance is quite competitive, indicating it never strays too far from the surface.

It is well known that Poisson and Fourier tends to oversmooth the data, and in our experiments this is reflected in their rather large error in mean distance. However, in terms of Hausdorff distance and mean angle deviation they perform rather well, and are fairly consistent in their performance. This indicates that these algorithms are reliable in reconstructing surfaces which don't deviate too far from the original, while also remaining close in differential quantities. We note that Fourier is more consistent than Poisson, as Poisson suffers from a lack of resolution in regions of missing data.

While RBF performed well on the Dancing Children and Quasimoto models, on the Gargoyle model we see that it performed poorly across all metrics. The Gargoyle model is particularly difficult to sample as it has many concavities, and as shown by the lower

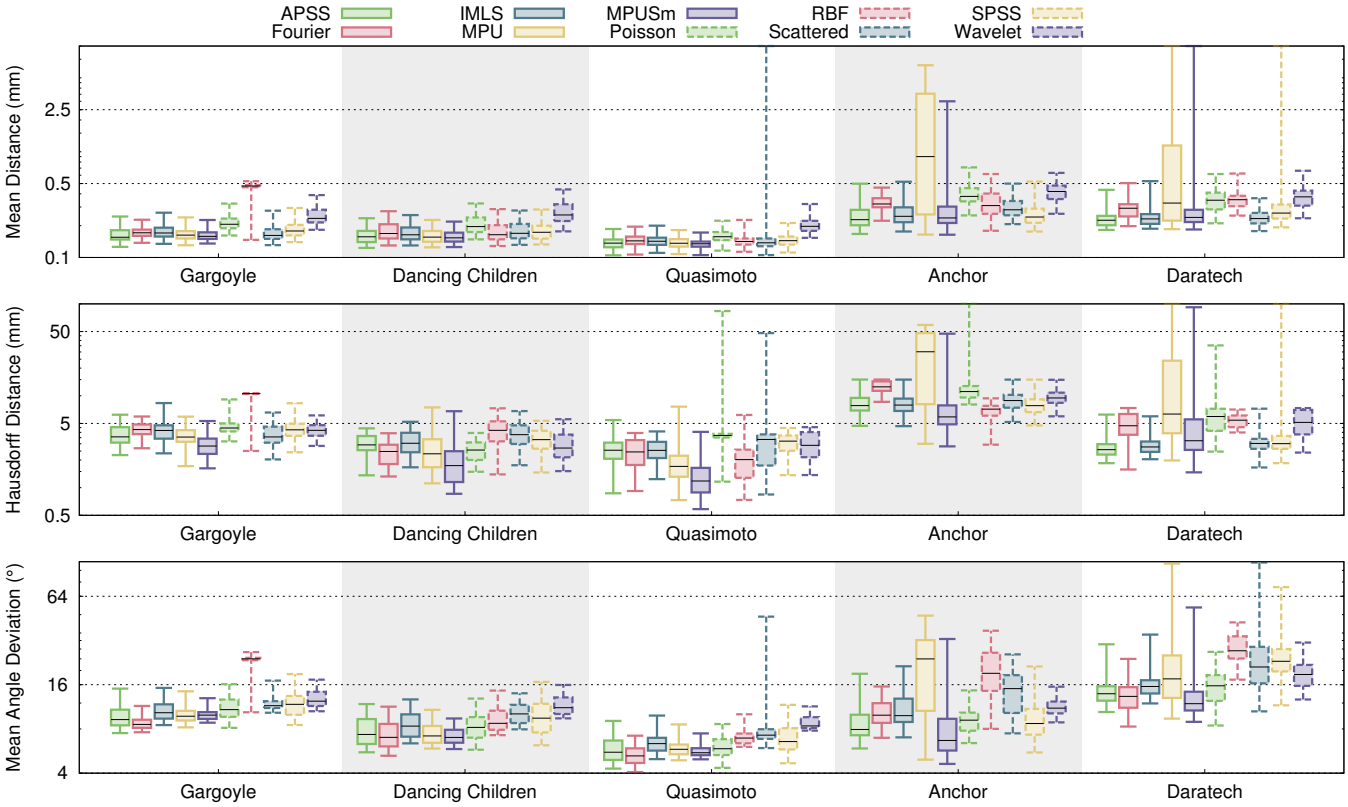


Fig. 11. Plots for all of the error distribution experiments. Each bar plot represents the distribution of a particular error measure for a given shape, sampled under a wide variety of scan parameters. The median provides a good indication of overall algorithmic performance for a given error measure, while the quartiles give an indication of algorithmic robustness.

quartile having large error across all metrics, RBF would tend to fill in the inside of the surface.

Table I. Summary of Error Distribution Experiments

algorithm	comps	bndry	manifold	genus	time
apss	47.37	140.86	0.50	1.82	36.02
fourier	1.54	0.00	1.00	0.49	28.70
imls	38.48	194.65	0.74	1.66	34.11
mpu	100.69	9.71	0.49	0.79	12.83
mpusmooth	2.88	2.93	0.91	0.67	17.83
poisson	1.54	0.44	1.00	0.63	36.83
rbf	51.73	6.30	0.82	13.55	34.78
scattered	1.90	214.21	1.00	7.47	4.48
spss	174.53	143.14	0.26	3.98	33.53
wavelet	1.35	0.04	1.00	0.71	2.13

Additional information for experiment 1, averaged across all point clouds and shapes. Here, comps refers to number of connected components, bndry is the length of boundary components, manifold is whether or not a mesh is manifold, 1 being it is and 0 otherwise, genus refers to the amount which deviates from the actual genus, and time is in seconds.

Sharp Features: The Anchor and Daratech shapes are particularly difficult to reconstruct. As these are shapes with sharp features, algorithms which only model smooth surfaces will have difficulty in reproducing sharp features. Additionally, these shapes have small topological features which are difficult to adequately scan due to occlusion. Hence we do not necessarily expect these algorithms to perform as well on these shapes as the others, and instead we use these shapes to measure robustness.

In observing MPU and MPUSm, we find instability in the presence of the Anchor and Daratech point clouds, where large spurious surface sheets are produced as a result of improperly fitting smooth shape functions to sharp features. However note that the PSS methods perform much better, despite also using smooth shape functions. PSS methods fit shape functions at every point, hence the error will be contained locally if there exists a poor fit, whereas MPU fits shape functions to the entire shape, resulting in a potentially unbounded error if a poor fit exists. Interestingly, RBF performs quite well in distance, yet has rather large error in normals. We found the RBF interpolant to remain quite close to the surface, at the expense of producing high-frequency details, hence the large normal deviations.

Topology: Overall, we find that the PSS methods and Scattered tend to perform quite well in the error metrics. However, these are also methods which produce holes in the presence of insufficient data. To depict the performance of these algorithms in terms of topology, we also show how these algorithms behave in their number of connected components, total length of boundary components, whether or not the reconstructed mesh is manifold, and the deviation from the true genus, averaged over all point clouds and shapes – see Table I. As shown, Fourier and Poisson tend to outperform these methods in all categories. With respect to the PSS

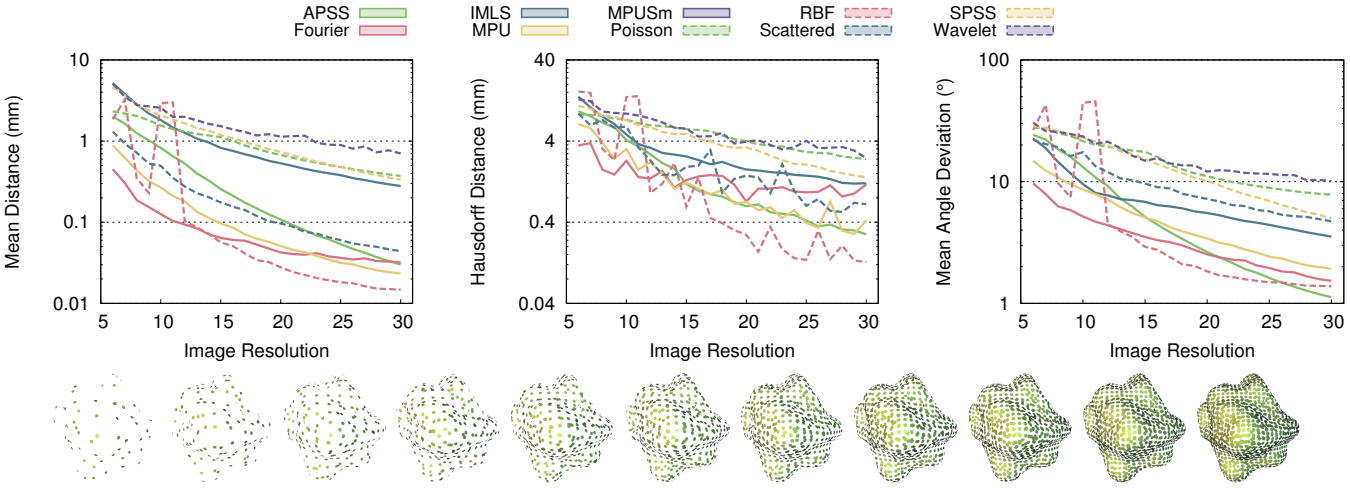


Fig. 12. Plots for the sparsity experiment, where we have sampled the bumpy sphere in increasing image resolution. The bottom row depicts a subset of these point clouds in decreasing sparsity. This experiment demonstrates how well these algorithms infer the surface from a sparse sampling.

methods, this demonstrates that they tend not to produce topologically clean implicit functions, likely due to their local nature. Additionally, we see that Scattered produces large holes, yet all of the shapes are watertight.

7.2 Sparse Sampling

A common data characteristic of point clouds is sparsity. Namely, for range scan data it is common for certain areas of the surface to be sampled less densely than others. Here we investigate how reconstruction algorithms behave as data sparsity varies in a controlled setting. We are interested in observing how these algorithms infer the surface *between* the given input points.

In this experiment we only vary the sampling resolution. We fix the number of scans and camera positions such that the shape is sufficiently covered, i.e. no missing data. We use the analytical normals of the surface, and no noise or misalignment. We use such clean input in order to restrict the problem to only data inference. We use the bumpy sphere as the test shape, as the coarse-scale features of the surface make data inference plausible.

See Figure 12 for plots of the experiment. MPUSm was unable to smooth its spherical covering on half of the point clouds due to the extreme sparsity, so we have omitted it from this experiment. From the distance measures we immediately see a partitioning of the algorithms: IMLS, Poisson, SPSS, and Wavelet all tend to behave rather poorly, while the other algorithms perform well. We should certainly expect this for Poisson and Wavelet, as the resolution of the output is proportional to the input size. However, it is interesting to observe the significant improvement of APSS over IMLS and SPSS, indicative that fitting spheres under sparse data is more advantageous than trying to fit planes to the data.

We also see that Fourier demonstrates remarkable robustness to sparse data. Usnder very sparse data Fourier performs best, whereas though APSS, MPU, RBF, and Scattered perform better as resolution increases, they perform rather poorly under very sparse data. However, observe that as the sampling resolution becomes somewhat dense, the distance error in APSS, MPU, and RBF steadily decreases while Fourier remains stagnant. This is a consequence of Fourier's inherent data smoothing, whereas those algorithms which fit shape functions to the data only improve their fits as resolution increases.

7.3 Missing Data

Missing data will almost always be present in scanned data, simply due to concavities in the shape which can not be reached by the scanner or insufficient scanning due to physical restraints of the scanner. In order to have a controlled setting to replicate missing data, we vary the peak threshold at which range may be rejected from consideration. We note that this is quite common for scanners, since the accuracy of the scanner suffers when the angle at which the laser line-of-sight and the normal becomes large, and the preferred option may be to reject range rather than accept outliers.

Similar to the previous experiment, here we fix the number of scans and camera positions, and use no additive noise, in order to isolate missing data as the primary challenge in the input. We then vary the peak threshold at which to reject samples from 0.8 to 0.4, where 1 is the expected peak. We have used the bumpy sphere and mailbox shapes, in order to observe the behavior of these algorithms in the presence of missing data on both smooth and sharp features.

See Figure 13 for plots of the experiment. We find that all of the indicator function methods perform quite well across both shapes, with the notable exception of Wavelet failing to converge to the limit surface as missing data decreases. We credit the robustness of indicator function methods to being global methods which do not attempt to fit shape functions.

Indeed, methods which fit shape functions have rather erratic behavior, particularly in the mailbox shape. MPU, MPUSm, and RBF are quite unstable, producing spurious surface sheets as missing data is introduced. When a neighborhood of a sharp feature, namely an edge, is sampled on one side and not on the other, shape functions of this kind are expected to be produced. As missing data increases, the samples used for shape fitting change, which results in spurious surface sheets only occasionally appearing. This variability in the points used for shape fitting is the cause of the inconsistencies found across MPU, MPUsm, RBF, and the PSS methods as missing data increases.

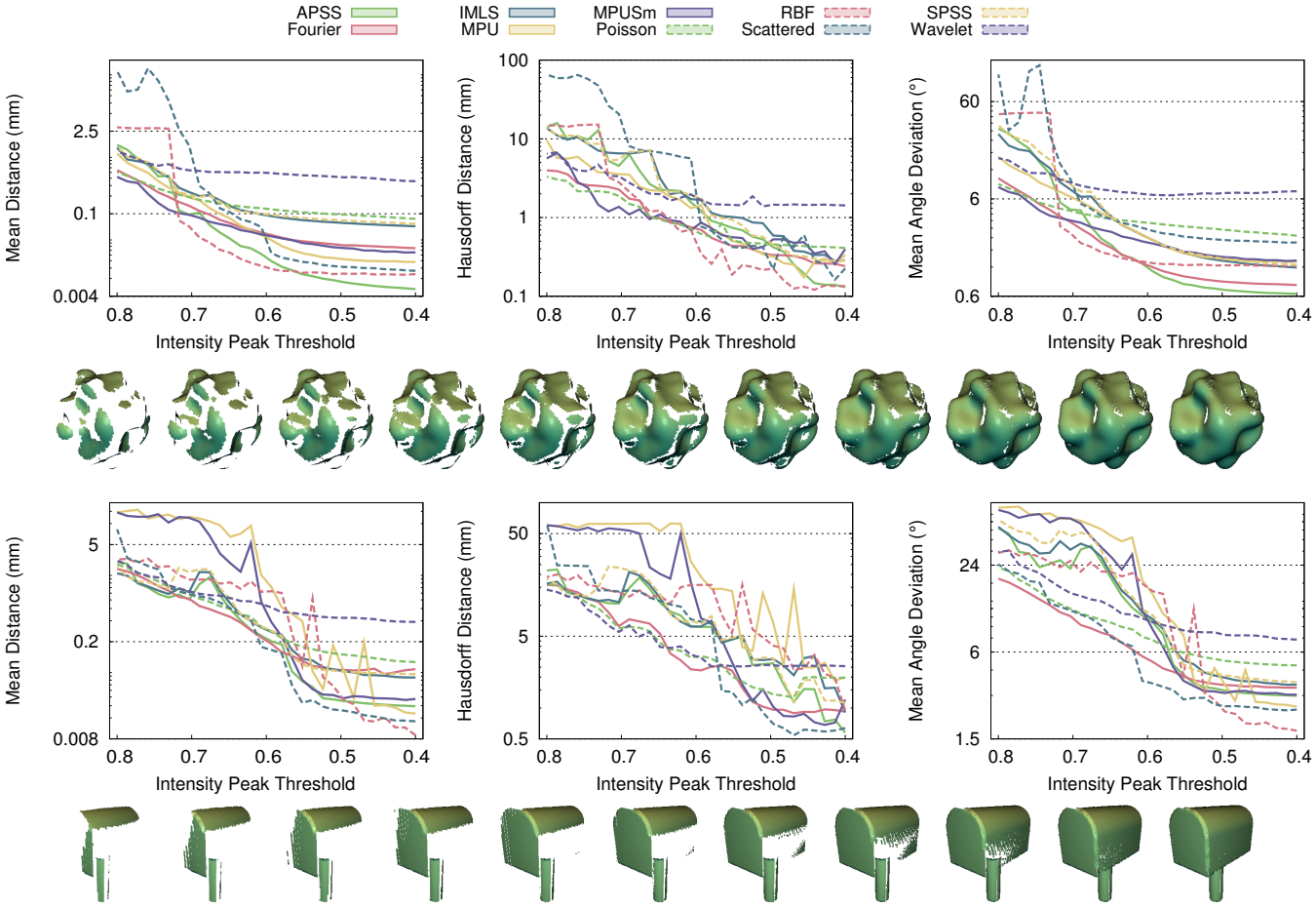


Fig. 13. Plots for the missing data experiments on the bumpy sphere (top row) and mailbox (bottom row). We generate missing data by varying the peak intensity threshold at which range is rejected. Note the differences in performance between the shape with smooth features and the shape with sharp features, as missing data is varied.

Scattered and the PSS methods tend to produce holes in the presence of large missing data, due to an insufficient number of samples in these areas. This missing data is the cause for their unstable behavior in the mailbox shape, as more missing data is introduced.

7.4 Noise

Finally we consider how robust reconstruction algorithms are to noise in the range data. We consider two scan parameters which have a signficiant impact on noise, noise magnitude and laser frustum field of view. The effect of noise magnitude is fairly clear, however we note that the thickness of the laser plays a significant impact on outliers. The thicker the laser, the more difficult peak detection becomes at depth discontinuities, resulting in outliers.

To this end, we have taken the spiral shape and sampled it under varying noise magnitudes, and varying laser thickness. We sufficiently sample it so that missing data or sparsity is not an issue, and compute normals directly from the points, allowing for improper orientation if direction propagation is incorrect. For each algorithm and each point cloud we also manually set the parameters to perform best, considering the scale of the noise. For the PSS and indicator function methods, this is quite intuitive. However for all other methods a maximum error tolerance effectively determines the amount of smoothing performed, which can be quite sensitive.

See Figure 14 for plots of the noise experiments. Note that Fourier and Poisson, in terms of all error metrics, are quite robust in the presence of noise. This is likely due to the global nature of these methods, where smoothing the data is a natural consequence. As observed by its large variance, RBF performs rather poorly in the presence of noise. Indeed, the necessity to produce dipoles for RBF becomes especially problematic in the presence of noise and outliers.

We observe that MPU and MPUsm are somewhat robust in the presence of noise given their small variance in Hausdorff distance, though interestingly we see significant differences between them in the two different distance measures. The smoothing performed via MPUSm tends to expand the surface outward, resulting in poor mean distance, yet it never strays too far from the surface, hence its good behavior in terms of Hausdorff distance.

The PSS methods all tend to smooth out noise and remain robust to outliers. However, we find their far-field behavior to be quite poor, see Table II. They tend to produce many extraneous connected components, as well as boundary components.

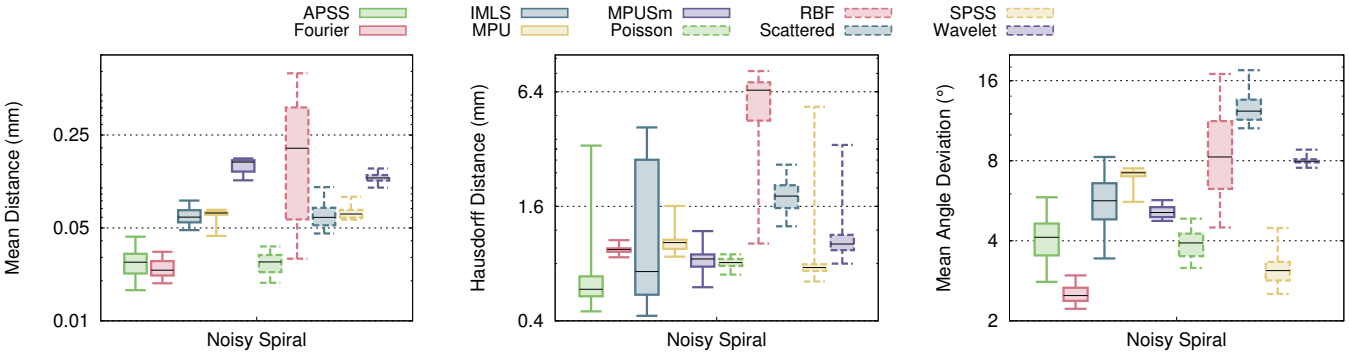


Fig. 14. Noise experiments for the spiral shape. Here we vary noise level and laser thickness, and aggregate this into distributions. A small variance in a distribution is a good indication of robustness to noise.

Table II. Noisy Spiral Statistics

algorithm	comps	bndry	manifold	genus	time
apss	221.60	0.71	1.00	0.00	50.59
fourier	1.00	0.00	1.00	0.00	27.24
imls	193.16	4.76	1.00	0.00	48.62
mpu	1.20	0.00	1.00	0.00	7.13
mpusmooth	1.08	0.06	1.00	0.00	23.08
poisson	1.00	0.00	1.00	0.00	30.90
rbf	12.48	4.69	0.92	0.30	18.90
scattered	1.08	0.00	1.00	0.44	3.11
spss	257.20	1.13	1.00	0.00	48.18
wavelet	1.00	0.00	1.00	0.00	2.26

Additional information for the noisy spiral experiments, averaged across all point clouds.

7.5 Discussion

Our small scale experiments tend to correlate well with the results of the error distribution experiments. For instance, the unstable behavior of RBF in the presence of sparse and missing data manifests itself in its unstable behavior across the gargoyle model, which is particularly difficult to adequately sample due its numerous concavities. Likewise, the behavior of MPU and to a lesser extent MPUSm in the presence of missing data on the mailbox correlates with their large variance in the Anchor and Daratech, indicative that they have trouble reconstructing sharp features. Conversely, we see that the stable behavior of Fourier in the small scale experiments correlates well with its relatively small variance in the distribution plots.

Our experiments point toward a number of deficiencies in the state of surface reconstruction. Our results demonstrate the remarkable robustness of methods based on computation of the indicator functions, yet these methods tend to oversmooth the data, reflected in their poor performance in mean distance across complex shapes. Developing an algorithm based on the indicator function which does not oversmooth the data would be very useful. Conversely, although MLS methods perform rather well in terms of mean and Hausdorff distance across the complex shapes, they demonstrate poor far-field behavior. We think that combining MLS methods with global constraints of some nature may rectify these issues.

Our benchmark should also prove to be useful for recent methods which resample point clouds with large missing data [Tagliasacchi et al. 2009; Cao et al. 2010; Shalom et al. 2010]. Although we have produced such point clouds in order to test robustness, it would be interesting to see how well these more recent resampling methods perform quantitatively.

All told, our benchmark consists of 351 point clouds across eight shapes, providing rich data for surface reconstruction developers. For our first set of experiments, we have 48 point clouds for each shape. Over 10 algorithms this amounts to a total of 2400 different reconstruction outputs, and over both distance and normal correspondences we have a total of 4800 correspondence mappings. We think that this construction of a distribution of point clouds for a given shape could be used in other areas, for instance potentially *learning* surface reconstruction, by using the point clouds and ground truth data as training data.

Limitations: While the surfaces in our benchmark cover a broad range of shapes, they are by no means exhaustive. As surface reconstruction becomes more specialized, such as the reconstruction of large-scale architectural buildings [Nan et al. 2010], we envision our benchmark to expand to these specific forms of surfaces. Our implicit shape representation should easily be able to accomodate other types of shapes.

Although we have generated a large variety of point cloud data with our sampling scheme, we are still fixing certain settings which may otherwise be worthwhile to further explore. For instance, we assume a diffuse BRDF in the scanning simulation, where it may be interesting to consider different forms of surface reflectance, and even spatially-varying BRDFs. Though laser-based optical triangulation scanners are quite popular, other forms of scanning may be worth simulating in order to replicate different acquisition artifacts, such as time-of-flight scanners.

8. CONCLUSIONS

We have presented a benchmark for the evaluation and comparison of surface reconstruction algorithms, restricted to the class of algorithms which take an oriented point cloud as input, and produce an approximating surface as output. Central to our benchmark is a controlled mechanism for simulating point cloud data acquired from laser-based scanners. We use a broad class of implicit surfaces as reference shapes to sample, which allows us to obtain accurate quantitative measurements for evaluating surface reconstruction error.

Our extensive experiments enables us to observe a wide range of behavior across existing algorithms. For instance, global methods such as those which reconstruct the indicator function are very robust in the presence of noise, while more local methods such as MPU and MLS methods produce highly accurate reconstructions in

the presence of clean data. The experiments point towards potential future work in surface reconstruction by illustrating the specific advantages and disadvantages in existing approaches.

By publicly releasing our data and code, researchers will now be able to benchmark their algorithms against existing algorithms and see where they stand. Additionally, our modeling and sampling methods will allow researchers to generate surfaces and point clouds tailored towards their interests. Hence we envision our benchmark to grow over time, continually incorporating data provided by the surface reconstruction community.

Acknowledgements

We thank Gaël Guennebaud, Michael Kazhadan, Josiah Manson, Yukie Nagai and Yutaka Ohtake for providing source code for their respective surface reconstruction algorithms. We also thank Benedict Brown for the registration code, Miriah Meyer for the particle system library, Jan Möbius for the OpenMesh library, and AIM@SHAPE for providing part of the meshes used in surface modeling. We thank Tiago Etiene and Lee Seversky for their useful feedback, and Bigyan Mukherjee who helped with some of our experiments. This work was partially funded by the National Science Foundation Foundation and grants from Fapesp-Brazil and CNPq-NSF.

APPENDIX

A. CLOSED-FORM SOLUTION OF POLYGONAL WEIGHT FUNCTIONS

In this section we detail the closed-form solution for Equation 2, used in the formation of our implicit functions. The basic idea is to cast the integral into the local coordinate system of the triangle, and perform integration in terms of polar coordinates, analogous to the construction of Green coordinates [Lipman and Levin 2010].

For a given evaluation point $\mathbf{x}$ and triangle t composed of the vertices $\mathbf{p}_1$, $\mathbf{p}_2$, and $\mathbf{p}_3$, and normal $\mathbf{n}$, we project $\mathbf{x}$ onto the plane of t:

$$\tilde{\mathbf{x}} = \mathbf{x} + \langle \mathbf{p}_1 - \mathbf{x}, \mathbf{n} \rangle \mathbf{n} \tag{21}$$

Now, for a given $\mathbf{p} \in t$, $|\mathbf{x}-\mathbf{p}|^2+\epsilon^2 = |\tilde{\mathbf{x}}-\mathbf{p}|^2+|\mathbf{x}-\tilde{\mathbf{x}}|^2+\epsilon^2 = |\tilde{\mathbf{x}}-\mathbf{p}|^2+\lambda_1$, where $\lambda_1 = |\mathbf{x}-\tilde{\mathbf{x}}|^2+\epsilon^2$ and is constant throughout the integration. We can now rewrite the integral as:

$$\int_{\mathbf{p}\in t} w(\mathbf{x},\mathbf{p}) \mathrm{d}\mathbf{p} = \sum_{t_i} sgn(t_i) \int_{\mathbf{p}\in t_i} \frac{\mathrm{d}\mathbf{p}}{(|\tilde{\mathbf{x}}-\mathbf{p}|^2+\lambda_1)^2} \tag{22}$$

Where t is broken up into t_1, t_2, t_3, formed from the triangles composed of $\tilde{\mathbf{x}}$ and $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3$, and sgn represents the orientation of the triangle: positive if oriented properly, and negative otherwise.

Without loss of generality we consider a single triangle t_1. We now convert this integral into polar coordinates:

$$\begin{aligned}\int_{\mathbf{p}\in t_1} \frac{\mathrm{d}\mathbf{p}}{(|\tilde{\mathbf{x}}-\mathbf{p}|^2+\lambda_1)^2} &= \int_{\theta=0}^{\theta=\beta} \int_{r=0}^{R(\theta)} \frac{r\,\mathrm{d}r\,\mathrm{d}\theta}{(r^2+\lambda_1)^2} \\ &= -\frac{1}{2}\int_0^{\beta} \frac{\mathrm{d}\theta}{R(\theta)^2+\lambda_1} + \frac{\beta}{2\lambda_1}\end{aligned}$$

Where β is the angle in t_1 opposite $\tilde{\mathbf{x}}$.

In order to have a clean parameterization of the length $R(\theta)$, we break up the integral into two parts by considering the orthogonal projection of the point $\tilde{\mathbf{x}}$ onto its opposing edge, $\hat{\mathbf{x}}$, and breaking t_1 into: $t_1^1 =< \tilde{\mathbf{x}}, \mathbf{p}_2, \hat{\mathbf{x}} >$ and $t_1^2 =< \tilde{\mathbf{x}}, \hat{\mathbf{x}}, \mathbf{p}_3 >$. Without loss of generality we consider t_1^1, and we obtain: $R(\theta) = \frac{|\tilde{\mathbf{x}}-\hat{\mathbf{x}}|}{\cos(\theta)}$. Hence the integral becomes:

$$\begin{aligned}\int_0^{\beta_1} \frac{\mathrm{d}\theta}{R^2(\theta)+\lambda_1} &= \int_0^{\beta_1} \frac{\cos^2(\theta)}{|\tilde{\mathbf{x}}-\hat{\mathbf{x}}|^2+\lambda_1\cos^2(\theta)} \\ &= \frac{\beta_1}{\lambda_1} - \frac{|\tilde{\mathbf{x}}-\hat{\mathbf{x}}|^2}{\lambda_1}\int_0^{\beta_1} \frac{\mathrm{d}\theta}{|\tilde{\mathbf{x}}-\hat{\mathbf{x}}|^2+\lambda_1\cos^2(\theta)}\end{aligned}$$

Applying the double angle formula to the above integral we obtain:

$$= \int_0^{\beta_1} \frac{\mathrm{d}\theta}{(|\tilde{\mathbf{x}}-\hat{\mathbf{x}}|^2+\frac{\lambda_1}{2}) + \frac{\lambda_1}{2}\cos(2\theta)}$$

Setting $a = 2$, $b = |\tilde{\mathbf{x}}-\hat{\mathbf{x}}|^2 + \frac{\lambda_1}{2}$, and $c = \frac{\lambda_1}{2}$, we may apply the relevant antiderivative to obtain:

$$\int \frac{\mathrm{d}\theta}{b+c\cos(a\theta)} = \frac{2}{a\sqrt{b^2-c^2}} \tan^{-1}\left\{\sqrt{\frac{b-c}{b+c}}\tan\frac{a\theta}{2}\right\} + C$$

B. DESCRIPTION OF SYNTHETIC SCANNER

Here we provide additional details on our synthetic scanner, as described in Section 5.1. To clarify the following discussion, we note that for each shape in our benchmark we have set its maximum dimension to be 70mm. Hence any scanning parameter based on distance is defined with respect to the bound of 70mm. Additionally, we place an upper bound on the radiance to be 1.

Our synthetic scanner is controlled by the following parameters:

—**Image resolution**. The image resolution, in conjunction with the number of scans used, effectively defines the resolution of the point cloud.

—**Baseline distance**. A small baseline distance magnifies depth errors in triangulation, while a large baseline results in greater occlusion. We have fixed our baseline to be with respect to the x-axis of the camera, though this may easily be adjusted to the y-axis by changing the laser sweep direction. We found that baseline distances ranging from 10mm to 150mm provide good variety in triangulation acccuracy and occlusions.

—**Stripe frustum field of view**. The thickness of the laser stripe has an impact on peak detection, in appropriately fitting a Gaussian. By default, we set the field of view such that the number of pixels visible within a distance of 50mm from the camera is roughly 10, which is a function of the image resolution.

—**Stripe resolution**. The number of laser stripes to project impacts the resolution of the depth. By default, we set this to be the x resolution of the camera, in order to obtain sufficient coverage. Setting the stripe resolution to be lower than the x resolution may result in some points not being affected by the laser stripes. By assigning a sufficiently large stripe frustum field of view, one may be able to obtain sufficient coverage.

—**Noise magnitude**. The magnitude of the noise corrupts the laser projection, making peak detection imprecise. Typical noise magnitudes we have used range from 0, or no noise, to 0.6, which can greatly corrupt the radiance signal.

—**Radiance smoothing bandwidth**. Smoothing the radiance image reduces noise, though at the potential cost of sacrificing the expected Gaussian laser profile. The bandwidth to use is largely dependent on the stripe frustum field of view and noise level. For instance, a thick laser under large noise magnitude will require a fairly large bandwidth to sufficiently smooth out the noise. We

note that smoothing, in conjunction with additive noise, may result in a radiance signal with smaller peak magnitudes, which can impact the peak magnitude threshold.

—**Peak magnitude threshold**. For large thresholds this will reject parts of the surface whose radiance signal is determined weak by a pixel's corresponding Gaussian fit. This is a major cause of missing data. For a laser containing little or no noise, typical thresholds range from 0.8, which will result in only highly confident range data, to 0.1, which will result in the rejection of few points. Under noise and radiance smoothing, the peak threshold must be adjusted to account for an expected reduction in peak magnitude.

—**Variance threshold**. Range at depth discontinuities are likely to be rejected under this threshold. We set the variance with respect to the width of the laser, where by default we only reject range whose variance in the Gaussian fit is larger than twice that of the laser width. Similar to the peak magnitude threshold, the variance threshold is sensitive to the noise magnitude and smoothing bandwidth.

We note that in our experiments, although we have generated quite a large number of point clouds, we have hardly explored the full parameter space of our scanner. By publicly releasing our synthetic scanner software, surface reconstruction researchers and practitioners will be able to replicate specific scanning conditions which they are interested in operating on.

REFERENCES

ABBASINEJAD, F., KIL, Y., SHARF, A., AND AMENTA, N. 2009. Rotating scans for systematic error removal. In *Computer Graphics Forum*. Vol. 28. 1319–1326.

ADAMSON, A. AND ALEXA, M. 2003. Approximating and intersecting surfaces from points. In *Symposium on Geometry processing*. Eurographics Association, 239.

ALEXA, M., BEHR, J., FLEISHMAN, D., LEVIN, D., AND SILVA, C. 2001. Point Set Surfaces. In *IEEE Visualization 2001*. IEEE, 21.

ALLIEZ, P., COHEN-STEINER, D., TONG, Y., AND DESBRUN, M. 2007. Voronoi-based variational reconstruction of unoriented point sets. In *Symposium on Geometry Processing*. Eurographics Association, 39–48.

AMENTA, N. AND BERN, M. 1999. Surface reconstruction by Voronoi filtering. *Discrete and Computational Geometry 22,* 4, 481–504.

AMENTA, N., CHOI, S., DEY, T. K., AND LEEKHA, N. 2002. A simple algorithm for homeomorphic surface reconstruction. *Int. J. Comput. Geometry Appl. 12,* 1-2, 125–141.

AMENTA, N., CHOI, S., AND KOLLURI, R. 2001. The power crust. In *Symposium on Solid modeling and applications*. ACM, 249–266.

BARIBEAU, R. AND RIOUX, M. 1991. Influence of speckle on laser range finders. *Applied Optics 30,* 20, 2873–2878.

BOISSONNAT, J.-D. AND CAZALS, F. 2002. Smooth surface reconstruction via natural neighbour interpolation of distance functions. *Comput. Geom. 22,* 1-3, 185–203.

BOUGUET, J. 2004. Camera calibration toolbox for matlab.

BROWN, B. AND RUSINKIEWICZ, S. 2007. Global non-rigid alignment of 3-D scans. *ACM Transactions on Graphics (TOG) 26,* 3, 21.

CAO, J., TAGLIASACCHI, A., OLSON, M., ZHANG, H., AND SU, Z. 2010. Point Cloud Skeletons via Laplacian Based Contraction. In *Shape Modeling International Conference (SMI), 2010*. IEEE, 187–197.

CAZALS, F. AND GIESEN, J. 2006. Delaunay triangulation based surface reconstruction. In *Effective Computational Geometry for Curves and Surfaces*, J.-D. Boissonnat and M. Teillaud, Eds. Springer-Verlag, Mathematics and Visualization, 231–276.

CIGNONI, P., MONTANI, C., AND SCOPIGNO, R. 1998. A comparison of mesh simplification algorithms. *Computers & Graphics 22,* 1, 37–54.

CURLESS, B. AND LEVOY, M. 1995. Better optical triangulation through spacetime analysis. In *Proceedings Fifth International Conference on Computer Vision*. IEEE, 987–994.

CURLESS, B. AND LEVOY, M. 1996. A volumetric method for building complex models from range images. In *Computer graphics and interactive techniques*. ACM, 312.

DEY, T. K. 2007. *Curve and Surface Reconstruction: Algorithms with Mathematical Analysis*. Cambridge University Press, New York.

DEY, T. K., LI, G., AND SUN, J. 2005. Normal estimation for point clouds: A comparison study for a Voronoi based method. In *Symposium on Point-Based Graphics*. Eurographics Association, 39–46.

FEDERER, H. 1959. Curvature measures. *Transactions of the American Mathematical Society*, 418–491.

FLEISHMAN, S., COHEN-OR, D., AND SILVA, C. T. 2005. Robust moving least-squares fitting with sharp features. *ACM Transactions on Graphics (TOG) 24,* 3, 544–552.

GARLAND, M. AND HECKBERT, P. S. 1997. Surface simplification using quadric error metrics. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 209–216.

GUENNEBAUD, G. AND GROSS, M. 2007. Algebraic Point Set Surfaces. *ACM Transaction on Graphics (Proceedings of SIGGRAPH) 26*, 23–9.

HILDEBRANDT, K., POLTHIER, K., AND WARDETZKY, M. 2006. On the convergence of metric and geometric properties of polyhedral surfaces. *Geometriae Dedicata 123,* 1, 89–112.

HOPPE, H., DEROSE, T., DUCHAMP, T., MCDONALD, J., AND STUETZLE, W. 1992. Surface reconstruction from unorganized points. In *Computer graphics and interactive techniques*. ACM, 71–78.

KAZHDAN, M. 2005. Reconstruction of solid models from oriented point sets. In *Symposium on Geometry processing*. Eurographics Association, 73.

KAZHDAN, M., BOLITHO, M., AND HOPPE, H. 2006. Poisson surface reconstruction. In *Symposium on Geometry processing*. Eurographics Association, 70.

KOLLURI, R. 2005. Provably Good Moving Least Squares. In *Symposium on Discrete Algorithms*. SIAM, 1008–1018.

LI, H., ADAMS, B., GUIBAS, L. J., AND PAULY, M. 2009. Robust single-view geometry and motion reconstruction. *ACM Transactions on Graphics (Proceedings SIGGRAPH Asia 2009) 28,* 5 (December).

LIPMAN, Y. AND LEVIN, D. 2010. Derivation and Analysis of Green Coordinates. *Computational Methods and Function Theory 10,* 1, 167–188.

MANSON, J., PETROVA, G., AND SCHAEFER, S. 2008. Streaming surface reconstruction using wavelets. *27,* 5, 1411–1420.

MEYER, M., DESBRUN, M., SCHR
"ODER, P., AND BARR, A. 2002. Discrete differential-geometry operators for triangulated 2-manifolds. *Visualization and mathematics 3,* 7, 34–57.

MEYER, M., KIRBY, R., AND WHITAKER, R. 2007. Topology, accuracy, and quality of isosurface meshes using dynamic particles. *IEEE Transactions on Visualization and Computer Graphics 13,* 6, 1704–1711.

MITRA, N. AND NGUYEN, A. 2003. Estimating surface normals in noisy point cloud data. In *Symposium on Computational geometry*. ACM, 328.

NAGAI, Y., OHTAKE, Y., AND SUZUKI, H. 2009. Smoothing of Partition of Unity Implicit Surfaces for Noise Robust Surface Reconstruction. *28,* 5, 1339–1348.

NAN, L., SHARF, A., ZHANG, H., COHEN-OR, D., AND CHEN, B. 2010. SmartBoxes for interactive urban reconstruction. In *ACM SIGGRAPH 2010 papers*. ACM, 1–10.

OHTAKE, Y., BELYAEV, A., ALEXA, M., TURK, G., AND SEIDEL, H. 2003. Multi-level partition of unity implicits. *ACM Transactions on Graphics (TOG) 22,* 3, 463–470.

OHTAKE, Y., BELYAEV, A., AND SEIDEL, H. 2005a. An integrating approach to meshing scattered point data. In *Symposium on Solid and physical modeling*. ACM, 69.

OHTAKE, Y., BELYAEV, A. G., AND SEIDEL, H.-P. 2005b. 3d scattered data interpolation and approximation with multilevel compactly supported rbfs. *Graphical Models 67,* 3, 150–165.

SCHARSTEIN, D. AND SZELISKI, R. 2002. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision 47,* 1, 7–42.

SEITZ, S., CURLESS, B., DIEBEL, J., SCHARSTEIN, D., AND SZELISKI, R. 2006. A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms. In *Computer Vision and Pattern Recognition, Volume 1*. IEEE Computer Society, 528.

SHALOM, S., SHAMIR, A., ZHANG, H., AND COHEN-OR, D. 2010. Cone Carving for Surface Reconstruction. *ACM Transactions on Graphics, (Proceedings SIGGRAPH Asia 2010) 29,* 5.

SHARF, A., ALCANTARA, D., LEWINER, T., GREIF, C., SHEFFER, A., AMENTA, N., AND COHEN-OR, D. 2008. Space-time surface reconstruction using incompressible flow. *ACM Transactions on Graphics (TOG) 27,* 5, 1–10.

SHEN, C., O'BRIEN, J., AND SHEWCHUK, J. 2004. Interpolating and approximating implicit surfaces from polygon soup. In *ACM SIGGRAPH 2004 Papers*. ACM, 896–904.

SUSSMUTH, J., MEYER, Q., AND GREINER, G. 2010. Surface reconstruction based on hierarchical floating radial basis functions. In *Computer Graphics Forum*. John Wiley & Sons.

TAGLIASACCHI, A., ZHANG, H., AND COHEN-OR, D. 2009. Curve skeleton extraction from incomplete point cloud. *ACM Transactions on Graphics, (Proceedings SIGGRAPH 2009) 28,* 3, Article 71, 9 pages.

TER HAAR, F., CIGNONI, P., MIN, P., AND VELTKAMP, R. 2005. A Comparison of Systems and Tools for 3D Scanning. In *3D Digital Imaging and Modeling: Applications of Heritage, Industry, Medicine and Land, Workshop Italy-Canada*. Session P.12.

VLASIC, D., PEERS, P., BARAN, I., DEBEVEC, P., POPOVIĆ, J., RUSINKIEWICZ, S., AND MATUSIK, W. 2009. Dynamic Shape Capture using Multi-View Photometric Stereo. *ACM Transactions on Graphics 28,* 5, 174.