

# A Survey and Benchmark of Automatic Surface Reconstruction from Point Clouds

Raphael Sulzer, Loic Landrieu, Renaud Marlet, and Bruno Vallet

**Abstract**—We survey and benchmark traditional and novel learning-based algorithms that address the problem of surface reconstruction from point clouds. Surface reconstruction from point clouds is particularly challenging when applied to real-world acquisitions, due to noise, outliers, non-uniform sampling and missing data. Traditionally, different handcrafted priors of the input points or output surface have been proposed to make the problem more tractable. However, hyperparameter tuning for adjusting priors to different acquisition defects can be a tedious task. To this end, the deep learning community has recently addressed the surface reconstruction problem. In contrast to traditional approaches, deep surface reconstruction methods can learn priors directly from a training set of point clouds and corresponding true surfaces. In our survey, we detail how different handcrafted and learned priors affect the robustness of methods to defect-laden input and their capability to generate geometric and topologically accurate reconstructions. In our benchmark, we evaluate the reconstructions of several traditional and learning-based methods on the same grounds. We show that learning-based methods can generalize to unseen shape categories, but their training and test sets must share the same point cloud characteristics. We also provide the code and data to compete in our benchmark and to further stimulate the development of learning-based surface reconstruction: <https://github.com/raphaelsulzer/dsr-benchmark>.

**Index Terms**—surface reconstruction, point clouds, deep learning, mesh generation, survey, benchmark

## 1 INTRODUCTION

MODERN three-dimensional (3D) acquisition technology, such as range scanning or multi-view stereo (MVS) brought the ability to record the world in the form of 3D point clouds. However, point clouds are usually not sufficient to model complex physical processes such as fluid dynamics. Instead, a variety of applications in science and engineering require a representation of objects or scenes in the form of a continuous surface. Therefore, surface reconstruction from point clouds is a key step between acquisition and analysis of surface models and is a long-standing problem in digital geometry processing. In this paper, we survey and benchmark several traditional and learning-based methods that address the problem of surface reconstruction from point clouds.

If no prior information about the sought surface is known, surface reconstruction from point clouds is an ill-posed problem, as there are an infinite number of surfaces with different geometry and topology that can pass through, or near, the point samples. Furthermore, acquisition defects in the point cloud, such as non-uniform sampling, noise, outliers or missing data complicate the reconstruction of a geometrically and topologically accurate surface [1]. See Figure 1 for an illustration. Traditionally, surface reconstruction methods made the problem more tractable by using handcrafted priors, imposed on the input, such as point

density, level of noise or outliers, and on the output, such as smoothness, topological properties or the shape category. In contrast, recent methods introduced by the deep learning community can learn point cloud defects or shape patterns directly from training data and therefore promise to reconstruct more accurate surfaces without the need for manual parameter tuning. However, so far deep surface reconstruction (DSR) methods have mostly been applied on datasets with a small number of different object categories. Such datasets are not representative for real-world applications, where algorithms have to reconstruct surfaces containing a large variety of shapes unseen during training.

Furthermore, DSR methods are often applied on uniformly sampled point clouds. Likewise, such point clouds are not representative for real-world acquisitions, as they do not model non-uniformity or missing data stemming *e.g.* from occlusions, or transparent and low texture areas. The ability to reconstruct shapes, either from unseen shape classes or from point clouds with unseen defects is rarely studied in a systematic manner for DSR methods.

To this end, we propose several experiments to benchmark algorithms for surface reconstruction from point clouds. We make use of a variety of publicly available shape datasets with object surfaces of different complexities. The objects are represented by a true surface  $\mathcal{S}$ , which is a boundary-free 2-manifold, *i.e.* each point on the surface has a neighborhood that is homeomorphic to an open subset of the Euclidean plane. We synthetically scan the objects to produce point clouds with real characteristics. Having access to the true surfaces allows us to measure the geometric and topological reconstruction quality of the benchmarked methods. We also verify our findings on real-world point clouds.

We compare novel learning-based algorithms to tradi-

- R. Sulzer was with LASTIG, Univ Gustave Eiffel, IGN-ENSG, F-94160 Saint-Mandé, France during this work and is now with Centre Inria d’Université Côte d’Azur, Sophia Antipolis, France. E-mail: raphael.sulzer@gmx.de.
- R. Marlet is with LIGM, Ecole des Ponts, Univ Gustave Eiffel, CNRS, Marne-la-Vallée, France and with valeo.ai, Paris, France.
- L. Landrieu and B. Vallet are with LASTIG, Univ Gustave Eiffel, IGN-ENSG, F-94160 Saint-Mandé, France. E-mail: firstname.lastname@ign.fr.

Corresponding author: R. Sulzer

tional test-of-time methods to specifically study the influence of learned priors incorporated into the surface reconstruction process. We thereby pay special attention to the generalization capability of methods to unseen domains. Our main contributions are as follows:

- We review methods for surface reconstruction from point clouds from over three decades up to recent learning-based methods. We contrast popular test-of-time with novel DSR methods.
- We benchmark traditional and learning-based methods on the same ground across several experiments, using openly available shape datasets and point clouds generated with synthetic scanning.

## 2 RELATED WORK

### 2.1 Surveys

There exists only few works that survey the broad field of surface reconstruction from point clouds [1], [2], [3], [4], most of them predating the advance of learning-based surface reconstruction [1], [2], [4]. Surface reconstruction methods are often grouped into interpolating or approximating methods [5]. Interpolating methods “connect” all points of the input point cloud, or a subset thereof, usually by linearly interpolating between pairs of points. Approximating methods often define one or several smooth functions approximating the point cloud globally or locally. See Figure 2 for an illustration. Berger *et al.* [1] and Cazals & Giesen [2] provide detailed reviews for approximating and interpolating surface reconstruction methods, respectively.

To the best of our knowledge, only one survey includes learning-based methods [3]. However, this survey predates important developments for learning-based methods, such as the incorporation of local information [6], [7], [8], [9], [10], [11], [12]. In this work, we review both interpolating and approximating methods and focus on novel ideas in learning-based surface reconstruction. While many reconstruction methods can be distinguished by the prior assumptions they impose [1], we argue that a variety of successful methods combine different priors. This makes grouping by priors difficult. We thus organize methods into two groups: surface-based and volume-based approaches. This breakdown closely relates to the two main classes of mathematical representations of a surface: parametric and implicit.

### 2.2 Benchmarks

To date, benchmarks for surface reconstruction from point clouds are rare. Many methods use custom datasets to evaluate their approach, usually generated by uniformly sampling point clouds from ground truth shapes of existing shape collections [6], [7], [8], [11], [12], [13]. However, the characteristics of the sampled point clouds often differ across publications, which hampers the ability to fairly compare the results of different works. Furthermore, the point clouds often lack common defects of real acquisitions, such as missing data or outliers. One notable exception is the benchmark of Berger *et al.* [14]. The authors develop a synthetic range scanning procedure to produce scans with realistic artifacts, such as noise, non-uniformity and

misaligned scans and create point clouds from shapes with non-trivial topology and details of various feature sizes. While providing interesting results, the benchmark predates learning-based surface reconstruction and only considers traditional approximating methods. In the benchmarks proposed in this paper, we reuse their synthetic range scanning procedure and their five test shapes, as they provide realistic and challenging input for both learning-based and traditional algorithms. We also implement our own synthetic scanning procedure for MVS-like point clouds. We use the synthetic scanning to scan existing large shape datasets to create training datasets with true surfaces and point clouds with realistic characteristics.

A problem related to surface reconstruction is the generation of point clouds from 2D information such as overlapping images. There exists a variety of benchmarks using data captured in a laboratory environment [15], [16] or in the wild [17], [18], [19]. These benchmarks often use a low quality image acquisitions as reconstruction input. Simultaneously, a higher quality acquisition, *e.g.* from LiDAR scans, serves as reference.

One problem with this approach is that, even for high quality acquisition techniques, it is difficult to produce complete ground truth point clouds. This issue is sometimes addressed by decreasing the ground truth domain to specific evaluation areas, in which reliable information is available either from recorded points or sightlines between points and sensors [16], [18]. However, in contrast to true surfaces, reference point clouds, do not allow to calculate topologic metrics such as the number of components or differential metrics such as surface normals. Furthermore, most learning-based methods require closed reference surfaces instead of reference point clouds for training.

## 3 SURFACE DEFINITION, REPRESENTATIONS, PROPERTIES AND RECONSTRUCTION

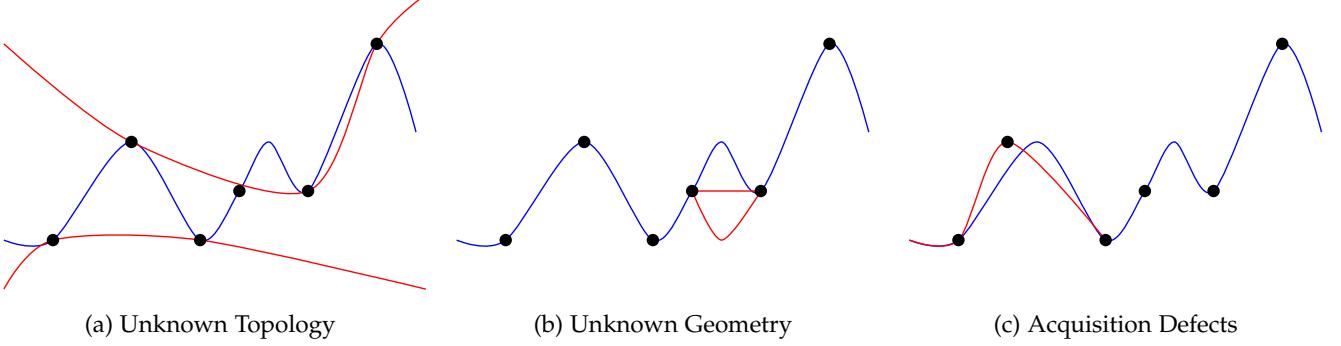
In this section, we first provide a definition of a surface and its mathematical and digital representations. We then discuss important surface properties. Finally, we establish the connection between mathematical surface representations, and the grouping of surface reconstruction algorithms used in our survey.

### 3.1 Definition

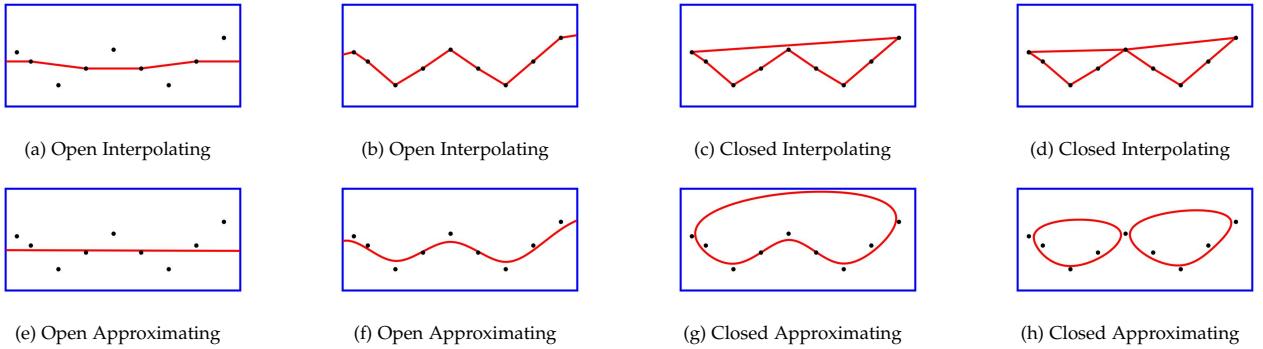
A surface can be defined as an orientable, continuous 2-manifold in  $\mathbb{R}^3$ , with or without boundaries [5], [20], [21]. These properties are important for surface visualisation and processing, and we will discuss them further down. Mathematically, there are two main classes of surface representations: *parametric* and *implicit*.

### 3.2 Representations

*Parametric surfaces* are defined by a function  $f : \Omega \mapsto S$  that maps a parameter domain  $\Omega \in \mathbb{R}^2$  to the surface  $S = f(\Omega) \in \mathbb{R}^3$ . However, for complex surfaces it is not feasible to find a single function that can parameterise  $S$ . Therefore, the parameter domain  $\Omega$  is usually split into sub-regions for which an individual function is defined [5]. The most common way is to segment  $\Omega$  into triangles, which



**Figure 1: Difficulties in surface reconstruction from point clouds:** In each plot, we show the real surface —, point samples ●, and possible reconstructions —. The correct topology and geometry of the real surface are not known from the point samples (a,b). The point samples may also include acquisition defects such as noise (c). The goal of any surface reconstruction algorithm is finding a good approximation of the real surface, in terms of its geometry and topology. Learning-based surface reconstruction can learn shape patterns or sampling errors such as the one exemplified here, and use the learned knowledge during reconstruction for a better approximation.



**Figure 2: Approximating and interpolating surfaces from point clouds:** A surface generated from point samples can either interpolate (top row) or approximate (bottom row) the samples. Theoretically, there exist an infinite number of surfaces with different geometry and topology that can pass through, or near, the samples. We show eight different surfaces — reconstructed from the same point cloud ●●●● in (a) - (h). The point cloud can be seen as a sampling of a part of a real surface. All reconstructed surfaces are watertight, as they are either closed and boundary-free, or their only boundary is the intersection with the domain boundary □. The surface in (d) is non-manifold in the center-vertex. All other surfaces are manifold. Except for (h), all surfaces are comprised of only one component. In contrast to the point cloud depicted here, in our benchmark, we mainly consider point clouds sampled from closed surfaces.

are planar by definition. A set of triangles approximating  $\mathcal{S}$  can be efficiently stored and processed as a triangle surface mesh  $\mathcal{M} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$ , with triangle facets  $\mathcal{F}$ , edges  $\mathcal{E}$  and vertices  $\mathcal{V}$ .

*Implicit surfaces* are defined by the level-set  $c$  of a scalar valued function  $F : \mathbb{R}^3 \mapsto \mathbb{R}$ :

$$\mathcal{S}_c = \{\mathbf{x} \in \mathbb{R}^3 \mid F(\mathbf{x}) = c\}. \quad (1)$$

The most common choice of the implicit function  $F$  is either a signed distance or an occupancy function. A signed distance function (SDF) gives the distance from a 3D point  $\mathbf{x}$  in space to the surface; with points in the interior signed a negative value, and points on the exterior signed a positive value. An indicator or occupancy function (OF) usually has a value of 1 inside the surface and 0 outside. The  $c$ -level-set of  $F$  then yields the surface  $\mathcal{S}$ , where  $c = 0$  in the case of a signed distance function and  $c = 0.5$  in the case of an occupancy function. Similar to the parametric case, the implicit function domain is often split into sub-regions, such

as voxels, octree-nodes or tetrahedra, and constant functions are defined in each sub-region.

### 3.3 Properties

The reconstructed surface  $\mathcal{S}^r$  should be close in terms of geometry and topology to the real surface  $\mathcal{S}$  from which the point cloud  $\mathcal{P}$  is sampled. To facilitate subsequent geometric operations on  $\mathcal{S}^r$ , such as sampling or deforming the surface, a mesh reconstruction  $\mathcal{M}$  is also desirable.  $\mathcal{S}^r$  and  $\mathcal{M}$ , respectively, should have the following properties (see Figure 2 for illustrations):

- **Watertight:** A geometric surface is closed if it is boundary-free. A mesh  $\mathcal{M}$  is closed—or boundary-free—if no edge is incident to exactly one facet. However, a reconstructed surface of a real scene necessarily has a border defined e.g. by the limit of the scan coverage. One may still reconstruct a closed surface by intersecting it with the boundary of the domain in which

$f$  or  $F$  is defined: *e.g.* the convex hull or bounding box of  $\mathcal{P}$ . However, this procedure may not be desirable, as it can hinder simple geometric analysis such as the calculation of surface area. Instead, we define a surface as watertight if it is boundary free, *except* for a possible intersection with the domain boundary.

- **Manifold:** We consider real and geometric surfaces to be 2-manifolds, *i.e.* each point on the surface has a neighborhood that is homeomorphic to an open subset of the Euclidean plane. A mesh  $\mathcal{M}$  is manifold if it is edge- and vertex-manifold, and intersection-free.
  - *Edge-manifold:* For each edge  $\mathcal{E}$ , the set of facets  $\mathcal{F}$  sharing this edge form a topological (half-)disk. This means that no edge can be incident to more than two facets.
  - *Vertex-manifold:* For each vertex  $\mathcal{V}$ , the set of facets sharing this vertex form a topological (half-)disk. This means that facets with a common vertex form an open or closed fan, *i.e.* there are no dangling facets.
- **Intersection-free:**  $\mathcal{M}$  is intersection free if all pairs of facets not sharing an edge or vertex do not intersect.
- **Orientable:**  $\mathcal{M}$  is orientable if one can define a consistent continuous orientation of each facet. This means that the order of the vertices of all facets is either clockwise or counter-clockwise and a common edge of two adjacent facets has opposite orders on the two sides.

The watertight property is useful for simulations such as fluid dynamics. Manifoldness and orientability are often required for mesh storing and processing, in particular because they are a prerequisite for the widely-used half-edge data structure [22], [23]. Furthermore, intersection-free and orientable surfaces lead to a well-defined notion of inside and outside, which is important for mesh visualization and a variety of geometric operations.

### 3.4 Reconstruction

Surface reconstruction from point clouds is the process of constructing a continuous surface of which discrete point samples have been acquired. In our survey, we group methods for surface reconstruction from point clouds into two groups: surface- and volume-based. *Surface-based* reconstruction methods consists in finding (a set of) parameterised surfaces  $\mathcal{S}^r$  that approximate the point cloud  $\mathcal{P}$ , either in the form of triangles or larger two-dimensional (2D) patches, or by deforming parameterised enclosing envelops such as meshed spheres. The main challenge for surface-based methods using a single function  $f$  is that the topology of  $\Omega$  has to be equivalent to the topology of  $S$ , which is usually unknown. The main challenge for surface-based methods with individual functions for sub-regions of  $S$ , on the other hand, is to guarantee a consistent transition between each region. Hence, these methods often struggle to produce an intersection-free, manifold and watertight surface.

*Volume-based* methods, on the other hand, segment a subset of  $\mathbb{R}^3$  into interior (inside) and exterior (outside) subspaces. The surface is implicitly defined as the interface between the two subspaces. Most, but not all algorithms in

this class formulate the problem as finding an implicit function. Surfaces from volume-based methods are guaranteed to be watertight and intersection-free, but not necessarily manifold [2].

While surface-based methods can directly yield a mesh, *e.g.* by triangulating  $\Omega$ , volume-based methods usually require an additional processing step. If the implicit field is discretized with tetrahedra, one can simply use a process which is sometimes called triangle-from-tetrahedra (TFT). TFT builds a triangle mesh from all triangles that are adjacent to one inside- and one outside-tetrahedra. Another option is the algorithm of Boissonnat and Oudot [24] that iteratively samples  $F$  along lines from inside to outside to find points that lie on  $S$  and builds a triangle mesh from these points. One of the most popular methods for mesh extraction from an implicit field is Marching Cubes [25], which (i) discretizes the implicit function into voxels, (ii) constructs triangles inside each voxel that have at least one inside and one outside vertex and (iii) extracts a triangulation as the union of all triangles. Recently, mesh extraction has also been addressed by the deep learning community. Neural meshing [26] specifically addresses the case where an implicit function is represented by a neural network, and aims to extract meshes with fewer triangles compared to Marching Cubes from such a function.

In both, surface- and volume-based groups, there are methods that come with theoretical guarantees about the topology and geometry of the reconstruction in the absence of noise and when the point sampling is dense enough [2]. However, in this paper, we are mostly interested in the robustness of methods to defect-laden input point clouds from 3D scanning.

## 4 SURVEY

In this section, we review important surface- and volume-based surface reconstruction methods and discuss their robustness against different point cloud defects. We also show that learning-based approaches are often related to more traditional methods.

### 4.1 Surface-based reconstruction

#### 4.1.1 Interpolating approaches

*Advancing-front* techniques.: Most traditional surface-based approaches linearly interpolate between the point samples  $\mathcal{P}$ , or a subset thereof. This can be done efficiently by triangulating triplets of points which respect the empty ball property *i.e.* no other point lies within their circumsphere. Triangulating all triplets of  $\mathcal{P}$  that have this property leads to the 3D Delaunay tetrahedralisation (3DT) of  $\mathcal{P}$ . The Ball Pivoting algorithm [27] is a greedy approach to find local triplets of points that form a triangle which is part of the surface. The first step is to (i) define a ball with constant radius, related to the density of  $\mathcal{P}$  and to (ii) select a seed triplet of points. The ball must touch all three points and have no other point in its interior. The points then form the first surface triangle. Then, (iii) the ball pivots around an edge of the triangle until it touches a new point, forming a new surface triangle. Once all possible edges have been processed the algorithm starts

**Table 1: Overview of surface- and volume-based surface reconstruction methods:** We show an overview of surface- and volume-based surface reconstruction methods, both non-learning and learning-based, together with their input requirements (*normals*, *sensor pose*) and output type (*triangle mesh* or *implicit field*). Attributes denoted in brackets are optional. Methods with a *local* receptive field divide the point cloud into smaller sub-regions and define individual functions or surface patches for each sub-region. Methods with a *global* receptive field consider the entire point cloud at once. Methods denoted with *both* combine local and global receptive fields. We test methods in **bold** in our benchmark.

Method		learning	normals	sensor pose	receptive field	output
<b>Surface-based</b>						
<b>BPA</b> [27]						
Sharf <i>et al.</i> [28]					local	triangle mesh
AtlasNet [29]	✓				both	triangle mesh
IER [30]	✓				local	triangle mesh
PointTriNet [31]	✓				both	triangle mesh
DSE [11]	✓				local	triangle mesh
P2M [32]					both	triangle mesh
<b>Volume-based</b>						
SPSR [33]		✓			both	implicit field
<b>Labatut <i>et al.</i></b> [34]			✓		global	triangle mesh
ONet [35]	✓				global	implicit field
DeepSDF [13]	✓				global	implicit field
IM-Net [36]	✓				global	implicit field
<b>ConvONet</b> [6]	✓				both	implicit field
IGR [37]	(✓)	(✓)			global	implicit field
LIG [8]	✓	✓			local	implicit field
DGNN [10]	✓			✓	both	triangle mesh
SAP [38]	✓				both	implicit field
P2S [9]	✓				both	implicit field
SAP [38]	(✓)				both	implicit field
POCO [12]	✓	(✓)			local	implicit field

with a (iv) new seed triangle until all points of  $\mathcal{P}$  have been considered. The algorithm has later been refined to be more robust to non-uniform sampling [39], [40]. The Ball Pivoting algorithm and its related variations are often called advancing-front techniques. Their main drawback is that they are not robust to point cloud defects such as noise or point clouds with large missing parts.

**Selection-based:** Similar to advancing-front techniques, the idea to iteratively build the triangulation from initial candidate triangles has also been explored in learning-based methods [30], [31]. PointTriNet [31] (i) starts with an initial set of seed triangles from a  $k$ -nearest neighbor graph of  $\mathcal{P}$ . Then, (ii) a first network takes in neighboring points and triangles of each seed triangle, and estimates its probability to be part of the surface. (iii) Triangles with high probability are selected to be part of the final surface and (iv) a second network proposes new candidate triangles constructed from two points of already selected surface triangles and neighboring points. The proposed new candidates are, again, processed by the first network and the algorithm continues for  $n$  user-defined iterations. The loss function is based on Chamfer distance between input points and the reconstructed surface, which allows the method to

be trained without the need for ground truth meshes. IER-meshing [30] also (i) starts with a large set of seed triangles from a  $k$ -nearest neighbor graph. It then defines a so-called intrinsic-extrinsic ratio (IER), as the quotient of geodesic and Euclidean distance between points of a triangle. (ii) This ratio is estimated by an multilayer perceptron (MLP) from learned point features per triangle and supervised with IER's from a ground truth mesh. (iii) Only triangles with an IER close to 1 (*i.e.* Euclidean distance  $\approx$  geodesic distance) are considered to be part of the surface and (iv) selected based on handcrafted heuristics. Both aforementioned methods have shown to be robust against small amounts of noise in the input point cloud. However, their reconstructed surfaces are neither manifold nor watertight.

**Tangent plane and other projection methods:** Another class of surface-based interpolating approaches are tangent plane methods. This class includes the algorithm of Boissonnat [41], which is according to Cazals and Giesen [2] probably the first algorithm to address the surface reconstruction problem. The basic idea is to (i) find a tangent plane for each sample point, (ii) project the points local neighborhood on the tangent plane, (iii) construct 2D Delaunay triangulations of the projected points and (iv) merge the local reconstructions. A shortcoming of such an approach is that tangent planes are difficult to use in areas with high curvature or thin structures [11]. To this end, the idea of using local 2D Delaunay triangulations of projected points has been refined in a recent learning-based approach [11]. Instead of tangent planes, DSE-meshing [11] uses *logarithmic maps*, local surface parametrizations around a point  $p$ , based on geodesics emanating from  $p$ . This method (i) classifies geodesic neighbors of each point in  $\mathcal{P}$  from a set of  $k$ -nearest neighbors. Then, (ii) an MLP approximates a logarithmic map parametrization to gain a 2D embedding of the geodesic neighbors. Lastly, (iii) neighboring logarithmic maps are mutually aligned and triangulated. This step allows the method to reconstruct surfaces with fewer non-manifold edges, compared to methods that process triangles independently. However, the surface is still not watertight and the method has not been tested for reconstruction from noisy point clouds.

#### 4.1.2 Patch-fitting

Patch-fitting methods are related to tangent plane approaches. Instead of interpolating the initial point set, a new triangulation patch is formed. AtlasNet [29] is based on this idea and was one of the first learning-based surface reconstruction methods. Small 2D triangulated patches are transformed to fit  $\mathcal{P}$  based on transformations predicted by an MLP. Similar to interpolating approaches, this method cannot guarantee to fill all gaps between patches, which results in a non-watertight and potentially self-intersecting surface.

#### 4.1.3 Surface deformation

One of the only classes of surface-based approaches that can guarantee a watertight surface are deformation-based methods. Sharf *et al.* [28] introduced a method that (i) iteratively expands an intial mesh contained within the input point cloud along the face normal directions, and (ii) moves the mesh vertices to fit the input point cloud using moving

least squares. The method is shown to be robust against missing data, but requires careful parameter tuning to be robust against noise or outliers. Point2Mesh (P2M) [32] is also based on the aforementioned idea, but avoids the need for tuning parameters by hand. The method takes as input a convex hull or a low resolution Poisson reconstruction [33] of  $\mathcal{P}$ , and shrink-wraps this initial surface to best fit the point cloud. The process is guided by multiple local convolutional neural networks (CNNs) that share weights. The idea is that the weight sharing between the CNNs acts as a prior that identifies symmetric features in the shape while being able to ignore unsystematic, random defects in the point cloud. One problem with this approach is that the topology of the initial surface stays constant during reconstruction. If the correct topology of the surface is not known, it cannot be recovered. For example, if the sought surface has holes, they cannot be reconstructed from a convex hull initialisation. This poses a limitation for reconstructing arbitrary objects in the wild.

## 4.2 Volume-based reconstruction

### 4.2.1 Interpolating approaches

Volume-based interpolating approaches commonly start by constructing a 3DT of  $\mathcal{P}$ . In  $\mathbb{R}^3$  a Delaunay triangulation (or tetrahedralization) subdivides the convex hull of  $\mathcal{P}$  with tetrahedra. The 3DT is created in such a way that no point of  $\mathcal{P}$  is contained in the circumspheres of any tetrahedra. For well distributed point clouds it can be constructed in  $O(n \log n)$  [42]. The Delaunay triangulation does not directly generate the surface, as it connects points in any direction. However, if the sampling  $\mathcal{P}$  of  $\mathcal{S}$  is dense enough a subcomplex of the 3DT is guaranteed to include a surface  $\mathcal{S}'$  closely approximating the geometry and topology of  $\mathcal{S}$  [2]. One of the simplest ways to recover this subcomplex from a 3DT is to (i) prune all tetrahedra with circumspheres larger than a user specified constant radius  $\alpha$  and then (ii) keeping only the boundary triangles. This leads to a so-called  $\alpha$ -shape [43]. Similar to the Ball Pivoting algorithm the radius of the ball (here  $\alpha$ ) depends on the point density. For error free and dense samplings, alpha-shapes and some other interpolation methods [2], [41], [44] provide provable guarantees that the reconstructed surface is topologically correct [2]. Another way to recover a surface from a 3DT is inside-outside labelling [10], [10], [34], [45], [46], [47], [48], [49], [50], [51], [52], [53]. Here, all tetrahedra of a 3DT of  $\mathcal{P}$  are (i) labelled as either *inside* or *outside* with respect to  $\mathcal{S}'$ , and (ii) the surface is defined as the interface between tetrahedra with different labels. This guarantees to produce intersecting-free and watertight surfaces. The inside-outside labelling is usually implemented through a global energy minimized with graph-cuts. Inside-outside potentials are computed using visibility information and spatial regularization is achieved through surface smoothness or low area priors in the energy. This approach has been shown to be robust against most kinds of acquisition defects of moderate levels [34], [50], [51] and is capable of reconstructing (very) large scale scenes [49]. Delaunay-Graph Neural Network (DGNN) [10] is a learning-based method that replaces the handcrafted potentials in the aforementioned energy with

a graph neural network (GNN). The GNN takes local geometric attributes and visibility information as input and operates locally on small subgraphs of the 3DT. The locality makes the method scale to large scenes. The method of Luo *et al.* [54] proceeds similarly, but without the use of visibility information and a global energy formulation. Instead, the GNN processes the 3DT of entire objects at once, which can hamper scalability.

### 4.2.2 Implicit functions

Arguably the largest class of surface reconstruction algorithms represent the surface with an implicit function (cf. Equation 1). One of the first methods that used implicit functions for surface reconstruction was presented in Hoppe *et al.* [20]. Hoppe *et al.* (i) calculate tangent planes at each input point of  $\mathcal{P}$ , using principal component analysis (PCA) of the local neighborhood. They then (ii) approximate an SDF by mapping an arbitrary point  $x \in \mathbb{R}^3$  to its signed distance to the closest tangent plane. (iii) The surface is defined as the 0-level-set of the SDF. The local tangent plane estimation makes the process sensitive to low density sampling and noise, and computationally expensive.

Poisson surface reconstruction.: The most popular approach for surface reconstruction based on implicit functions is Poisson Surface Reconstruction (PSR) [55]. The idea is that the Laplacian of an indicator function  $\chi$ , whose  $c$ -level-set approximates the unknown surface  $\mathcal{S}$ , should equate the divergence of a vector field  $\vec{N}$  associated with  $\mathcal{P}$ :

$$\Delta\chi = \nabla \cdot \vec{N}. \quad (2)$$

The vector field  $\vec{N}$  is defined by the oriented normals of  $\mathcal{P}$ . To define  $\chi$  the algorithm (i) builds an octree on  $\mathcal{P}$  and (ii) sets up a system of hierarchical functions, locally supported in each octree node, and (iii) globally solved by using a sparse linear system, which makes the method time and memory efficient. Dirichlet conditions can be imposed on the bounding box of the surface with  $\chi = 0$  to ensure that the surface is closed. The approach is known to inherently produce smooth surfaces, but also over-smooth the surface in parts. The later introduced Screened Poisson Surface Reconstruction (SPSR) [33] can reconstruct much sharper surfaces by constraining Equation 2 to pass through  $\mathcal{P}$ . Additionally, it introduces the choice of Neumann boundary conditions which allows the surface to intersect the boundary of the domain in which  $F$  is defined. This is useful for open scene reconstruction. Recently the method has been revisited again, to impose Dirichlet constraints on a tight envelope around  $\mathcal{P}$ , enabling better reconstructions in areas of missing data [56]. Poisson surface reconstruction produces watertight meshes and has shown to be robust against almost all kinds of acquisition defects of moderate levels. However, all Poisson-based approaches require well oriented normals as input, which can pose a significant limitation in practice.

Neural implicit functions: The most common approach to surface reconstruction with deep networks is to model  $F$  in Equation 1 with a neural network. This was first done in the pioneering works of Mescheder *et al.* [35], Park *et al.* [13], and Chen & Zhang [36].

In the case of Occupancy Networks (ONet) [35],  $F$  is modelled with a simple fully connected network (FCN) architecture. The network takes as input a point cloud  $\mathcal{P}$  and one or several test points  $\mathbf{x}$  and outputs the occupancy of the test points in relation to the surface from which  $\mathcal{P}$  was sampled. The conditioning on the input point cloud slightly changes the formulation of Equation 1 to:

$$\mathcal{S} = \{\mathbf{x} \in \mathbb{R}^3 \mid F_\theta(\mathbf{x}, \mathcal{P}) = c\}. \quad (3)$$

To estimate the network weights  $\theta$ , the network is trained with batches  $\mathcal{B}$  of  $K$  objects using a simple binary cross entropy (BCE) loss:

$$\mathcal{L}_{\mathcal{B}}(\theta) = \frac{1}{|\mathcal{B}|} \sum_{i=1}^{|\mathcal{B}|} \sum_{j=1}^K \text{BCE}(F_\theta(\mathbf{x}_{ij}, \mathcal{P}_i), o_{ij}), \quad (4)$$

where  $o_{ij}$  is the ground truth occupancy of test point  $\mathbf{x}_{ij}$ . To compute the ground truth occupancy  $o_{ij}$ , the training objects have to be available in the form of watertight surfaces. A common approach is to use large shape collections, such as ShapeNet [57] for training. Similar ideas have been introduced in IM-Net [36] and DeepSDF [13] to model an occupancy or signed distance function with a neural network. Instead of an encoder-decoder architecture as in ONet, the authors of DeepSDF [13] introduce an auto-decoder which is trained to find a shape code  $z$  that best explains an object's shape. This slightly changes Equation 3 and Equation 4, where the point cloud input  $\mathcal{P}$  is replaced by a shape code  $z$  in the form of a 256-dimensional vector. The DeepSDF architecture then allows to reconstruct a complete signed distance field (and thus the shape), given a shape code  $z$ . However, to find the shape code for a specific shape during inference, at least a few ground truth signed distance values are necessary. This can be a significant limitation in practice. A common downside of the first DSR networks based on neural implicit fields is their simple fully connected network architecture. This architecture does not allow the incorporation of local point cloud information [6] and often leads to oversmoothing or inaccuracies of the inferred surface.

To this end, occupancy networks have later been refined by prepending 2D or 3D U-Nets [58], [59] before the fully connected occupancy network, to better incorporate local information. The idea is to (i) extract point features from local neighborhoods and (ii) aggregate these features in 2D or 3D grid cells. The U-Nets are then used to (iii) integrate local and global information using multiple down- and up-samplings. (iv) Finally, the fully connected ONet is used to compute test point occupancies. The approach is called Convolutional Occupancy Networks (ConvONet) [6]. Just as for the fully connected architectures, the network can be trained with test points  $\mathbf{x}$  with known occupancy values  $o$ . In the same work, the authors also introduce an overlapping sliding-window approach in which a single trained ConvONet can be used to reconstruct entire indoor scenes. However, this approach requires to carefully scale the scene, such that the sliding window captures parts of the scene with comparable surface features during training and inference. Furthermore, for large-scale scenes, a sliding-window approach can be very time-consuming.

Local Implicit Grids (LIG) and DeepLS [7] also split input point clouds into overlapping subregions, and treat each subregion separately. The methods infer local shape codes  $z$  for parts of objects or scenes. These local shape codes have the additional benefit that they can represent parts from several different object classes. For example, a flat part-surface may belong to a table top or to a TV screen. This makes the methods less prone to overfit on specific shape categories used during training. However, the methods are largely based on IM-Net and DeepSDF. This means they also require a sort of ground truth test point during inference to optimize for the shape codes. Additionally, similar to the sliding window method of ConvONet, the region size (*i.e.* part size) has to be tuned.

Using the same encoder architecture as ConvONet, Shape As Points (SAP) [38] introduces the combination of neural implicit fields with a differentiable Poisson solver. The method estimates (i) oriented normals as well as  $k$  point offsets for each input point, to correct and densify the point cloud  $\mathcal{P}$ . (ii) The resulting point cloud of size  $k|\mathcal{P}|$  is fed to a differentiable Poisson solver [33] that computes an indicator grid, *i.e.*  $\hat{\chi}$  evaluated on all nodes of a regular voxel grid. (iii) This indicator grid is supervised with a ground truth indicator grid  $\chi$ . The ground truth indicator grid is created prior to training, from a Poisson reconstruction of a dense and error free point cloud, sampled from a ground truth mesh. A simple mean square error (MSE) loss is used for training the network:

$$\mathcal{L} = |\hat{\chi} - \chi|^2 \quad (5)$$

The entire pipeline is differentiable which allows to update point offsets, oriented normals and the network parameters during training (with batches of shapes). During inference, the computed indicator grid can simply be converted to a mesh using marching cubes. In contrast to the original Poisson Surface Reconstruction, SAP allows to incorporate learned priors and does not need  $\mathcal{P}$  to be equipped with oriented normals.

In general, all of the methods based on voxel grids in this paragraph require the size of the initial voxels to be constant during training, because the resolution of the convolution layers depends on the voxel grid. This poses problems for training on point clouds with different densities. A dense voxel grid can be memory intensive and long to train, while a coarse voxel grid can oversmooth the input and lead to loss of information.

Another way to combine local and global information, that avoids the use of grids was introduced in Points2Surf (P2S). P2S uses both a local test point neighborhood sampling, and a global point cloud sampling which are both processed using MLPs and combined to predict a signed distance for the test point. The  $k$ -nearest neighbor sampling makes this method less sensitive to point density, at the cost of increasing computational complexity, since the local neighborhood sampling has to be performed for each test point during inference.

Point Convolution for Surface Reconstruction (POCO) only relies on local neighborhoods and computes a latent vector per point using a point convolution backbone. The

occupancy of a test point  $x$  is then predicted using attention-based weighing of neighboring latent vectors. This approach can focus the parameters of the learned implicit function to be used close to the surface. However, it also requires neighborhood sampling during inference. Similar to most other DSR methods, POCO is trained on object point clouds with a fixed number of points for easy mini-batching. However, to make the method more robust to point clouds with higher density during inference, the authors use a procedure called test-time augmentation. During inference, the latent vectors of each input point  $p$  are computed several times, from different local subsamples and then averaged.

Another approach to use neural implicit surface representations is to "train" (or optimize) the weights of a deep neural network per shape [37], [38]. The idea is to leverage inherent symmetries of deep neural networks to act as priors in the reconstruction process, similar to the surface deformation based Point2Mesh discussed above. To this end, Gropp *et al.* [37] designed a simple fully connected network representing a signed distance function. To encourage the reconstruction of a smooth 0-level-set, given an input point cloud  $\mathcal{P}$ , they design a loss function which (i) should vanish on  $\mathcal{P}$  and (ii) which gradients  $\Delta_{\mathcal{P}} F$  should be of unit 2-norm and similar to the normals of  $\mathcal{P}$ . The method is called Implicit Geometric Regularisation (IGR). SAP also has an optimization-based variant where (i) the indicator grid, computed with the differential Poisson solver from the input point cloud  $\mathcal{P}$  is used to compute a mesh. (ii) The mesh is then sampled, which allows to calculate a Chamfer loss between the sampled and input point cloud and, again, update the network weights, point offsets and oriented normals. (iii) This process is repeated until a user defined stopping criterion. The optimization-based variants of SAP and IGR can be trained per shape, without the need for ground truth meshes for supervision. However, in this optimization-based setting, they cannot learn and incorporate shape priors from a training set.

An upside of all DSR methods based on neural implicit representations is that they can store an implicit function, potentially conditioned on a point cloud, in the weights of a neural network. Especially DSR architectures that are entirely grid-less can directly relate their degrees of freedom to represent the surface. This can be more flexible compared to voxel, octree, or tetrahedral representations. Being a relatively new discovery, the full potential of neural network-based surface representations has probably yet to be explored.

## 5 BENCHMARK SETUP

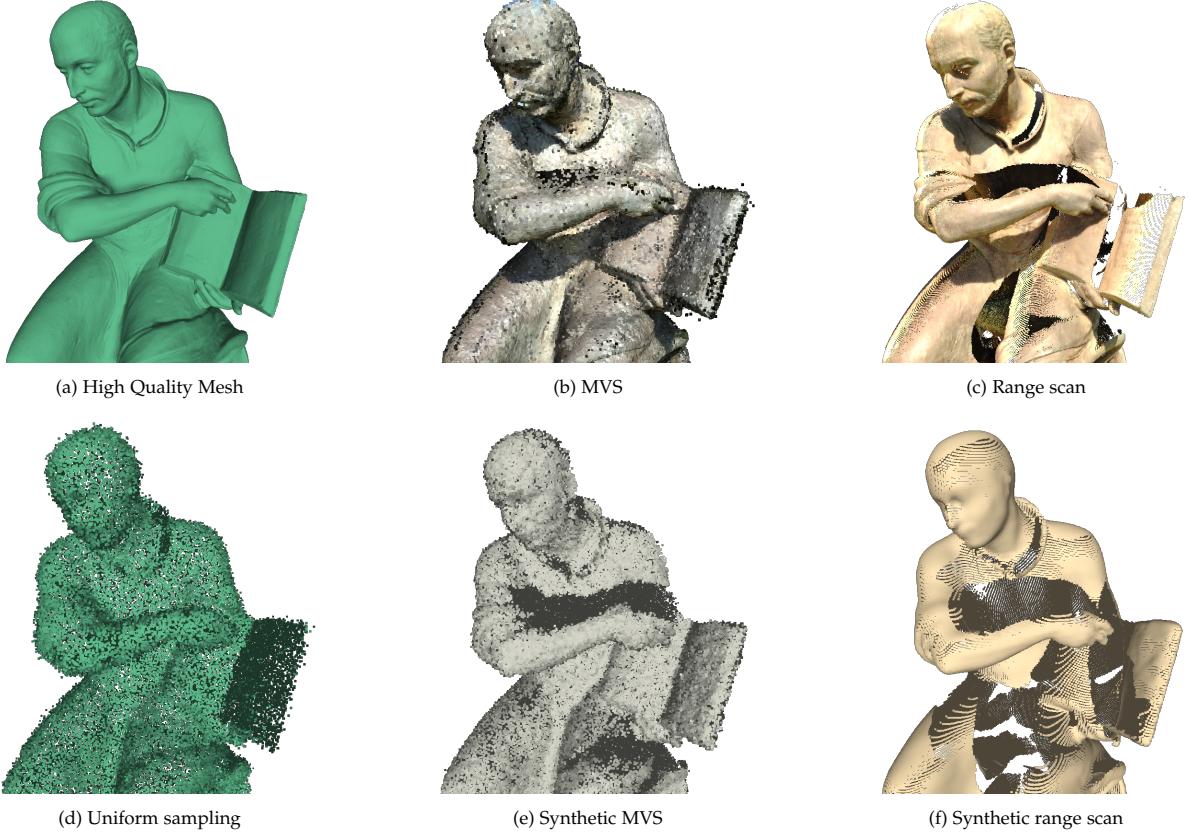
In this section, we describe our set up of a series of experiments for benchmarking several surface reconstruction algorithms discussed in the previous section. We first describe how we generate realistic point clouds by using synthetic range and MVS scanning procedures. We then describe the datasets we used and several experiments to evaluate the performance of reconstruction methods. Finally, we provide an overview of the competing methods.

**Synthetic scanning for point cloud generation:** In an ideal setting, we would evaluate methods on real point

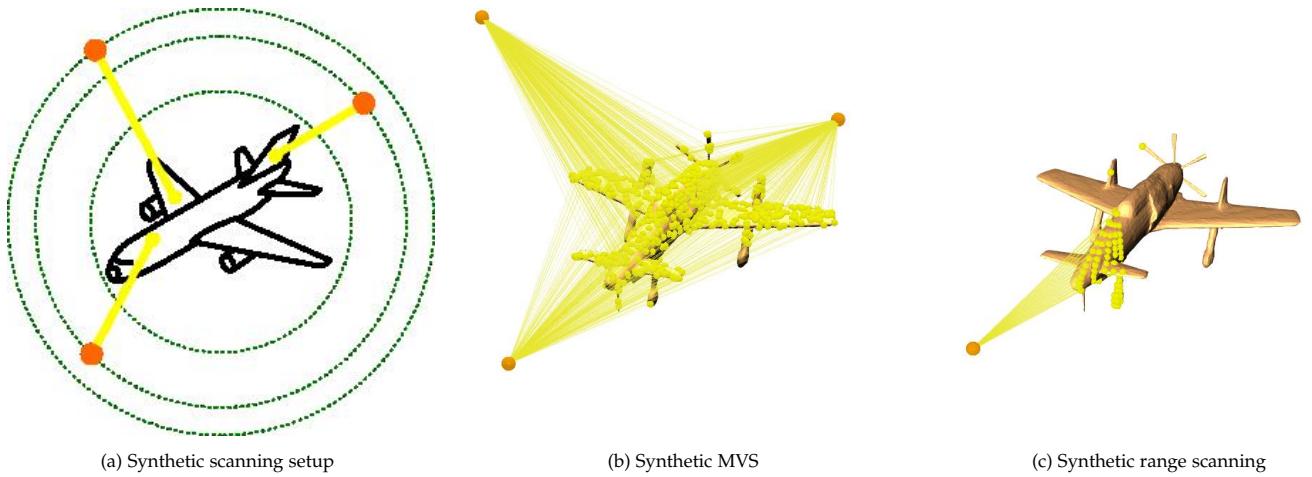
cloud acquisitions together with their true surfaces. However, generating true surfaces of real objects requires error free and dense input point clouds or substantial manual intervention. Therefore, such a dataset is difficult to produce. MVS benchmarks [15], [16], [17], [18], [19] commonly use image acquisitions for the reconstruction input and a highly complete and precise acquisition, *e.g.* from multiple stationary Light Detection and Ranging (LiDAR) scans as reference. We make use of such datasets for evaluation. Using such a dataset for training surface reconstruction networks requires reconstructing a watertight surface from the high-quality acquisition. However, even with high-quality acquisitions, parts of the object or scene may be missing due to occlusions, for example. These issues ultimately lead to inconsistencies in the ground truth and make this source of data unreliable to train DSR networks. Additionally, existing datasets of point cloud acquisitions and reliable ground truth surface information only consist of a handful of objects or scenes. Instead, training and evaluation of learning-based surface reconstruction is often done on point clouds sampled from synthetic surfaces stemming from large shape collections. However, such point clouds are not representative for real-world acquisitions, as they do not model non-uniformity or missing data stemming *e.g.* from occlusions, or transparent and low texture areas. To this end, we resort to synthetic scanning to produce point clouds from synthetic surfaces in our benchmark. In contrast to directly sampling the surfaces, synthetic scanning can produce point clouds with realistic defects, such as anisotropy and missing data from (self-)occlusion, see Figure 3. At the same time, the synthetic surfaces provide reliable information for training and evaluation.

**Synthetic range scanning:** We use the range scanning procedure from the surface reconstruction benchmark of Berger *et al.* [14]. To this end, we modified their provided code to export the camera positions of the scanning process along with the point cloud. We also add outliers to the produced point clouds by uniformly sampling the bounding box of the object. The scanning procedure produces uniform, evenly spaced point clouds. We choose five different scanner settings to scan each test shape: (i) a low resolution setting replicates point clouds obtained from long range scanning and (ii) a high resolution setting produces point clouds with close to no defects. Three further settings produce high resolution point clouds with challenging defects such as (iii) noise, (iv) outliers or (v) noise and outlier defects combined. See the supplementary material for details. Because Berger *et al.*'s provided code pipeline is too time and memory extensive, we cannot generate a dataset sufficiently large for training DSR methods. Thus, we only use this dataset for testing. We refer the reader to the original benchmark paper [14] for further details about the scanning pipeline.

**Synthetic MVS:** To mimic MVS acquisitions, we synthetically scan objects by placing virtual sensors on two bounding spheres around an object and shooting rays to the circumsphere of the object. Sensor positions (ray origin) and ray target points are uniformly sampled on the surface of the spheres. A 3D point is then given as the intersection of the ray and the objects surface. Our goal is not to mimic an MVS pipeline but rather produce point clouds with similar characteristics. We depict our scanning procedure in



**Figure 3: Synthetic and real point clouds:** Surface reconstruction methods are often tested on uniform surface samplings (d). Instead, we test methods on synthetic MVS (e) and synthetic range scans (f). In contrast to uniform surface sampling, synthetic scanning can produce realistic point cloud defects, such as missing data from occlusion, often present in real scans (b,c).



**Figure 4: Synthetic scanning procedure:** We randomly place sensors on bounding spheres with multiple radii around the object (a). To produce MVS like point clouds, we consider rays aiming at uniformly sampled points on the circumsphere of the object (b). This produces non-uniform point clouds with missing data similar to real MVS point clouds. For synthetic range scanning, we use Berger *et al.*'s [14] pipeline, which considers ray targets arranged on a uniform grid aiming at the object (c). This produces uniform point clouds with missing data similar to real range scanning point clouds.

Figure 4. We produce two different scans with our approach: (i) sparse point clouds with 3,000 points per object and Gaussian noise on the point position with zero mean and standard deviation 0.005 as in [6], and (ii) dense point clouds with 10,000 points per object of which 10% are outliers and Gaussian noise on the point position with zero mean and standard deviation 0.005. For both versions we scan from 10 different sensor positions.

## 5.1 Datasets

We consider a variety of datasets to evaluate the versatility and precision of different reconstruction methods. We use closed surfaces from ShapeNet, ModelNet and Berger *et al.*, as they are widely available. ShapeNet and ModelNet are sufficiently big to train surface reconstruction networks. Most learning-based methods require reliable inside/outside querying of the models for training. To this end, we make the models watertight using ManifoldPlus [60]. Note that we also use the train sets to tune the parameters of learning-free methods. The watertight surfaces of the test sets allow for a reliable quantitative evaluation of the reconstructions. For qualitative evaluation, we also test on real scans [15], [16], [19] which further allows us to evaluate the reconstruction of open surfaces. All surfaces are scaled to be contained inside the unit cube. In the following we give additional details for each dataset used in our benchmark. See the supplementary material for example shapes.

**ShapeNet:** As is common practice in related studies, we use Choy *et al.*'s [61] 13 class subset of ShapeNet as well as its train/val/test split. We generate point clouds with 3,000 and 10,000 points using our synthetic MVS-like scanning.

**ModelNet10:** We use ModelNet10 shapes as a second object shape dataset. Its shapes are less complex than ShapeNet's, with more flat surfaces and fewer details. Additionally, the number of training shapes is smaller (4k vs 30k objects). We use the full train set and the test sets for the 6 out of 10 classes which are not represented in ShapeNet (see supplementary material for details). We generate point clouds with 3,000 points with our synthetic MVS-like scanning.

**Berger *et al.*:** We select five shapes from the benchmark of Berger *et al.*. These shapes include challenging characteristics such as details of various sizes or a non-trivial topology, which makes them more difficult to reconstruct than ModelNet shapes. We generate point clouds between 3,000 and 10,000 points using our synthetic MVS and range scanning procedures.

**Real MVS and range scans:** We select a range scan from Tanks and Temples [19], and two MVS point clouds from DTU [16] and from Middlebury [15]. We subsample these point clouds to 50,000 points.

## 5.2 Experimental Setup

We show a summary of our experimental setup on Table 2. In the following, we provide details for each experiment.

**In-distribution (E1):** First, we train and evaluate methods on ShapeNet using all 13 categories and sparse point clouds with 3,000 points and Gaussian noise with

zero mean and standard deviation 0.005. With this experiment, we evaluate the capacity of learning methods to complete missing data of sparse point clouds and eliminate noise.

**Out-of-distribution (unseen point cloud characteristics) (E2):** We evaluate the models trained in E1 on test shapes scanned with a different setting than the train shapes. We use dense point clouds with 10,000 points of which 10% are outliers. We add the same noise as in E1. Here, we investigate whether learning methods are able to generalize to different point cloud characteristics.

**Out-of-distribution (unseen shape categories, less complex) (E3):** We evaluate the models trained in E1 on shapes from unseen categories but with the same point cloud characteristics. We use six categories of ModelNet which are not present in the ShapeNet training set. In this experiment, we investigate whether learning methods generalize to unseen categories.

**Out-of-distribution (unseen shape categories, similar complexity) (E4):** This experiment is similar to E3, but the test set is comprised of five shapes from Berger *et al.* which do not correspond to ShapeNet's categories, but have similar complexity.

**Out-of-distribution (unseen shape categories, more complex) (E5):** This experiment is similar to E3 and E4, but we retrain all methods on the simpler shapes from ModelNet10. Here, we assess whether learning methods can generalize from simple shapes to more complex ones, a difficult out-of-distribution setting.

**Optimization (E6):** We evaluate several recently developed optimization-based methods, and two traditional test-of-time optimization-based methods. We use the Berger *et al.* dataset for this experiment.

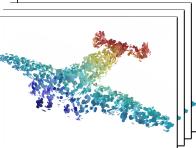
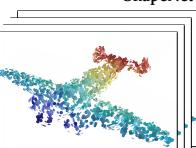
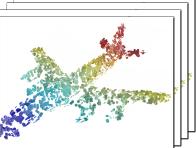
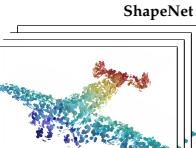
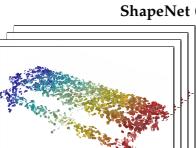
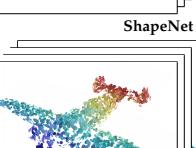
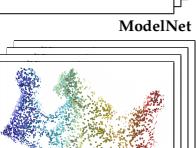
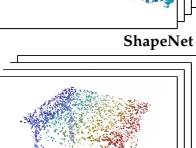
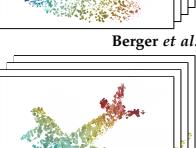
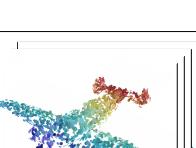
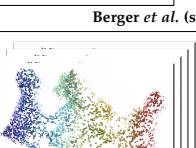
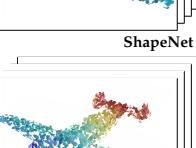
**Out-of-category vs. optimization (E7):** We compare learning- and optimization-based methods on the same dataset. For this we run optimization-based methods on MVS scans of the Berger *et al.* shapes and compare the results to experiment E4.

**Out-of-distribution vs. optimization (E8):** Finally, we compare learning- and optimization-based methods on real MVS and range scanning point clouds. For learning-based methods we use the models from E1.

## 5.3 Surface reconstruction methods

We briefly describe the optimization- and learning-based methods that we will benchmark below. For a more complete description of these methods and their related concepts we refer the reader to our survey in Section 4. Note that while some of the optimization-based methods are based on deep networks, and we call them DSR methods, they do not learn shape priors from a training set. Instead, the networks are “trained” (or optimized) for each new point cloud to reconstruct a surface and rely on novel regularization techniques to increase their robustness to noise, outliers and missing data. Conversely, while some traditional methods are not based on a deep network architecture, we tune their (hyper)parameters on the training set by using a grid search over different parameter combinations. When we need to extract a surface from an implicit field, we use marching cubes [62] with a resolution of  $128^3$ .

**Table 2: Benchmark setup:** Overview of our experimental setup. In E1 to E5, we train surface reconstruction methods on noisy point clouds of ShapeNet objects. In E1, we test on the ShapeNet test set. In E2, we test on ShapeNet, but from denser point clouds with noise and outliers. In E3, we test on the simpler ModelNet objects with the same sampling as in E1. In E4, we test on five Berger *et al.* shapes with the same sampling as in E1. In E5, we train the methods on the simpler ModelNet dataset and test on ShapeNet, both with the same sampling as in E1. In E6, we test optimization-based methods on synthetic range scans of the Berger *et al.* dataset. Finally, in E7 and E8, we directly compare learning- and optimization-based methods on synthetic and real scans.

Experiment	Training set	Test set
1 In-distribution	 ShapeNet (synthetic MVS)	
2 Out-of-distribution <i>unseen point cloud characteristics</i>	 ShapeNet (synthetic MVS)	 ShapeNet (synthetic MVS)
3 Out-of-distribution <i>unseen shape categories, less complex</i>	 ShapeNet (synthetic MVS)	 ModelNet (synthetic MVS)
4 Out-of-distribution <i>unseen shape categories, similar complexity</i>	 ShapeNet (synthetic MVS)	 Berger <i>et al.</i> (synthetic MVS)
5 Out-of-distribution <i>unseen shape categories, more complex</i>	 ModelNet (synthetic MVS)	 ShapeNet (synthetic MVS)
6 Optimization	—	 Berger <i>et al.</i> (synthetic range scan)
7 Out-of-distribution vs. optimization <i>unseen shape categories vs. optimization</i>	 ShapeNet (synthetic MVS)	 Berger <i>et al.</i> (synthetic MVS)
8 Out-of-distribution vs. optimization <i>unseen point cloud characteristics and shape categories vs. optimization</i>	 ShapeNet (synthetic MVS)	 Middlebury, DTU (MVS), TaT (range scan)

### 5.3.1 Optimization-based methods

IGR [37]: Implicit Geometric Regularisation (IGR) is a DSR method, operating directly on the point cloud using a simple fully connected network architecture that estimates an indicator function from point positions and normals. We optimize the network weights for 100,000 iterations for each scan/shape.

LIG [8]: Local Implicit Grids (LIG) trains an autoencoder to encode crops of a signed distance function gained from ground truth shapes. For inference, only the decoder part of the autoencoder is retained. Then, crops of the input point cloud with oriented normals are augmented with 10 new points along each normal, representing ground truth signed distance information. An initial latent vector is then decoded to produce an SDF and iteratively optimized so that the augmented point cloud crop best matches the SDF. A post-processing removes falsely-enclosed volumes. As code for training is unavailable, we only use the optimization part, with a pretrained model on ShapeNet (without noise). We use the sensor position to orient jet-estimated normals [63].

P2M [32]: Point2Mesh (P2M) is an optimization-based method which iteratively moves vertices of an initial mesh to fit a point cloud.

SAP [38]: Shape As Points (SAP) has a supervised learning- and an optimization-based variant. In the learning variant, the method estimates the oriented normals as well as  $k$  point offsets for each input point, to adjust and densify the point cloud. The resulting point cloud of size  $k + |\mathcal{P}|$  is then used by a differentiable Poisson solver [33] to compute an indicator grid, which is supervised with a ground truth indicator grid computed prior to training. The entire pipeline is differentiable which allows for updating point offsets, oriented normals and the network parameters.

SPSR [33]: Screened Poisson Surface Reconstruction (SPSR) is a classic non learning-based method which approximates the surface as a level-set of an implicit function estimated from point positions and normal information. We use the sensor position to orient jet-estimated normals [63]. We chose an octree of depth 10 and Dirichlet boundary condition. We also use the provided surface trimming tool for post-processing, but could not find parameters that consistently improve the reconstructed surface.

Labatut *et al.* [34]: Labatut *et al.* is a graph-cut-based method for range scans that makes use of visibility information. Because there is no official implementation of the algorithm, we reimplemented it ourselves. To compare with optimization-based methods, we use the parametrization suggested by the authors: point weights  $\alpha_{vis} = 32$  and  $\sigma = 0.01$ ; regularization strength  $\lambda = 5$ .

### 5.3.2 Learning-based methods

ConvONet [6]: Convolutional Occupancy Networks (ConvONet) is a DSR method that first extracts point features and averages them on cells of three 2D grids, or one 3D grid (variant). 2D or 3D grid convolutions then create features capturing the local geometry. Last, the occupancy of a query-point is estimated with a fully connected network from interpolated features stored on each node of the 2D or 3D grid.

SAP [38]: In the optimization variant, the method starts as the learning-based variant described above. Then, the estimated indicator grid is used to compute a mesh and points are sampled on the mesh to calculate a Chamfer loss between the mesh and input point cloud.

DGNN [10]: This method uses a graph neural network to estimate the occupancy of Delaunay cells in a point cloud tetrahedralization from cell geometry and visibility features. A graph-cut-based optimization then reinforces global consistency.

POCO [12]: Point Convolution for Surface Reconstruction (POCO) extracts point features using point cloud convolution [64], then estimates the occupancy of a query point with a learning-based interpolation from nearest neighbors.

SPRS [33]: See method description above. For the learning-based experiments, we perform a grid search over octree depth  $d = \{6, 8, 10, 12\}$  and boundary conditions  $b = \{\text{dirichlet}, \text{neumann}, \text{free}\}$ . We use the parametrization with the best mean volumetric IoU for reconstructions of the training set.

Labatut *et al.* [34]: See method description above. For the learning-based experiments, we perform a grid search over regularization strength  $\lambda = \{1.5, 2.5, 5, 10\}$ , and point weights  $\alpha = \{16, 32, 48\}$  and  $\sigma = \{0.001, 0.01, 0.1, 1\}$ . We use the parametrization with the best mean volumetric IoU for reconstructions of the training set.

## 5.4 Evaluation metrics

We want the reconstructed surface  $\mathcal{S}^r$  to be as close as possible to the real (or ground truth) surface  $\mathcal{S}$  in terms of geometry and topology. To measure this “closeness” we use several metrics.

### 5.4.1 Geometric metrics

We evaluate the geometric quality of reconstructions with the volumetric intersection over union (IoU), symmetric Chamfer distance (CD) and normal consistency (NC).

Volumetric IoU: In the following, let  $\mathcal{S}^g$  and  $\mathcal{S}^r$  be the set of all points that are inside or on the ground truth and reconstructed surface, respectively. The volumetric IoU is defined as:

$$\text{IoU}(\mathcal{S}^g, \mathcal{S}^r) = \frac{|\mathcal{S}^g \cap \mathcal{S}^r|}{|\mathcal{S}^g \cup \mathcal{S}^r|}.$$

We approximate volumetric IoU by randomly sampling 100,000 points in the union of the bounding boxes of  $\mathcal{S}^g$  and  $\mathcal{S}^r$ .

Chamfer distance: To compute the Chamfer distance and normal consistency, we sample a set of points  $\mathcal{P}^g$  and  $\mathcal{P}^r$  on the facets of the ground truth mesh and the reconstructed mesh, respectively, with  $|\mathcal{P}^g| = |\mathcal{P}^r| = 100,000$ . We approximate the symmetric Chamfer distance between  $\mathcal{S}^g$  and  $\mathcal{S}^r$  as follows:

$$\begin{aligned} \text{CD}(\mathcal{S}^g, \mathcal{S}^r) &= \frac{1}{2|\mathcal{P}^g|} \sum_{x \in \mathcal{P}^g} \min_{y \in \mathcal{P}^r} \|x - y\|_2 \\ &\quad + \frac{1}{2|\mathcal{P}^r|} \sum_{y \in \mathcal{P}^r} \min_{x \in \mathcal{P}^g} \|y - x\|_2. \end{aligned}$$

Normal consistency: Let  $n(x)$  be the unit normal of a point  $x$ . We set this normal to be the normal of the facet from which  $x$  was sampled. Let  $\langle \cdot, \cdot \rangle$  the Euclidean scalar product in  $\mathbb{R}^3$ . Normal consistency is defined as:

$$\begin{aligned} \text{NC}(\mathcal{S}^g, \mathcal{S}^r) = & \frac{1}{2|\mathcal{P}^g|} \sum_{x \in \mathcal{P}^g} \left\langle n(x), n \left( \operatorname{argmin}_{y \in \mathcal{P}^r} \|x - y\|_2 \right) \right\rangle \\ & + \frac{1}{2|\mathcal{P}^r|} \sum_{y \in \mathcal{P}^r} \left\langle n(y), n \left( \operatorname{argmin}_{x \in \mathcal{P}^g} \|y - x\|_2 \right) \right\rangle. \end{aligned}$$

#### 5.4.2 Topological metrics

We evaluate the topological quality of reconstructions through the number of components, the number of non-manifold edges and the number of boundary edges.

Number of components: If not stated otherwise, the ground truth surfaces of our datasets have exactly one component. In consequence, the reconstructed surfaces should also have one component.

Number of boundary edges: The surfaces of all ground truth objects in our datasets are closed. We verify this by measuring the number of boundary edges of the reconstructed meshed surface which should be zero. Note that if boundary edges only appear on the intersection of the reconstruction with its bounding box we still classify the reconstruction as watertight, according to the definition in Section 3.3.

Number of non-manifold edges: The surfaces of all ground truth objects in our datasets are 2-manifolds. We verify this by measuring the number of non-manifold edges of the reconstructed meshed surface which should be zero.

#### 5.4.3 Runtimes

To evaluate the scalability of methods, we measure the average time it takes to reconstruct a surface of ShapeNet from 3,000 points.

## 6 EXPERIMENTS

### 6.1 Learning-based surface reconstruction from synthetic MVS point clouds (E1 - E5)

We examine the precision and versatility of novel supervised-learning methods and two traditional methods for which training sets were used for tuning parameters. All evaluated methods perform well when reconstructing shapes from known categories and known point cloud characteristics (E1). The learning-based methods show a significantly superior performance of at least 5% over SPSR and Labatut *et al.* (see Table 3). The methods based on neural implicit fields (POCO, SAP and ConvONet) produce visually and quantitatively the best reconstructions (see Figure 5, first column). DGNN does not perform as well as most other learning methods in this experiment. The sparse point clouds used in this experiment do not contain point samples on all details. However, due to the interpolating nature of DGNN surface details cannot be reconstructed without input points.

In E2, domain shifts results in worse performance, both quantitatively and qualitatively for all methods except SPSR. SPSR shows robustness against outliers and benefits from the higher point density. Most learning methods do

not produce satisfying results (see Figure 5, second column). The reconstruction of SAP is too smooth and lacks details, but does not show as severe defects as the reconstructions of other learning-based methods. Labatut *et al.* suffers from the low regularization weight tuned for the outlier free point clouds and could benefit from higher regularization to remove erroneous floating components from outliers.

When reconstructing out-of-category ModelNet shapes (E3), the neural implicit field methods exhibit visually the best reconstructions. SAP and POCO produce quantitatively the best reconstructions (see Table 3). The interpolating method DGNN performs better than ConvONet.

In E4, we reconstruct shapes from Berger *et al.* which have similar complexity than the shapes from ShapeNet used for training. The only learning methods able to leverage information from the common point cloud characteristics to improve the test results are DGNN and POCO.

In E5, most methods overfit the simpler ModelNet shapes when retrained and used to reconstruct the more complex ShapeNet shapes. Even SPSR slightly suffers from tuning parameters on ModelNet. The best reconstructions on ModelNet are achieved with an octree depth of  $d = 8$  (instead of  $d = 10$  on ShapeNet) leading to worse results on ShapeNet: 77.1 vIoU in E1 vs. 74.6 vIoU in E5. The parameter tuning of Labatut *et al.* stays unchanged. DGNN is the only method that does not overfit on ModelNet and yields the best results, both quantitatively and qualitatively. In fact, it performs as well as when trained on ShapeNet directly.

ConvONet is only able to outperform traditional methods when the training and test sets share the same point cloud characteristics *and* shape categories. SAP produces much better reconstructions and is the learning-based method with the highest robustness against outliers. It is also the only method explicitly predicting normals. As a result SAP reconstructs surfaces with the highest mean normal consistency over all experiments. The local learning and global regularisation approach of DGNN produces competitive results in all experiments, except for the outlier setting of E2. DGNN is the learning-based method producing surfaces with highest mean IoU over all experiments. The local attention-based learning mechanism of POCO leads to the best results when the task does not involve reconstruction from unseen domains. It provides the most faithful reconstructions in three experiments in which point cloud characteristics are identical in train and test set (E1, E3, E4). However, POCO is heavily affected by outliers (E2), which can be explained by its purely local approach. POCO also tends to overfit on simple training shapes (E5). The reconstructions of POCO, as well as the ones of SAP contain boundary edges only in areas where the reconstructions intersect the bounding box *i.e.* they are still watertight. SPSR proves robust to various defects and shape characteristics, providing fair results, with the highest mean IoU and Chamfer distance across the board. However, its reconstructions are the least compact, *i.e.* they have the highest number of components. Labatut *et al.*'s parametrization proves slightly less robust, as the method is affected by outliers. Its mean IoU is higher than that of any learning method, and its reconstructions are the most compact surfaces with an average number of components of 2.7. However, it is also the only



Figure 5: **Learning-based reconstructions (E1 to E5):** In each column we show learning-based reconstructions of experiments E1 to E5. DGNN [10], SAP [38] and SPSR [33] provide visually the best results with exhibiting dominant defects.

**Table 3: Numerical results for learning-based experiments (E1 to E5):** We show the numerical results of the learning experiments E1 to E5. SPSR [33] is the only method that produces surfaces with a high volumetric intersection over union and a low Chamfer distance in each experiment. Therefore, its surfaces have the highest mean volumetric IoU and the lowest mean CD. However, SPSR also produces the least compact surfaces on average (*i.e.* surfaces with the highest number of components). Labatut *et al.* [34] produces the most compact surfaces. DGNN [10] has the highest mean volumetric IoU of the tested learning methods. SAP [38] has the lowest mean CD of the tested learning methods and the highest normal consistency. ConvONet and SPSR are the only methods that produce surfaces without boundary and non-manifold edges.

Method	Volumetric IoU (%) [ $\uparrow$ ]						Normal consistency (%) [ $\uparrow$ ]					
	E1	E2	E3	E4	E5	Mean	E1	E2	E3	E4	E5	Mean
ConvONet2D [6]	85	47.3	79.3	65.1	68.3	69	92.7	76.4	90	78	87.8	85
ConvONet3D [6]	84.8	15.1	83.6	76.4	51	62.2	93	71.8	93.1	87.2	82.5	85.5
SAP [38]	88.7	59.8	89.2	78.3	54.9	74.2	93.5	<b>86.7</b>	94.1	89	87.1	<b>90.1</b>
DGNN [10]	84.5	38.1	87	82.9	<b>84.4</b>	75.4	85.4	68.8	88.5	85.2	85.5	82.7
POCO [12]	<b>89.5</b>	8.74	<b>90.6</b>	<b>83.9</b>	40.9	62.7	<b>93.6</b>	75.6	<b>94.2</b>	<b>89.5</b>	82.9	87.1
SPSR [33]	77.1	<b>80.7</b>	80.7	77.6	74.6	<b>78.1</b>	87.7	83.2	89.1	86.3	<b>88</b>	86.9
Labatut <i>et al.</i> [34]	80.3	60.4	83.9	79.4	80.3	76.9	81	73	84.6	80.8	81	80.1
Chamfer distance (per-point ave. %) [ $\downarrow$ ]												
Method	E1	E2	E3	E4	E5	Mean	E1	E2	E3	E4	E5	Mean
ConvONet2D [6]	0.553	7.51	0.997	1.43	0.979	2.29	1.6	34.8	2.55	3.6	3.2	9.16
ConvONet3D [6]	0.546	10.9	0.76	0.887	2.44	3.1	1.37	13.6	1.6	2.6	1.5	4.13
SAP [38]	0.437	2.09	0.547	0.734	0.924	0.946	2.71	86	3.45	5.6	10.5	21.7
DGNN [10]	0.549	2.54	0.635	0.586	<b>0.55</b>	0.973	1.31	16.1	1.13	<b>1</b>	1.31	4.16
POCO [12]	<b>0.416</b>	10.5	<b>0.516</b>	<b>0.579</b>	1.32	2.67	2.32	178	2.82	2	16.3	40.2
SPSR [33]	0.801	<b>0.659</b>	0.873	0.786	0.886	<b>0.801</b>	9.26	185	11.1	8	3.24	43.3
Labatut <i>et al.</i> [34]	0.665	6.97	0.747	0.671	0.665	1.94	<b>1.22</b>	<b>9.02</b>	<b>1.05</b>	<b>1</b>	<b>1.22</b>	2.7
Number of boundary edges [ $\downarrow$ ]												
Method	E1	E2	E3	E4	E5	Mean	E1	E2	E3	E4	E5	Mean
ConvONet2D [6]	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>
ConvONet3D [6]	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>
SAP [38]	<b>0</b>	0.00923	<b>0</b>	<b>0</b>	8.44	1.69	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>
DGNN [10]	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	1.35	2.24	0.646	0.4	1.69	1.26
POCO [12]	<b>0</b>	121	<b>0</b>	<b>0</b>	41.7	32.5	<b>0</b>	0.00154	<b>0</b>	<b>0</b>	<b>0</b>	0.000308
SPSR [33]	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>
Labatut <i>et al.</i> [34]	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	9.35	28.5	8.47	9.6	9.35	13.1

method that produces a significant amount of non-manifold edges.

## 6.2 Optimization-based surface reconstruction from synthetic range scanning point clouds (E6)

This experiment evaluates the precision and versatility of non-learning methods. The benchmarked approaches consist in neural network based methods optimizing a function to fit an input point cloud and rely on novel regularization techniques to increase their robustness to noise, outliers and missing data. Furthermore, we benchmark the two traditional methods SPSR and Labatut *et al.* with standard parameter settings. We reconstruct surfaces of Berger *et al.* from synthetic range scanning point clouds with various different defects. We show numerical results in Table 4 and visualisations in the supplementary material. Almost all reconstructions provided by the two traditional methods are much more truthful than the DSR methods, with a mean volumetric IoU almost 10 points higher across all point cloud defects. IGR does visually not provide a good result on the exemplary shape, especially on thin surface parts. Quantitatively, the method provides the best reconstruction for the neural networks based methods in the absence of outliers, and even the best overall reconstruction for the noisy high resolution scans. LIG does not provide good

reconstructions for any of the settings. This can be explained by its pretrained model on defect-free uniform high density point clouds. Furthermore, its post-processing makes the reconstructions non-watertight. P2M provides geometrically fair reconstructions and the topologically best reconstructions with a low number of components, and watertight and manifold surfaces for all reconstructions. SAP provides fair reconstructions in the absence of outliers. None of the neural network based methods is robust against outliers. As in the learning-based experiments, SPSR generates high quality reconstructions for all input defects, and achieves the best mean normal consistency. Labatut *et al.* achieves the best mean IoU and mean Chamfer distance while providing the reconstructions with the lowest number of components. However, the reconstructions of Labatut *et al.* are the only ones with a significant number of non-manifold edges.

## 6.3 Learning- and optimization-based surface reconstruction from synthetic MVS point clouds (E7)

To directly compare learning- and optimization-based reconstructions on the same dataset, we also reconstruct the Berger *et al.* shapes from synthetic MVS scans (cf. E4) with the optimization-based methods. Thus, for learning-based methods, we use the models trained on synthetic MVS scans from ShapeNet (cf. E4) and we optimize non-learning

**Table 4: Numerical results for optimization-based reconstructions (E6):** Optimization-based reconstruction of the Berger *et al.* shapes from synthetic range scans. LR is a low resolution scan, HR a high resolution scan, HRN a high resolution scan with noise, HRO a high resolution scan with outliers, and HRNO a high resolution scan with noise and outliers. The methods are optimized per shape and per scan using standard settings as mentioned in the corresponding publications.

Method	Volumetric IoU (%) [↑]						Normal consistency (%) [↑]						
	LR	HR	HRN	HRO	HRNO	Mean	LR	HR	HRN	HRO	HRNO	Mean	
IGR	[37]	80.8	92.5	<b>83.6</b>	63.7	62.7	76.7	88	<b>96.3</b>	83.9	77.8	71.5	83.5
LIG	[8]	46.9	50.3	63.9	66	63.8	58.2	<b>88.7</b>	92.2	<b>89</b>	77	75.2	84.4
P2M	[32]	75.2	83.3	75.5	71.3	67.8	74.6	86.3	92.2	88.1	84.5	82.1	86.6
SAP	[38]	75.6	89.1	72.4	55.3	34.9	65.4	83.4	94.8	61.6	74.5	55.3	73.9
SPSR	[33]	77.7	90.2	82.8	90.3	<b>82.1</b>	84.6	88.1	96	88.1	<b>96.2</b>	<b>85.8</b>	<b>90.9</b>
Labatut <i>et al.</i>	[34]	<b>81.3</b>	<b>93.4</b>	80.1	<b>93.4</b>	79.1	<b>85.5</b>	87.6	96	66.3	94.9	66.5	82.3
Method	Chamfer distance (per-point ave. %) [↓]						Number of components [↓]						
	LR	HR	HRN	HRO	HRNO	Mean	LR	HR	HRN	HRO	HRNO	Mean	
IGR	[37]	0.674	0.322	<b>0.554</b>	7.96	7.72	3.45	6.8	<b>1.2</b>	35.2	44	97.4	36.9
LIG	[8]	0.745	0.581	0.781	7.89	7.8	3.56	<b>1</b>	<b>1</b>	<b>1</b>	1.6	<b>1</b>	1.12
P2M	[32]	0.817	0.473	0.729	1.53	2.13	1.13	<b>1.2</b>	<b>1</b>	<b>1.2</b>	1.4	1.6	1.28
SAP	[38]	0.852	0.32	0.701	3.99	3.93	1.96	73.2	85.6	937	1.8e+03	1.96e+03	971
SPSR	[33]	0.794	0.369	0.572	0.362	<b>0.607</b>	0.541	<b>1.2</b>	1.6	3.6	3.8	20.2	6.08
Labatut <i>et al.</i>	[34]	<b>0.635</b>	<b>0.314</b>	0.608	<b>0.339</b>	0.641	<b>0.507</b>	<b>1</b>	<b>1</b>	<b>1.2</b>	<b>1.2</b>	<b>1</b>	<b>1.08</b>
Method	Number of boundary edges [↓]						Number of non-manifold edges [↓]						
	LR	HR	HRN	HRO	HRNO	Mean	LR	HR	HRN	HRO	HRNO	Mean	
IGR	[37]	0	0	0	0	0	0	0.8	0.8	5.2	4.2	2.2	
LIG	[8]	69	42.8	17.2	0	0	25.8	0	0	0	0	0	
P2M	[32]	0	0	0	0	0	0	0	0	0	0	0	
SAP	[38]	0	0	0	0	449	89.8	0	0	0	0	0	
SPSR	[33]	0	0	0	0	0	0	0	0	0	0	0	
Labatut <i>et al.</i>	[34]	0	0	0	0	0	1	5.8	24.4	3.8	22	11.4	

methods per shape using standard settings. We show the numerical results in Table 5 and visualisations in the supplementary material. The learning-based methods DGNN and POCO benefit from the training on point clouds with the same characteristics as in the test set and reconstruct more truthful surfaces than the optimization-based methods.

Similar to E6, Labatut *et al.* produces the best results among the optimization-based methods.

#### 6.4 Learning- and optimization-based surface reconstruction from real point clouds (E8)

Finally, we reconstruct surfaces from real MVS and range scanning point clouds. Again, for learning-based methods, we use the models trained on synthetic MVS scans from ShapeNet (cf. E4) and we optimize non-learning methods per point cloud. We show the reconstructions in Figure 6. The MVS point cloud from Middlebury (Figure 6a) is contaminated with a large amount of varying noise. SAP is the only learning method which reconstructs a smooth surface without missing details (Figure 6d). However, it suffers from small amounts of topological noise in the form of holes. The optimization-based method P2M provides a visually good reconstruction with few defects (Figure 6i). In Figures 6m and 6y, optimization-based methods handle the additional domain shift to an open scene better compared to learning-based methods. The two traditional methods SPSR and Labatut *et al.* provide the visually best results on average.

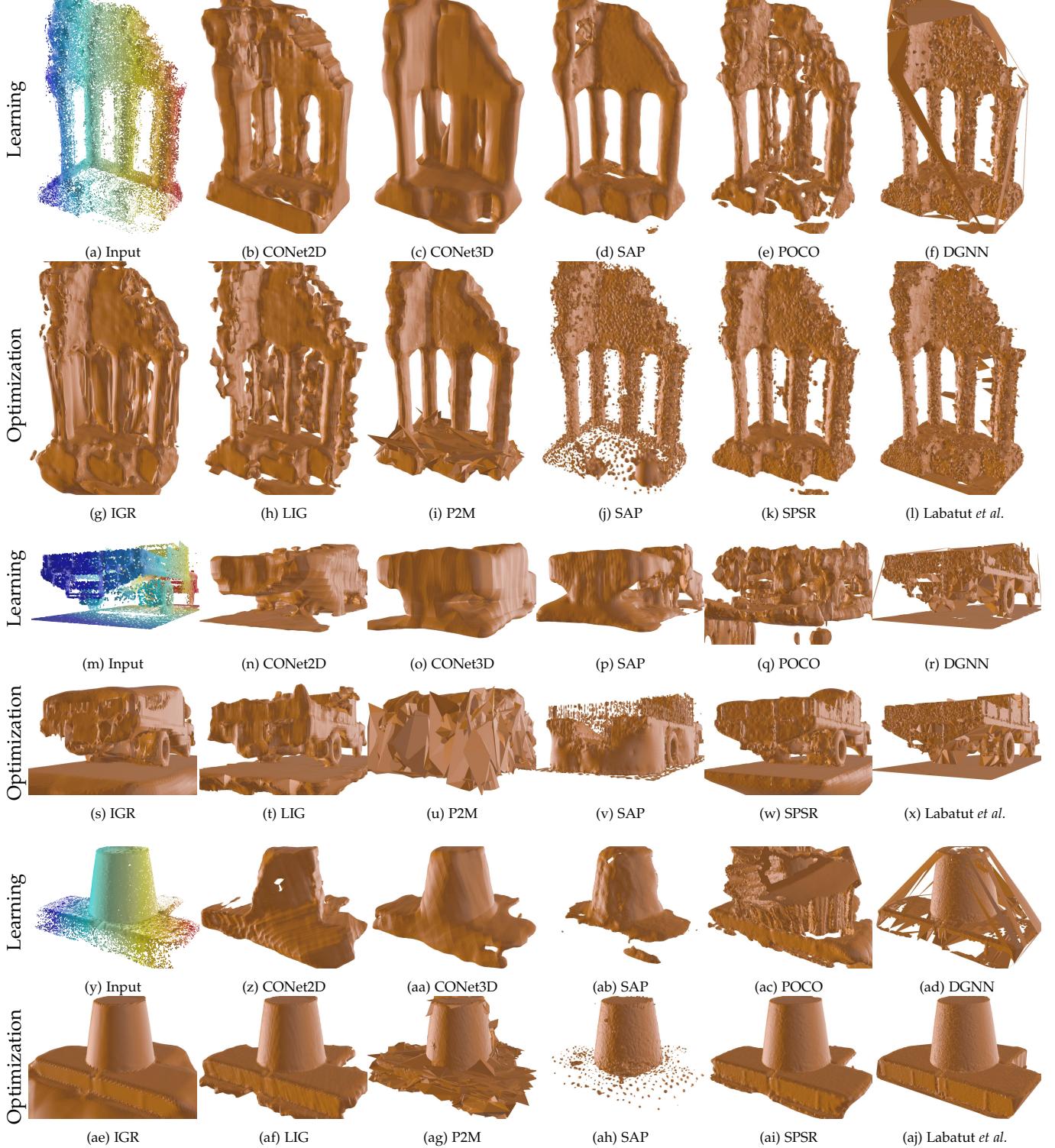
This experiment also shows that our findings on synthetic point clouds coincide with those on real-world point clouds, validating our experimental setup.

#### 6.5 Runtimes

On Table 6, we report detailed runtimes for the methods tested in the learning-based experiments. SAP is the fastest of all reconstruction methods. DGNN also shows fast runtimes, while POCO is slow, due to its extensive use of neighborhood sampling. We also compare runtimes of P2S. We were not able to include this method in experiments E1 to E5 due to its long runtime for training and inference.

#### 6.6 Summary and analysis

In the right circumstances, learning-based methods can produce highly detailed surfaces while remaining robust to noise and missing data. However, this requires training on large sets (30k shapes in our experiments) of sufficiently complex surfaces and associated point clouds. Even if learning methods can generalize to unseen shape categories to some extent, the training and test sets must share the same point cloud characteristics. This suggests that these methods mainly learn priors related to the acquisition characteristics of the input point clouds, and less on the shapes themselves. However, learning-based methods do not produce satisfying results when the training shapes are too simple, or when the point clouds include unknown defects, such as outliers (see Table 7). Mixing traditional and learning-based methods, as in SAP or DGNN, results in higher robustness to domain shifts and leads to short reconstruction times. Except for IGR, novel optimization-based methods are not robust to acquisition defects and they rarely provide better results compared to the two traditional methods SPSR and Labatut *et al.*.



**Figure 6: Learning- and optimization-based reconstructions (E8):** We show reconstructions of *Temple Ring* from Middlebury ((b) to (l)), *Truck* from Tanks And Temples ((n) to (x)) and *scan1* from the DTU dataset ((z) to (aj)). The learning methods (top rows) were trained on synthetic MVS scans from ShapeNet. Optimization-based methods (bottom rows) are optimized per shape using standard settings. The two traditional methods SPSR [33] and Labatut *et al.* [34] provide visually the best results. Their reconstructions are only affected by the heavy noise of the *Temple Ring* MVS point cloud.

**Table 5: Numerical results for learning- vs. optimization-based reconstructions (E7):** Learning- and optimization-based reconstruction of the Berger *et al.* test shapes from synthetic MVS scans. The learning methods were trained on synthetic MVS scans from ShapeNet. Optimization-based methods are optimized per shape using standard settings. BE stands for boundary edges and NME for non-manifold edges.

Method		Vol. IoU [↑]	Normal consist. [↑]	Chamfer dist. [↓]	Components [↓]	BE [↓]	NME [↓]
<i>Learning</i>							
ConvONet2D [6]		65.1	78	1.43	3.6	0	0
ConvONet3D [6]		76.4	87.2	0.887	2.6	0	0
SAP [38]		78.3	89	0.734	5.6	0	0
DGNN [10]		82.9	85.2	0.586	1	0	0.4
POCO [12]		<b>83.9</b>	<b>89.5</b>	<b>0.579</b>	2	0	0
<i>Optimization</i>							
IGR [37]		78.3	83.8	0.775	15.4	0	0.4
LIG [8]		45.7	86.6	0.831	1	65.6	0
P2M [32]		74.5	85	0.768	2	0	0
SAP [38]		71.9	77	0.811	133	0	0
SPSR [33]		77.6	86.4	0.785	8	0	0
Labatut <i>et al.</i> [34]		79.4	80.8	0.671	1	0	9.6

**Table 6: Runtimes of surface reconstruction methods:** Average times (in seconds) for reconstructing one object from a point cloud of 3,000 points. Times are averaged over the ShapeNet test set. GC stand for Graph-cut; SE stands for surface extraction, such as marching cubes or triangle-from-tetrahedron. Note that different variants and implementations of marching cubes are used by different methods, which also influences the runtimes.

Model	Feature extraction	Decoding/GC	SE	Total
ConvONet2D [6]	0.016	0.32	0.17	0.51
ConvONet3D [6]	0.008	0.21	0.17	0.40
SAP [38]	0.022	0.017	0.047	<b>0.088</b>
DGNN [10]	0.11	0.28	0.01	0.39
POCO [12]	0.088	13.72	0.33	15.74
P2S [9]		69.06		11.51 80.57
SPSR [33]				1.25
Labatut <i>et al.</i> [34]	0.1	0.07	0.01	0.18

## 7 CONCLUSION

Surface reconstruction from point clouds is a well studied subject in the field of digital geometry processing. However, constant developments in acquisition techniques and novel ideas for surface reconstruction and analysis bring forward new challenges. In this paper, we survey the field of surface reconstruction from point clouds and benchmark several related methods. We revisit traditional test-of-time approaches for surface reconstruction and detail how they inspired novel approaches. We evaluate traditional and novel optimization and learning-based methods on various tasks and datasets. We show that novel optimization-based methods are not as robust against defects as traditional methods. For in-distribution point clouds with characteristics similar to the ones of the training set, learning methods provide more accurate reconstructions than traditional approaches. However, real-world scenes often include a multitude of different and highly complex objects, and their acquisitions may contain a variety of defects. Most learning methods require shapes of similar complexity in training and test sets and they are not robust to out-of-distribution acquisition defects. These limitations of learning-based methods hinder

the reconstruction of point clouds in the wild. Generating or finding adequate training data that includes a large variety of complex shapes scanned with realistic defects is a difficult task. Future work in learning-based surface reconstruction should focus on training on point clouds with realistic acquisition defects, *e.g.* from common sensors and acquisition settings, or on increasing the methods’ robustness to unseen defects.

## ACKNOWLEDGMENTS

This work was partially funded by the ANR-17-CE23-0003 BIOM grant.

## REFERENCES

- [1] M. Berger, A. Tagliasacchi, L. Seversky, P. Alliez, G. Guennebaud, J. Levine, A. Sharf, and C. Silva, “A survey of surface reconstruction from point clouds,” *Computer Graphics Forum*, 2016.
- [2] F. Cazals and J. Giesen, “Delaunay triangulation based surface reconstruction,” in *Effective computational geometry for curves and surfaces*. Springer, 2006, pp. 231–276.
- [3] C. C. You, S. P. Lim, S. C. Lim, J. San Tan, C. K. Lee, and Y. M. J. Khaw, “A survey on surface reconstruction techniques for structured and unstructured data,” in *2020 IEEE Conference on Open Systems (ICOS)*. IEEE, 2020, pp. 37–42.
- [4] R. M. Bolle and B. C. Vemuri, “On three-dimensional surface reconstruction methods,” *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 13, no. 01, pp. 1–13, 1991.
- [5] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, and B. Lévy, *Polygon mesh processing*. CRC press, 2010.
- [6] S. Peng, M. Niemeyer, L. Mescheder, M. Pollefeys, and A. Geiger, “Convolutional occupancy networks,” in *European Conference on Computer Vision (ECCV)*, 2020, pp. 523–540.
- [7] R. Chabra, J. E. Lenssen, E. Ilg, T. Schmidt, J. Straub, S. Lovegrove, and R. Newcombe, “Deep local shapes: Learning local SDF priors for detailed 3D reconstruction,” in *European Conference on Computer Vision (ECCV)*, 2020, pp. 608–625.
- [8] C. M. Jiang, A. Sud, A. Makadia, J. Huang, M. Nießner, and T. Funkhouser, “Local implicit grid representations for 3D scenes,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 6001–6010.
- [9] P. Erler, S. Ohrhallinger, N. Mitra, and M. Wimmer, “Points2Surf: Learning implicit surfaces from point clouds,” in *European Conference on Computer Vision (ECCV)*, 2020, pp. 108–124.
- [10] R. Sulzer, L. Landrieu, R. Marlet, and B. Vallet, “Scalable surface reconstruction with delaunay-graph neural networks,” *Computer Graphics Forum*, vol. 40, no. 5, pp. 157–167, 2021.

**Table 7: Summary of benchmark results:** We summarise the findings of our benchmark. Each method is rated with one to three stars per attribute, determined by the qualitative and quantitative results of our benchmark.

Robustness to out-of-distribution					Mesh quality			Runtime	
Learning	noise	outliers	density	category	watertightness	manifoldness	compactness	w/o training	
ConvONet2D [6]	*	*	*	*	***	***	*	***	
ConvONet3D [6]	*	*	*	*	***	***	***	***	
SAP [38]	**	**	**	**	***	***	*	***	
DGNN [10]	*	*	*	***	***	**	***	***	
POCO [12]	**	*	*	**	***	***	*	**	
Robustness to					Mesh quality			Runtime	
Optimization	noise	outliers	low density	category	watertightness	manifoldness	compactness	w/ training	
IGR [37]	***	**	**	***	***	**	*	*	
LIG [8]	*	*	*	***	**	***	***	*	
P2M [32]	*	**	*	***	***	***	***	*	
SAP [38]	*	*	*	***	***	***	*	*	
SPSR [33]	***	***	***	***	***	***	*	***	
Labatut <i>et al.</i> [34]	***	***	***	***	***	*	***	***	

- [11] M.-J. Rakotosaona, P. Guerrero, N. Aigerman, N. J. Mitra, and M. Ovsjanikov, “Learning delaunay surface elements for mesh reconstruction,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 22–31.
- [12] A. Boulch and R. Marlet, “Poco: Point convolution for surface reconstruction,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 6302–6314.
- [13] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, “DeepSDF: Learning continuous signed distance functions for shape representation,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 165–174.
- [14] M. Berger, J. A. Levine, L. G. Nonato, G. Taubin, and C. T. Silva, “A benchmark for surface reconstruction,” *ACM Transaction on Graphics*, 2013.
- [15] S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, “A comparison and evaluation of multi-view stereo reconstruction algorithms,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006, pp. 519–528.
- [16] R. Jensen, A. Dahl, G. Vogiatzis, E. Tola, and H. Aanæs, “Large scale multi-view stereopsis evaluation,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 406–413.
- [17] C. Strecha, W. von Hansen, L. V. Gool, P. Fua, and U. Thoennessen, “On benchmarking camera calibration and multi-view stereo for high resolution imagery,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, 2008, pp. 1–8.
- [18] T. Schöps, J. L. Schönberger, S. Galliani, T. Sattler, K. Schindler, M. Pollefeys, and A. Geiger, “A multi-view stereo benchmark with high-resolution images and multi-camera videos,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 3260–3269.
- [19] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun, “Tanks and Temples,” *ACM Transactions on Graphics (TOG)*, 2017.
- [20] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, “Surface reconstruction from unorganized points,” in *Proceedings of the 19th annual conference on computer graphics and interactive techniques*, 1992, pp. 71–78.
- [21] B. O’neill, *Elementary differential geometry*. Elsevier, 2006.
- [22] L. Kettner, “Using generic programming for designing a data structure for polyhedral surfaces,” *Computational Geometry*, vol. 13, no. 1, pp. 65–90, 1999.
- [23] M. Mäntylä, *An introduction to solid modeling*. Computer Science Press, Inc., 1987.
- [24] J.-D. Boissonnat and S. Oudot, “Provably good sampling and meshing of surfaces,” *Graphical Models*, vol. 67, no. 5, pp. 405–451, 2005.
- [25] W. E. Lorensen and H. E. Cline, “Marching cubes: A high resolution 3d surface construction algorithm,” *SIGGRAPH Comput. Graph.*, vol. 21, no. 4, p. 163–169, Aug. 1987. [Online]. Available: <https://doi.org/10.1145/37402.37422>
- [26] M. Vetsch, S. Lombardi, M. Pollefeys, and M. R. Oswald, “Neuralmeshing: Differentiable meshing of implicit neural representations,” in *Pattern Recognition*, B. Andres, F. Bernard, D. Cremers, S. Frintrop, B. Goldlücke, and I. Ihrke, Eds. Cham: Springer International Publishing, 2022, pp. 317–333.
- [27] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin, “The ball-pivoting algorithm for surface reconstruction,” *IEEE transactions on visualization and computer graphics*, vol. 5, no. 4, pp. 349–359, 1999.
- [28] A. Sharf, T. Lewiner, A. Shamir, L. Kobbelt, and D. Cohen-Or, “Competing fronts for coarse-to-fine surface reconstruction,” in *Computer Graphics Forum*, vol. 25. Wiley Online Library, 2006, pp. 389–398.
- [29] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry, “AtlasNet: A papier-mâché approach to learning 3D surface generation,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 658–666.
- [30] M. Liu, X. Zhang, and H. Su, “Meshing point clouds with predicted intrinsic-extrinsic ratio guidance,” in *European Conference on Computer Vision*. Springer, 2020, pp. 68–84.
- [31] N. Sharp and M. Ovsjanikov, “pointrinet: Learned triangulation of 3d point sets,” in *European Conference on Computer Vision (ECCV)*, 2020, pp. 762–778.
- [32] R. Hanocka, G. Metzer, R. Giryes, and D. Cohen-Or, “Point2Mesh: A self-prior for deformable meshes,” *ACM Transaction on Graphics*, 2020.
- [33] M. Kazhdan and H. Hoppe, “Screened Poisson surface reconstruction,” *ACM Transaction on Graphics*, 2013.
- [34] P. Labatut, J. P. Pons, and R. Keriven, “Robust and efficient surface reconstruction from range data,” *Computer Graphics Forum (CGF)*, 2009.
- [35] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, “Occupancy networks: Learning 3D reconstruction in function space,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4460–4470.
- [36] Z. Chen and H. Zhang, “Learning implicit fields for generative shape modeling,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 5939–5948.
- [37] A. Gropp, L. Yariv, N. Haim, M. Atzmon, and Y. Lipman, “Implicit geometric regularization for learning shapes,” in *Proceedings of Machine Learning and Systems 2020*. JMLR.org, 2020, pp. 3569–3579.
- [38] S. Peng, C. M. Jiang, Y. Liao, M. Niemeyer, M. Pollefeys, and A. Geiger, “Shape as points: A differentiable poisson solver,” in *Conference on Neural Information Processing Systems (NeurIPS)*, 2021, pp. 13 032–13 044.
- [39] S. Petitjean and E. Boyer, “Regular and non-regular point sets: Properties and reconstruction,” *Computational Geometry*, vol. 19, no. 2–3, pp. 101–126, 2001.
- [40] J. Digne, “An analysis and implementation of a parallel ball pivoting algorithm,” *Image Processing On Line*, vol. 4, pp. 149–168, 2014.
- [41] J.-D. Boissonnat, “Geometric structures for 3d shape representation,” *ACM Transactions on Graphics*, vol. 3, no. 4, 1984.

- [42] D. Attali, J.-D. Boissonnat, and A. Lieutier, "Complexity of the delaunay triangulation of points on surfaces the smooth case," in *Proceedings of the nineteenth annual symposium on Computational Geometry*, 2003, pp. 201–210.
- [43] F. Bernardini and C. L. Bajaj, "Sampling and reconstructing manifolds using alpha-shapes," *Proc. 9th Canad. Conf. Comput. Geom.*, 1997.
- [44] N. Amenta, M. Bern, and D. Eppstein, "The crust and the  $\beta$ -skeleton: Combinatorial curve reconstruction," *Graphical models and image processing*, vol. 60, no. 2, pp. 125–135, 1998.
- [45] R. Kolluri, J. R. Shewchuk, and J. F. O'Brien, "Spectral surface reconstruction from noisy point clouds," in *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, 2004, pp. 11–21.
- [46] S. N. Sinha, P. Mordohai, and M. Pollefeys, "Multi-view stereo via graph cuts on the dual of an adaptive tetrahedral mesh," in *2007 IEEE 11th international conference on computer vision*. IEEE, 2007, pp. 1–8.
- [47] V. H. Hiep, R. Keriven, P. Labatut, and J.-P. Pons, "Towards high-resolution large-scale multi-view stereo," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 1430–1437.
- [48] Y. Zhou, S. Shen, and Z. Hu, "Detail preserved surface reconstruction from point cloud," *Sensors*, 2019.
- [49] C. Mostegel, R. Pretenthaler, F. Fraundorfer, and H. Bischof, "Scalable surface reconstruction from point clouds with extreme scale and density diversity," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 904–913.
- [50] H. H. Vu, P. Labatut, J. P. Pons, and R. Keriven, "High accuracy and visibility-consistent dense multiview stereo," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2012.
- [51] M. Jancosek and T. Pajdla, "Multi-view reconstruction preserving weakly-supported surfaces," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 3121–3128.
- [52] ——, "Exploiting visibility information in surface reconstruction to preserve weakly supported surfaces," *International Scholarly Research Notices*, 2014.
- [53] L. Caraffa, M. Brédif, and B. Vallet, "3D watertight mesh generation with uncertainties from ubiquitous data," in *Asian Conference on Computer Vision (ACCV)*, 2017, pp. 377–391.
- [54] Y. Luo, Z. Mi, and W. Tao, "Deepdt: Learning geometry from delaunay triangulation for surface reconstruction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, 2021, pp. 2277–2285.
- [55] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson surface reconstruction," in *Proceedings of the fourth Eurographics symposium on Geometry processing*. Eurographics Association, 2006, pp. 61–70.
- [56] M. Kazhdan, M. Chuang, S. Rusinkiewicz, and H. Hoppe, "Poisson Surface Reconstruction with Envelope Constraints," *Computer Graphics Forum*, vol. 39, no. 5, pp. 173–182, 2020.
- [57] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "ShapeNet: an information-rich 3D model repository," Stanford University, Princeton University, Toyota Technological Institute at Chicago, Tech. Rep., 2015.
- [58] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.
- [59] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3d u-net: learning dense volumetric segmentation from sparse annotation," in *International conference on medical image computing and computer-assisted intervention*. Springer, 2016, pp. 424–432.
- [60] J. Huang, Y. Zhou, and L. Guibas, "Manifoldplus: A robust and scalable watertight manifold surface generation method for triangle soups," *arXiv preprint arXiv:2005.11621*, 2020.
- [61] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, "3D-R2N2: A unified approach for single and multi-view 3D object reconstruction," in *European Conference on Computer Vision (ECCV)*, 2016, pp. 628–644.
- [62] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," *ACM SIGGRAPH Computer Graphics*, vol. 21, no. 4, 1987.
- [63] F. Cazals and M. Pouget, "Estimating differential quantities using polynomial fitting of osculating jets," *Computer Aided Geometric Design*, 2005.
- [64] A. Boulch, "Convpoint: Continuous convolutions for point cloud processing," *Computers & Graphics*, 2020.



**Raphael Sulzer** is a postdoctoral researcher in the TITANE team at INRIA Sophia-Antipolis. He received his PhD in geometry processing and deep learning from University Gustave Eiffel in 2022. During his PhD he was affiliated with the LASTIG lab at IGN, the French mapping agency, and the IMAGINE lab at École des Ponts Paris-Tech. He is committed to open and reproducible research that aims to solve real-world problems, mainly in the areas of 3D scene understanding and 3D reconstruction. <https://raphaelsulzer.de>



**Loic Landrieu** received a PhD in machine learning from ENS Paris in 2016. He is now a research scientist at IGN, the French mapping agency, working on 3D point clouds and satellite time series analysis. He is the main investigator of the ANR Ready3D on dynamic 3D analysis, co-chair of the ISPRS working group on temporal data understanding, co-lead of the GRSS group on image analysis, and was program chair of the XXIV ISPRS Congress. Committed to open and reproducible research, he has participated in numerous open-source projects and released several large-scale benchmarks. <https://loiclandrieu.com>



**Renaud Marlet** is a Senior Researcher at École des Ponts ParisTech (ENPC) and a Principal Scientist at valeo.ai, France. He has held positions both in academia (researcher at Inria) and in the software industry (expert at Simulog, deputy CTO of Trusted Logic). He was the head of the IMAGINE group at LIGM/ENPC (2010–2019). He is currently interested in scene understanding and semantized 3D reconstruction, with applications to robotics, autonomous driving and civil engineering.



**Bruno Vallet** is a senior researcher at IGN, the French national mapping agency. He is the head of the ACTE research team of the Lastig lab (ENSG/UGE) which specializes on image/Lidar/Radar data acquisition and processing. His research interests include geographic information science, computer vision, teledetection, lasergrammetry, change detection, 3D+T city modeling.

# Supplementary Material: A Survey and Benchmark of Automatic Surface Reconstruction from Point Clouds

Raphael Sulzer, Loic Landrieu, Renaud Marlet,  
and Bruno Vallet

arXiv:2301.13656v1 [cs.CV] 31 Jan 2023

In this supplementary document, we provide additional information about the datasets we used in our benchmark and additional results. All the datasets and the evaluation code for our benchmark are available on GitHub: <https://github.com/raphaelsulzer/dsr-benchmark>

## SM.1 DATASETS

### SM.1.1 Berger *et al.*

We use the range scanning procedure from the surface reconstruction benchmark of Berger *et al.* [?]. To this end, we modified their provided code to export the camera positions of the scanning process along with the point cloud. Our modified version of the code is available on Github: <https://github.com/raphaelsulzer/reconbench-CMake>. We choose five different scanner settings, detailed in Table SM.1

and visible in the first row of Figure SM.2 to scan each test shape shown in Figure SM.1a.

### SM.1.2 ModelNet10 and ShapeNet

We show example shapes for all classes of ShapeNet in Figure SM.1b and example shapes for ModelNet for the 6 out of 10 classes which are not represented in ShapeNet in Figure SM.1c.

## SM.2 BENCHMARK SETUP

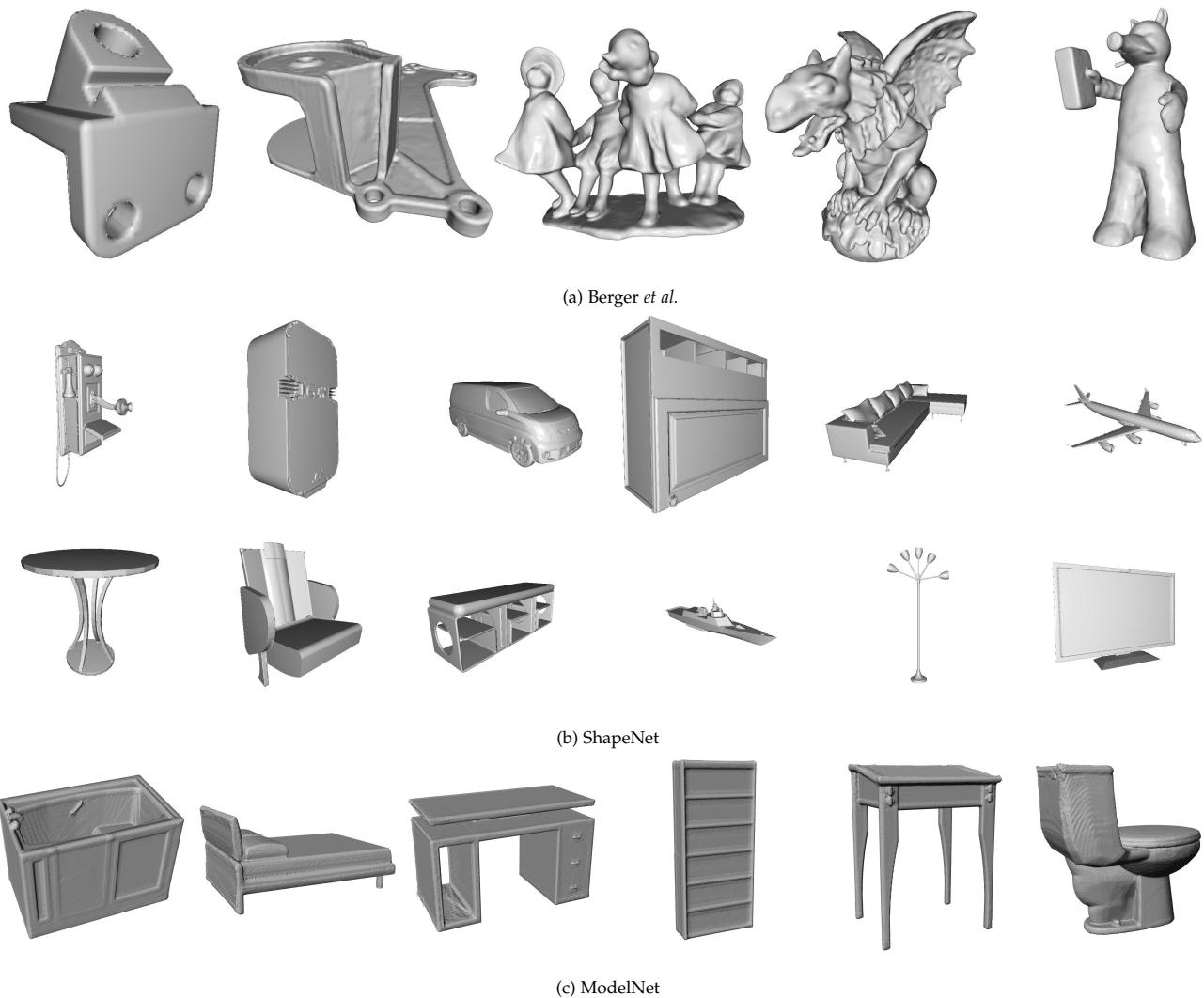
We show a detailed overview of our benchmark setup on Table SM.2.

## SM.3 ADDITIONAL RESULTS

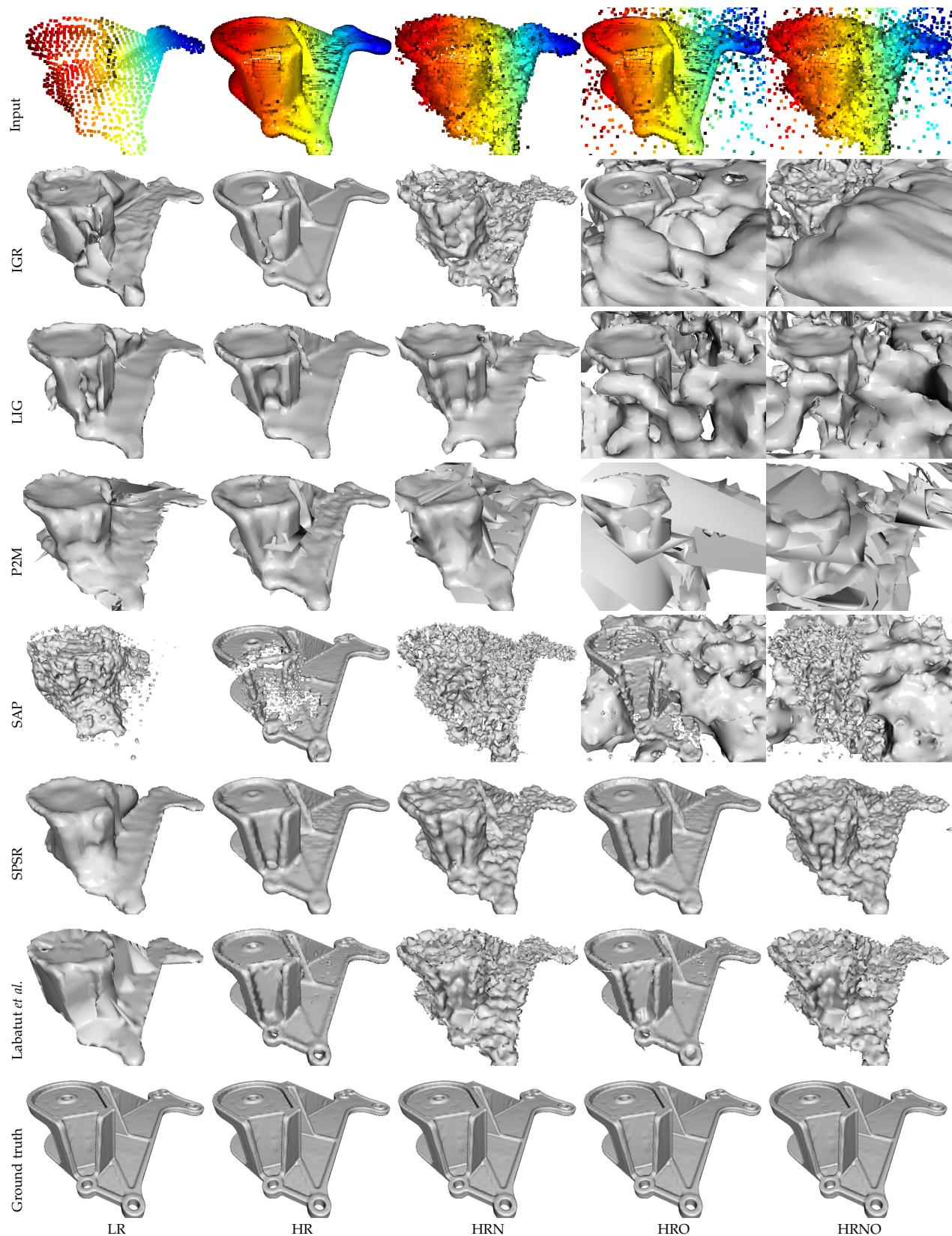
We show qualitative results of Experiment 6 in Figure SM.2.

**Table SM.1: Scanning configurations for Berger *et al.* benchmark:** We show the five different scanner configurations used in our modified version of the Berger et al.'s scanning procedure. We use the resulting scans to evaluate object-level reconstruction with varying point-cloud defects and for training data generation. For the low resolution (LR) scans the scanning process results in 1000 to 3000 points per shape, and for the high resolution (HR), the scanning process yields around 10 000 to 30 000 points.

	Low res. (LR)	High res. (HR)	HR + noise (HRN)	HR + outliers (HRO)	HR + noise + outliers (HRNO)
Camera resolution x, y	50, 50	100, 100	100, 100	100, 100	100, 100
Scanner positions	5	10	10	10	10
Min/max range	70/300	70/300	70/300	70/300	70/300
Additive noise	0	0	0.5	0	0.5
Outliers (%)	0	0	0	0.1	0.1



**Figure SM.1: Ground truth shapes of the benchmark datasets:** We show an example shape of each class of ModelNet in (c) and of ShapeNet in (b) and the five shapes of Berger *et al.* in (a).



**Figure SM.2: Optimization-based experiments:** In each column we show the results of different methods of one of the five learning-based experiments.

**Table SM.2: Benchmark setup:** We show an overview of our experimental setup. In E1 to E4, we train surface reconstruction methods on noisy point clouds of ShapeNet. In E1, we test on the ShapeNet test set. In E2, we test on ShapeNet, but from denser point clouds with noise and outliers. In E3, we test on the simpler ModelNet objects with the same sampling as in E1. In E4, we test on five Berger *et al.* shapes with the same sampling as in E1. In E5, we train the methods on the simpler ModelNet dataset and test on ShapeNet, both with the same sampling as in E1. In E6, we test optimization-based methods on synthetic range scans of the Berger *et al.* dataset. And finally, in E7, we compare learning- and optimization-based methods on the same dataset (synthetic MVS scans of the Berger *et al.* dataset).

Experiment	Name	# shapes	<i>Training set</i>				Name	# shapes	<i>Test set</i>			
			complexity	# points	$\sigma$ noise	% outliers			complexity	# points	$\sigma$ noise	% outliers
1	ShapeNet	30,661	**	3,000	0.005	0	ShapeNet	1,300	**	3,000	0.005	0
2	ShapeNet	30,661	**	3,000	0.005	0	ShapeNet	1,300	**	10k	0.005	10
3	ShapeNet	30,661	**	3,000	0.005	0	ModelNet	506	*	3,000	0.005	0
4	ShapeNet	30,661	**	3,000	0.005	0	Berger <i>et al.</i>	5	**	3,000	0.005	0
5	ModelNet	3,979	*	3,000	0.005	0	ShapeNet	1,300	**	3,000	0.005	0
6			—				Berger <i>et al.</i>	5	**			see Table SM.1
7	ShapeNet	30,661	**	3,000	0.005	0	Berger <i>et al.</i>	5	**	3,000	0.005	0