

Surface Reconstruction From Point Clouds: A Survey and a Benchmark

ZhangJin Huang , Yuxin Wen , ZiHao Wang, Jinjuan Ren , and Kui Jia

(Survey Paper)

Abstract—Reconstruction of a continuous surface of two-dimensional manifold from its raw, discrete point cloud observation is a long-standing problem in computer vision and graphics research. The problem is technically ill-posed, and becomes more difficult considering that various sensing imperfections would appear in the point clouds obtained by practical depth scanning. In literature, a rich set of methods has been proposed, and reviews of existing methods are also provided. However, existing reviews are short of thorough investigations on a common benchmark. The present paper aims to review and benchmark existing methods in the new era of deep learning surface reconstruction. To this end, we contribute a large-scale benchmarking dataset consisting of both synthetic and real-scanned data; the benchmark includes object- and scene-level surfaces and takes into account various sensing imperfections that are commonly encountered in practical depth scanning. We conduct thorough empirical studies by comparing existing methods on the constructed benchmark, and pay special attention on robustness of existing methods against various scanning imperfections; we also study how different methods generalize in terms of reconstructing complex surface shapes. Our studies help identify the best conditions under which different methods work, and suggest some empirical findings. For example, while deep learning methods are increasingly popular in the research community, our systematic studies suggest that, surprisingly, a few classical methods perform even better in terms of both robustness and generalization; our studies also suggest that the practical challenges of misalignment of point sets from multi-view scanning, missing of surface points, and point outliers remain unsolved by all the existing surface reconstruction methods. We expect that the benchmark and our studies would be valuable both for practitioners and as a guidance for new innovations in future research.

Index Terms—Surface reconstruction, surface modeling, point cloud, benchmarking dataset, literature survey, deep learning.

Manuscript received 18 April 2022; revised 6 April 2023; accepted 8 July 2024. Date of publication 16 July 2024; date of current version 5 November 2024. This work was supported by the Guangdong Provincial Key Laboratory of Human Digital Twin under Grant 2022B1212010004. Recommended for acceptance by K.M. Lee. (*ZhangJin Huang and Yuxin Wen contributed equally to this work.*) (*Corresponding author: Kui Jia.*)

ZhangJin Huang, Yuxin Wen, ZiHao Wang, and Kui Jia are with the School of Electronic and Information Engineering, South China University of Technology, Guangzhou 510641, China (e-mail: eehuangzhangjin@mail.scut.edu.cn; wen.yuxin@mail.scut.edu.cn; eezihaowang@mail.scut.edu.cn; kuijia@gmail.com).

Jinjuan Ren is with the University of Macau, Macau 314100, China (e-mail: jinjuanren@um.edu.mo).

We make the benchmark publicly accessible at <https://Gorilla-Lab-SCUT.github.io/SurfaceReconstructionBenchmark>.

This article has supplementary downloadable material available at <https://doi.org/10.1109/TPAMI.2024.3429209>, provided by the authors.

Digital Object Identifier 10.1109/TPAMI.2024.3429209

I. INTRODUCTION

MODELING and reconstruction of object or scene surfaces is a fundamental problem in computer vision and graphics research. Its applications range from virtual/augmented reality, computer animation, to computer-aided design and robotics. Given that the mathematical nature of a surface shape is a continuous 2D manifold embedded in the 3D Euclidean space, different approximations are usually adopted when capturing, transmitting, and storing surface shapes, where the prominent examples include point clouds, polygon meshes, and quantized volumes. In this work, we are particularly interested in reconstructing a continuous surface from its discrete approximation of point cloud, since many depth sensors (e.g., those based on multi-view stereo, structured light, or time-of-flight measurements) produce point clouds (or equivalently, the depth maps) as their original forms of data acquisition, and surface reconstructions from the obtained point clouds are subsequently demanded for various downstream applications.

The problem is technically ill-posed — infinitely many solutions of the underlying, continuous surface may exist given an observed set of discrete points. The challenges become even severer considering that various sensing imperfections would appear during the data acquisition process; the captured point clouds could be noisy and distributed in a non-uniform manner, and they could contain outliers and/or cover less on some surface areas; when point clouds are captured at multiple views, they could be subject to less accurate alignments. All these issues pose the classical problem of surface reconstruction from point clouds as a long-standing challenge that draws continuous efforts from the research community.

In literature, a rich set of methods has been proposed for the focused studies; depending on the types of data imperfection they assume, these methods leverage various priors of surface geometry to combat the otherwise ill-posed problem. While comprehensive reviews of these methods are given in [1], [2], [3], [4], these reviews are short of investigations and analyses on a common benchmark that could distinguish existing methods when they cope with the aforementioned data imperfections. In the meanwhile, the field has witnessed a recent surge of deep learning surface reconstruction, where models of deep networks are learned and employed to either decode surface shapes from point clouds explicitly [5], [6], [7], or generate implicit fields whose zero-level iso-surfaces can be extracted as the results

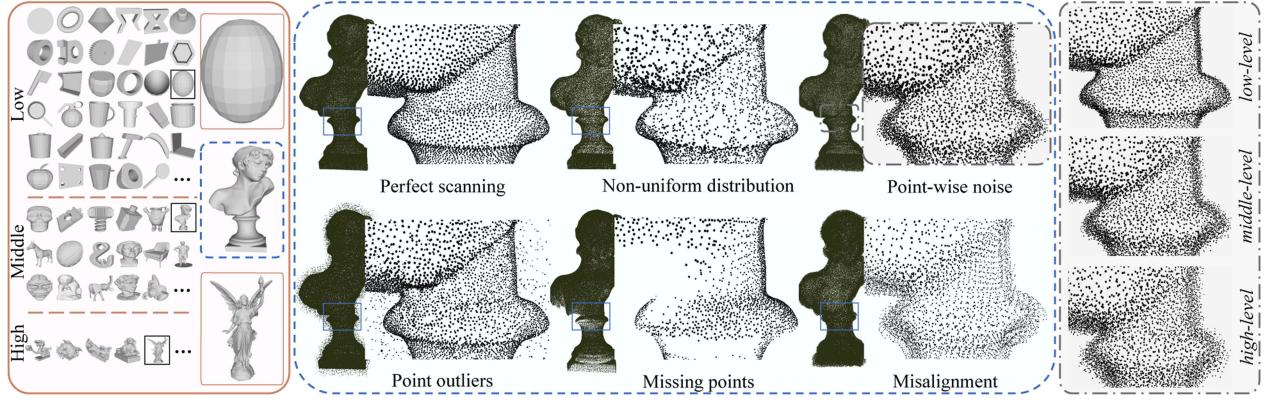


Fig. 1. An illustration of different surface complexities and scanning challenges included in our contributed benchmark. From left to right: example object surfaces of low, middle, and high complexities, the five challenges possibly encountered in practical surface scanning, and examples of different severity levels for the challenge of noisy scanning.

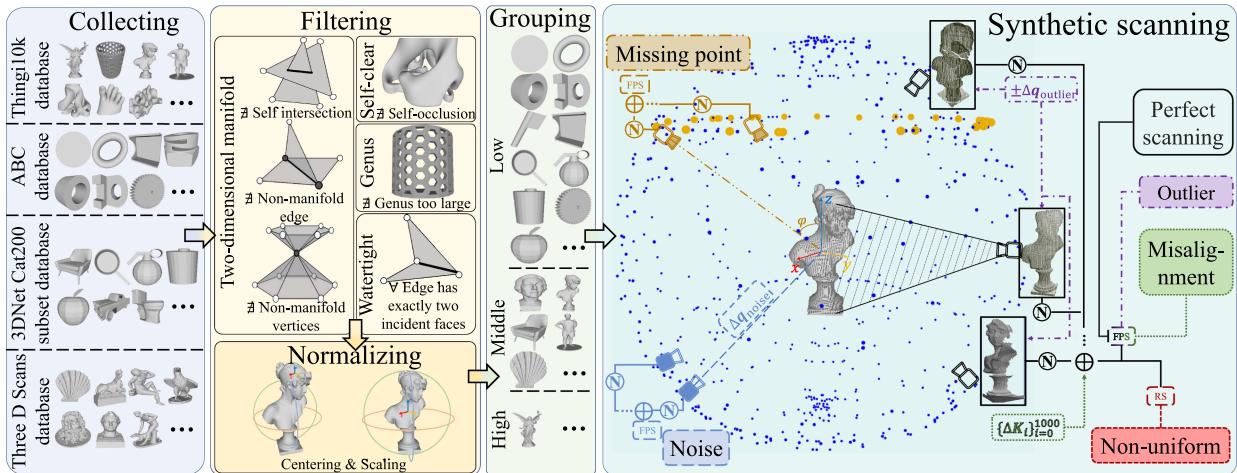


Fig. 2. The pipeline of constructing our synthetic object-level dataset (cf. Section IV-A for the details). We first collect CAD models of object surface from four repositories, as shown in the leftmost of the figure. We then filter the collected surfaces by dropping those failing to meet our requirements, and normalize the remaining ones, as shown in the middle of the figure. We further organize the normalized object surfaces into three groups of low, middle, and high complexities at a ratio of around 6 : 3 : 1, based on the criterion of (algebraic) surface complexity. We finally perform synthetic scanning as shown in the right of the figure, where different challenges possibly encountered in practical scanning are simulated. Blue spots indicate the scanning positions on viewing spheres; after estimating oriented normals (shown as N in the figure), we make registration between different viewpoints of scanning and get the final scanned point cloud, where farthest point sampling (FPS) is used for sampling a fixed number of points per point cloud (except otherwise mentioned cases in which random sampling (RS) is used).

of surface reconstruction [8], [9], [10]. It is thus desirable to benchmark both the classical and the more recent, deep learning solutions in order to understand their respective strengths and limitations; such investigations would be no doubt valuable for use of the appropriate methods by practitioners, and also as a guidance to new innovations in future research.

The present paper aims to provide a comprehensive review and benchmark existing methods in the new era of deep learning surface reconstruction. We organize our review by categorizing existing methods according to what priors of surface geometry they have used to regularize their reconstructions, where we include the more recent priors of deep models and deep learning, in addition to the classical, optimization-based ones. One of our key contributions is a large-scale benchmarking dataset consisting of both synthetic and real-scanned data (cf. Figs. 2, 3, and 5 for illustrations on how the benchmark is constructed).

The benchmark includes object- and scene-level surfaces and takes into account various sensing imperfections that are commonly contained in the point clouds obtained by practical 3D scanning, such as point-wise noise, non-uniform distribution of surface points, point outliers, missing of surface points, and misalignment of point sets from multi-view scanning; Fig. 1 gives an illustration of these scanning imperfections. We conduct thorough empirical studies on the constructed benchmark, by comparing existing methods in terms of their capabilities to reconstruct surfaces from observed point clouds. We pay special attention on robustness of existing methods against various scanning imperfections; we also study how different methods generalize in terms of reconstructing complex surface shapes. Our thorough studies help identify the strengths and limitations of existing methods from multiple perspectives. We summarize a few important findings as follows.

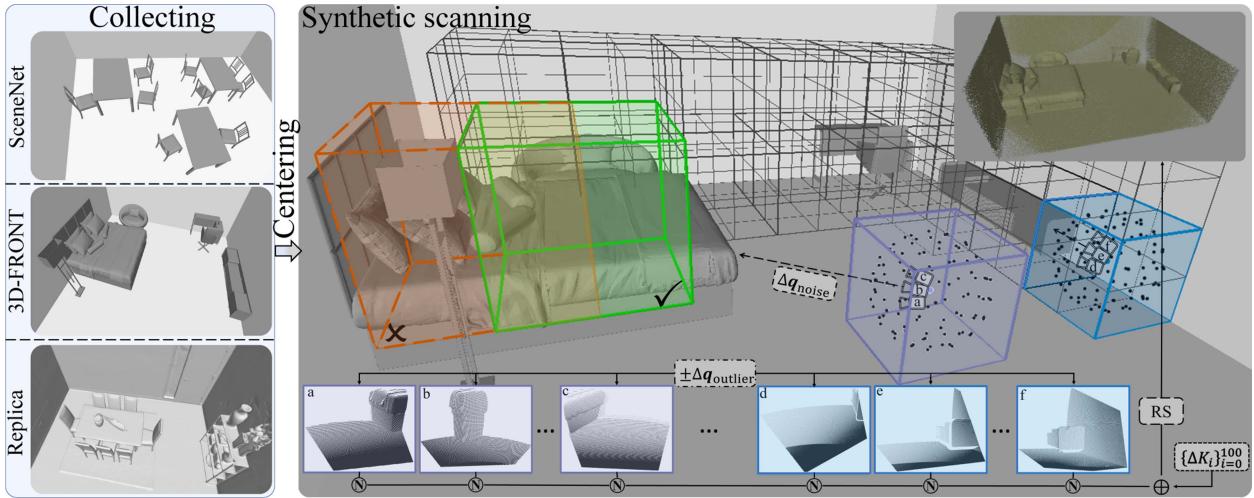


Fig. 3. The pipeline of constructing our synthetic scene-level dataset (cf. Section IV-B for the details). We collect CAD surface models of indoor scenes from three datasets, as shown in the left of the figure, and normalize them by centering at the origin in the 3D space. We partition the 3D space into overlapped cubes, as shown in the right of the figure; some of the cubes contain certain patches of the scene surface, while others are empty. The camera is positioned at centers of those empty cubes (e.g., the purple and blue cubes); for each positioned camera, we randomly sample 100 viewpoints for the scanning. We include all the five challenges of practical scanning into each scanning of the scene surface, whose individual implementations are similar to those for synthetic object scanning.

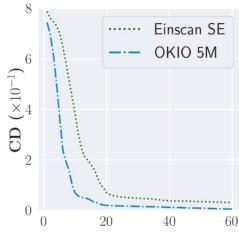


Fig. 4. The relation between Chamfer Distance (CD) and the number of scanning shots. The ground truth for CD is a 100-shot scan of OKIO 5 M.



Fig. 5. An illustration of the objects used in our real-scanned dataset. The objects are organized by the types of material, where “a” shows the objects made of plastic, “b” for ceramic, “c” for cloth, and “d” for metal.

- While many challenges of surface reconstruction from point clouds can be more or less tackled by existing methods, those of *misalignment*, *missing points*, and *outliers* have been less addressed and remain unsolved.

- Deep learning solutions have shown great promise recently for surface modeling and reconstruction; however, our systematic studies suggest that they struggle in generalizing to reconstruction of complex shapes. It is surprising that some classical methods perform even better in terms of both generalization and robustness.
- The use of surface normals is a key to the success of surface reconstruction from raw, observed point clouds, even when the normals are estimated less accurately; in many cases, the result improves as long as the interior/exterior of the surface can be identified in the 3D space.
- There exist inconsistencies between different evaluation metrics, and in many cases, good quantitative results are not always concordant with the visually pleasant ones. This suggests that more foundation studies are demanded to better benchmark different methods and advance the field.

A. Related Works

In this section, we give a summary of existing literatures on surface reconstruction from either point clouds or other observations. We also summarize existing datasets and benchmarks that have served for advancing the field.

Surface Reconstruction from Point Clouds There exists a rich set of existing methods studying surface reconstruction from point clouds. These methods are reviewed in [1], [2], [3], [4], [11]. Earlier reviews of [1] and [2] organize existing methods according to what functions of surface representation they use, e.g., implicit or explicit functions. More recently, existing methods are categorized in [4] based on the difference of used techniques, including interpolation and approximation techniques, learning-based techniques, and soft computing techniques. A most recent work after us [11] groups methods into surface- and volume-based categories. Our organization of review in the present paper is more similar to that in [3], which also organizes

existing methods based on the used geometry priors. Compared with [3], our review is more comprehensive and includes the recent methods of deep learning surface reconstruction. In addition, we also compare existing methods empirically on a benchmark, which helps identify the respective strengths and limitations of existing methods.

More General Surface Modeling and Reconstruction Surface reconstruction can be achieved from other raw observations as well, such as single- or multi-view images, motion, and/or illumination and shading. Literature reviews on the traditional methods of multi-view image reconstruction are provided in [12], [13], and [14]. A more recent survey is given in [15] on multi-view image reconstruction with deep learning. Fahim et al. [16] focus on a more challenging setting of deep learning surface reconstruction from as few as a single image. Zhu et al. [17] provide a more comprehensive survey of 3D modeling methods, including multi-view 3D reconstruction, structure from motion, and shape from shading. Different from the above reviews, the present work focuses on surface reconstruction from raw, observed point clouds, considering that depth sensors are increasingly popularly deployed in either portable or fixed devices.

Datasets and Benchmarks Existing datasets that support surface reconstruction studies are based on synthetic or real-scanned data; they may include object- and/or scene-level surfaces. For synthetic datasets, surface meshes are usually provided from which point clouds can be sampled. For example, the ShapeNet [18] and ModelNet [19] are two commonly used synthetic datasets consisting of simple, object-level shapes. More complex synthetic object surfaces are provided in the datasets of 3DNet [20], ABC [21], Thingi10k [22], and Three D Scans [23]. The datasets of SceneNet [24] and 3D-FRONT [25] provides synthetic, scene-level surfaces. In the meanwhile, there exist datasets of real-scanned, object-level surfaces [26], [27], [28] and those of real-scanned, scene-level surfaces [29], [30]; however, due to the lack of high-precision scanning, their reconstruction ground truths are usually obtained by appropriate surface reconstruction algorithms, which jeopardizes their roles for benchmarking different methods. Most of the above datasets do not consider sensing imperfections that may appear in practically scanned point clouds, except for [31] that uses virtual scanning to simulate point cloud imperfections; however, the dataset [31] is relatively small, with only eight instances of object surfaces. Subsequent work [11] benchmarks methods only on existing object-level dataset with its synthetic scanning approaches. In contrast, our contributed benchmarking dataset is more comprehensive, including both synthetic and real-scanned data, and covering both object- and scene-level surfaces; we intentionally inject various sensing imperfections into point cloud data of the dataset, including *point-wise noise*, *non-uniform distribution of surface points*, *point outliers*, *missing of surface points*, and *misalignment among point sets from multi-view scanning*. We expect our benchmark would facilitate more thorough studies in future research.

B. Contributions

As stated in Section I, with the surge of deep learning surface reconstruction, the present paper aims to provide a

comprehensive review of exiting methods in the new era, and study their respective advantages and disadvantages when reconstructing object- or scene-level surfaces from raw, observed point clouds. To this end, we contribute a large-scale benchmark consisting of both synthetic and real-scanned data. We use the constructed benchmark for systematic studies of existing methods, focusing on the robustness of these methods against various data imperfections, and also on how existing methods generalize in terms of reconstructing complex surfaces. We summarize our key contributions as follows.

- We provide a comprehensive review of existing surface reconstruction methods, by bridging together the classical, optimization-based methods with the more recent, deep learning-based ones, where we categorize these methods according to what priors of surface geometry they have used to regularize their solutions.
- We contribute a large-scale benchmarking dataset consisting of both synthetic and real-scanned data. The point cloud data in the benchmark have various sensing imperfections that are commonly encountered in practical 3D scanning processes; these imperfections are intentionally included to benchmark the robustness of existing methods.
- We compare existing methods by conducting thorough empirical studies on the constructed benchmark. Our studies help identify the strengths and limitations of existing methods, which are valuable both for choices of appropriate methods by practitioners and for guiding the directions of new innovations in future research.

C. Paper Organization

The paper is organized as follows. Section II gives the formal definition of our studied problem. Section III organizes and reviews existing methods based on priors of surface geometry that they have used to regularize the reconstructions. We present our contributed large-scale benchmark in Section IV, where we give details about how we construct the benchmark and also the benchmark statistics; we make the benchmark, our construction manner of the benchmark, and also implementation codes of representative methods publicly accessible at <https://Gorilla-Lab-SCUT.github.io/SurfaceReconstructionBenchmark>. Experimental setups of our empirical studies are given in Section V, before results, analyses, and important insights are presented in Section VI. We finally draw the paper conclusion in Section VII.

II. PROBLEM STATEMENT

Consider a discrete point set \mathcal{P} that may be obtained by scanning an object or scene surface using some 3D sensing devices; each $p \in \mathbb{R}^3$ of its contained points collects the coordinates in the Euclidean space. Our goal of interest is to recover its underlying, continuous surface \mathcal{S}^* from which the points $\{p \in \mathcal{P}\}$ are practically observed. Given that recovering the continuous \mathcal{S}^* from the discrete \mathcal{P} is an ill-posed problem, an appropriate regularization must be imposed in order to recover a geometry-aware approximation \mathcal{S} , e.g., a smooth and/or fair surface [32]. This formally amounts to solving the following

regularized optimization

$$\min_{\mathcal{S}} L(\mathcal{S}; \mathcal{P}) + \lambda R(\mathcal{S}), \quad (1)$$

where L is a loss term for data fidelity to the observed \mathcal{P} , R is a regularizer that constrains the solution with a certain prior of surface geometry, and λ is a scalar penalty.¹ Note that the objective (1) is only in an abstract form, since it is difficult to define both L and R directly on \mathcal{S} . In practice, one may represent a surface either explicitly as a parametric mapping $\mathbf{f} : \Omega^2 \rightarrow \mathcal{S}$, where $\Omega^2 \subset \mathbb{R}^2$ denotes the 2D domain and $\mathcal{S} = \{\mathbf{f}(\mathbf{x}) \in \mathbb{R}^3\}$ with $\mathbf{x} \in \Omega^2$, or implicitly as the zero-level set of an implicit function $F : \mathbb{R}^3 \rightarrow \mathbb{R}$, i.e., $\mathcal{S} = \{\mathbf{q} \in \mathbb{R}^3 | F(\mathbf{q}) = 0\}$. Correspondingly, one can instantiate the objective (1) as the following one that pursues \mathcal{S} by optimizing an explicit mapping

$$\min_{\mathbf{f} \in \mathcal{H}_f} L^{\text{exp}}(\mathbf{f}; \mathcal{P}) + \lambda R^{\text{exp}}(\mathbf{f}), \quad (3)$$

where L^{exp} and R^{exp} are respectively the instantiated loss function and regularizer, and \mathbf{f} is optionally constrained in a hypothesis space \mathcal{H}_f (e.g., by choosing \mathbf{f} as a neural network). Alternatively, one may instantiate the objective (1) as the following one that pursues an implicit representation of \mathcal{S}

$$\min_{F \in \mathcal{H}_F} L^{\text{imp}}(F; \mathcal{P}) + \lambda R^{\text{imp}}(F), \quad (4)$$

where L^{imp} and R^{imp} are again the instantiated functions, and \mathcal{H}_F denotes a hypothesis space that optionally constrains the implicit function F . We present the subsequent sections based on the notations defined in Table I, unless specified otherwise.

III. SURFACE RECONSTRUCTION WITH A CATEGORIZATION OF GEOMETRIC PRIORS

Given an observed point set \mathcal{P} , a reconstructed surface \mathcal{S} should be close to \mathcal{P} under some distance metric; this is guaranteed by the first term $L(\mathcal{S}; \mathcal{P})$ of data fidelity in the abstract objective (1). Section II also suggests that $L(\mathcal{S}; \mathcal{P})$ can be instantiated either explicitly or implicitly. An example of the explicit form can be written as

$$L^{\text{exp}}(\mathbf{f}; \mathcal{P}) = \frac{1}{n_{\mathcal{P}}} \sum_{\mathbf{p} \in \mathcal{P}} \min_{\mathbf{x} \in \Omega^2} \|\mathbf{f}(\mathbf{x}) - \mathbf{p}\|_{\ell}, \quad (5)$$

where $\|\cdot\|_{\ell}$ denotes a proper norm of distance, with ℓ typically set as 1 or 2; the above term (5) constrains the learning of mapping function $\mathbf{f} \in \mathcal{H}_f$. In practice, one may sample a fixed set $\{\mathbf{x} \in \Omega^2\}$ instead of optimizing over the whole domain Ω^2 , which gives variants of (5) based on point-set distances, such

¹By writing the simple form (1), we omit the notation of some information that can be computed from the observed \mathcal{P} , e.g., surface normals associated with individual points $\mathbf{p} \in \mathcal{P}$, and we also omit some information that would be useful for surface reconstruction but needs to be additionally provided, e.g., the pose of the scanner under which \mathcal{P} is captured; given the additionally provided camera pose, one can rewrite (1) as

$$\min_{\mathcal{S}} L(\mathcal{S}; \mathcal{P}, \mathbf{K}) + \lambda R(\mathcal{S}), \quad (2)$$

where \mathbf{K} denotes the extrinsic matrix of the camera. The formulation (2) appears to be useful for surface reconstruction, by using the provided \mathbf{K} to determine the orientations of surface normals (cf. Section VI-C for the details).

TABLE I
MATH NOTATIONS

Notation	Description
\mathcal{S}^*	An underlying surface to be recovered; $\mathcal{S}^* \subset \mathbb{R}^3$.
\mathcal{S}	A reconstructed surface; $\mathcal{S} \subset \mathbb{R}^3$.
$\mathcal{G}_{\mathcal{S}}$	A triangular mesh representation of surface \mathcal{S} . A mesh with $n_{\mathcal{G}}$ faces is collectively written as $\mathcal{G}_{\mathcal{S}} \triangleq \{\mathcal{T}_i\}_{i=1}^{n_{\mathcal{G}}}$, where each face \mathcal{T} is specified by $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}$ containing three vertices; we also write as \mathbf{e}_{ij} for the edge connecting vertices \mathbf{v}_i and \mathbf{v}_j .
\mathcal{P}	A set of $n_{\mathcal{P}}$ discrete points $\{\mathbf{p}_i \in \mathbb{R}^3\}_{i=1}^{n_{\mathcal{P}}}$, representing the practical sampling of an underlying surface \mathcal{S}^* .
$\mathbf{n}_{\mathbf{p}}$	An estimated, oriented surface normal defined at a surface point \mathbf{p} ; $\mathbf{n}_{\mathbf{p}} \in \mathbb{R}^3$.
$\mathcal{N}(\mathbf{p})$	A local neighborhood of points centered at \mathbf{p} .
Ω	A domain of subset space, e.g., $\Omega^k \subset \mathbb{R}^k$.
\mathcal{C}^k	The smoothness of a function, where k is the number of continuous derivatives the function has over some domain.
$d(\mathbf{p})$	A signed or unsigned distance field value at a point \mathbf{p} .
\mathbf{f}_{θ} or F_{θ}	An explicit or implicit model of surface reconstruction parameterized by θ (e.g., a neural network); the parameters are also denoted as θ_f or θ_F .
$\nabla_x \mathbf{f}_{\theta}$ or $\nabla_x F_{\theta}$	Model derivative with respect to x .
$L(\mathcal{S}; \mathcal{P})$	Loss function of a reconstructed surface \mathcal{S} for data fidelity to an observed \mathcal{P} .
$R(\mathcal{S})$	Regularizer imposed on a reconstructed surface \mathcal{S} .
\mathbf{K}	Extrinsic matrix of a camera.

as Chamfer or Hausdorff distances. An example of the implicit form of $L(\mathcal{S}; \mathcal{P})$ can be written as

$$L^{\text{imp}}(F; \mathcal{P}) = \frac{1}{n_{\mathcal{P}}} \sum_{\mathbf{p} \in \mathcal{P}} \|F(\mathbf{p})\|_{\ell}, \quad (6)$$

which learns the implicit function $F \in \mathcal{H}_F$ by minimizing a proper norm of $F(\mathbf{p})$ for any $\mathbf{p} \in \mathcal{P}$. Advanced versions of the implicit data fidelity loss exist, e.g.,

$$\begin{aligned} L^{\text{imp++}}(F; \mathcal{P}) &= \alpha_1 \mathbb{E}_{\mathbf{q} \in \mathbb{R}^3} \|F(\mathbf{q}) - d(\mathbf{q}; \mathcal{P})\|_{\ell_1} \\ &\quad + \alpha_2 \mathbb{E}_{\mathbf{q} \in \mathbb{R}^3} \|\nabla_{\mathbf{q}} F(\mathbf{q}) - \mathbf{n}(\mathbf{q}; \mathcal{P})\|_{\ell_2} + \dots, \end{aligned} \quad (7)$$

where when F models a Signed Distance Function (SDF) [8], [33], [34], $d(\mathbf{q}; \mathcal{P})$ denotes the signed distance between any space point $\mathbf{q} \in \mathbb{R}^3$ and the observed point set \mathcal{P} , which vanishes when \mathbf{q} hits any $\mathbf{p} \in \mathcal{P}$, and when F models an Occupancy Field (OF) [9], [35], [36], $d(\mathbf{q}; \mathcal{P}) \in \{0, 1\}$ depending on whether \mathbf{q} is inside or outside the surface \mathcal{S} , which is practically estimated by comparing \mathbf{q} with the observed \mathcal{P} ; $\mathbf{n}(\mathbf{q}; \mathcal{P})$ denotes the normal at point \mathbf{q} , which, when \mathbf{q} hits some $\mathbf{p} \in \mathcal{P}$, can be estimated by computing the local tangent plane of \mathcal{P} at \mathbf{p} , and $\|\mathbf{n}(\mathbf{q}; \mathcal{P})\|_2 = 1$ otherwise (when F models an SDF [37], the second term in (7) is correspondingly written as $\mathbb{E}_{\mathbf{q} \in \mathbb{R}^3} \|\nabla_{\mathbf{q}} F(\mathbf{q})\|_2 - 1$); one [37], [38], [39] may use $\{\alpha_1, \alpha_2, \dots\}$ to weight or switch on/off different terms in (7).

Due to the ill-posed nature of surface reconstruction from \mathcal{P} , neither of the data fidelity loss terms (5) or (6) is sufficient to reconstruct a geometry-plausible \mathcal{S} . In literature, various instantiations of the regularization $R(\mathcal{S})$ in (1) have been proposed, in order to make the problem be better posed. In the

remainder of this section, we discuss the essence of existing surface reconstruction methods by categorizing their adopted regularization of geometric priors, including *triangulation-based prior*, *smoothness prior*, *template-based prior*, *modeling prior*, *learning-based prior*, and *hybrid prior*, where we include both classical and the recent, deep learning ones. We expect our categorization and discussion would foster new innovations by bridging classical surface reconstruction methods with the more recent deep learning solutions.

A. Triangulation-Based Prior

A closed surface is continuous and could be locally differentiable up to different orders. When the \mathcal{S} to be recovered from \mathcal{P} is locally differentiable up to the first order, i.e., f is locally of C^1 , a piecewise linear assumption stands as a good prior for modeling the surface, which gives a *mesh* representation of the surface. Among various meshing schemes, Delaunay triangulation [40] is a classical one that approximates \mathcal{S} as a mesh that satisfies the specific empty sphere property, which can be written as

$$\begin{aligned} \mathcal{G}_{\mathcal{S}} &= \{\mathcal{T}_i\}_{i=1}^{n_{\mathcal{G}}} \\ \text{s.t. } v_1^{\mathcal{T}} &\in \mathcal{P}, v_2^{\mathcal{T}} \in \mathcal{P}, v_3^{\mathcal{T}} \in \mathcal{P} \forall \mathcal{T} \in \mathcal{G}_{\mathcal{S}}, \\ (\mathbf{p} &\notin \text{CC}(\mathcal{T}) \forall \mathbf{p} \in \mathcal{P}/\{v_1^{\mathcal{T}}, v_2^{\mathcal{T}}, v_3^{\mathcal{T}}\}) \forall \mathcal{T} \in \mathcal{G}_{\mathcal{S}}, \end{aligned} \quad (8)$$

where \mathcal{T} is a triangular face with its three vertices denoted as $\{v_1^{\mathcal{T}}, v_2^{\mathcal{T}}, v_3^{\mathcal{T}}\}$, $n_{\mathcal{G}}$ is the number of faces, and $\text{CC}(\mathcal{T})$ denotes the space enclosed by the circumscribing sphere of \mathcal{T} passing through its three vertices; note that by Delaunay triangulation, the explicit data fidelity (5) is satisfied simultaneously. We also note that although Delaunay triangulation is flexible in adapting to complex shapes, it is not well-defined in 3D. Most reconstruction methods take it as an initialization [41] or regularization [42], or extend Delaunay tetrahedralization to remove the triangles of undesirable shape in reconstruction [6].

Representative methods [41], [42] of Delaunay triangulation first generate a set of triangular faces directly from the observed \mathcal{P} , and then select the optimal subset from them to generate the final triangular mesh. Greedy Delaunay (GD) [41] proposes greedy algorithm based on topological constraints to select valid triangles sequentially, where the initial triangles are generated by Delaunay triangulation; Ball-Pivoting Algorithm (BPA) [42] uses balls with various radii rolling over the points in \mathcal{P} to generate triangles, where every three points touched by a rolling ball will construct a new triangle if the triangle does not encompass any other points, which can also be regarded as an approximation of Delaunay triangulation.

B. Surface Smoothness Priors

In more general cases, the surface \mathcal{S} to be recovered is expected to be *smooth* or continuously differentiable up to a certain order [32]. Surface smoothness is usually enforced by the following strategies.

Surface smoothness via local weighted combination – The first strategy smoothes out $\{\mathbf{p} \in \mathcal{P}\}$ via local weighted combination when fitting the explicit mapping function f . For example,

the smoothness prior in [43] is an *regularized* version of (5)

$$\begin{aligned} \min_{\mathbf{f}} L^{\text{exp}}(\mathbf{f}; \mathcal{P}) &= \frac{1}{n_{\mathcal{P}}} \sum_{\mathbf{p} \in \mathcal{P}} \min_{\mathbf{x} \in \Omega^2} \|\mathbf{f}(\mathbf{x}) - \hat{\mathbf{p}}(\mathbf{p})\|_2^2 \\ \text{s.t. } \hat{\mathbf{p}}(\mathbf{p}) &= \sum_{\mathbf{p}' \in \mathcal{N}(\mathbf{p})} \mathbf{p}' g(\|\mathbf{p} - \mathbf{p}'\|_2), \end{aligned} \quad (9)$$

where $\mathcal{N}(\mathbf{p})$ denotes a local neighborhood of the observed \mathbf{p} , containing $\{\mathbf{p}' \in \mathcal{P}\}$, and $g(\|\mathbf{p} - \mathbf{p}'\|_2)$ denotes a normalized weighted function (e.g., a normalized Radial Basis Function or RBF) whose value decreases as the distance $\|\mathbf{p} - \mathbf{p}'\|_2$ increases. In Poisson Surface Reconstruction (PSR) [38], a similar implicit objective is used by regularizing the second-order term in (7)

$$\begin{aligned} \min_F L^{\text{PSR}}(F; \mathcal{P}) &= \mathbb{E}_{\mathbf{q} \in \mathbb{R}^3} \|\nabla_{\mathbf{q}} F(\mathbf{q}) - \hat{\mathbf{n}}(\mathbf{q}; \mathcal{P})\|_2^2 \\ \text{s.t. } \hat{\mathbf{n}}(\mathbf{q}; \mathcal{P}) &= \sum_{\mathbf{p}' \in \mathcal{N}(\mathbf{q})} \mathbf{n}(\mathbf{p}'; \mathcal{P}) g(\|\mathbf{q} - \mathbf{p}'\|_2), \end{aligned} \quad (10)$$

where $\mathcal{N}(\mathbf{q})$ denotes a local neighborhood of a space point $\mathbf{q} \in \mathbb{R}^3$, containing observed points $\{\mathbf{p}' \in \mathcal{P}\}$. PSR constrains the normal field only and usually produces over-smooth results. As a remedy, Screened Poisson Surface Reconstruction (SPSR) [39] improves over PSR by incorporating regularized versions of both the first- and second-order terms in (7). More recently, Shape As Points [44] develops a differentiable Poisson solver in a spectral manner, which enables an end-to-end optimization and thus could be integrated into the learning of deep neural networks.

Surface smoothness by constraining function complexities – This can be equivalently achieved by constraining the hypothesis space of \mathcal{H}_f or \mathcal{H}_F , e.g., by constraining \mathcal{H}_f as B-spline functions [45], [46], [47], NURBS [48], [49] or Gaussians [50], [51], [52]. Intuitively, a more complex function is able to fit a surface of complex geometry; but it also tends to be overfitted to the observed \mathcal{P} , producing a less smooth surface. For example, using RBFs in [53] means that the approximate function is in the form of low-degree polynomials with an interpolation of many basic functions centered at the observed points. The works [54], [55] are similar to [53] but approximate their respective implicit field functions using different basis functions. More specifically, Kazhdan [54] first computes the Fourier coefficients of its implicit field function with the help of Monte-Carlo approximation of Divergence Theorem, and then uses inverse Fourier transform to obtain the implicit function for extraction of iso-surface; a regular grid is needed to perform fast Fourier transform in [54], and instead Manson et al. [55] use wavelets, which provide a localized, multi-resolution representation of the implicit function. Instead of using a regular grid, [50] optimizes the likelihood function over a finer grid under some topological constraints by solving the parameters of the Gaussian basis function. More recently, He et al. [56] propose to regularize the curvatures of the mesh when reconstructing the surface, which results in a reconstructed surface that is smoother. Wang [51] introduces a Gaussian convolution to the energy term, which promotes smoothness on the learned implicit function during the iterative optimization process. PGR [52] chooses the basis of the indicator

function that is guided by a Gaussian formulation to recover the surface from un-oriented points.

Surface smoothness by combination of strategies – Methods such as Point Set Surfaces (PSS) [43], [57], [58], [59] combine both of the above smoothing strategies. PSS is derived from Moving Least Squares (MLS) [60], [61], [62], [63], whose surface can be defined either by an explicit function with stationary projection operator [43], [57], [58], [60], [61] or by an implicit function [59], [62], [63]. In either case, a weighted combination of spatially-varying low-degree polynomials acts as the most important ingredient to locally approximate the observed points and construct the surfaces. In Simple Point Set Surfaces (SPSS) [58], the authors iteratively project all the given points along the normal directions onto the local reference planes, which are defined by a weighted average of the points to be projected and their neighborhood points; then the local reference plane at each evaluation point would give a local orthogonal coordinate system to compute a local bivariate polynomial approximation to the surface. However, the local reference plane can hardly be a good approximation and sometimes even becomes unstable when the observed points are sparse. Algebraic PSS [59] overcomes this issue by using an algebraic sphere to fit the observed points, which forms an implicit function to represent the algebraic distance between the evaluation point and the fitted sphere; the fitting problem is then solved by a least squares problem between the gradient field of the algebraic sphere and the normals of the observed points. Robust Implicit MLS (RIMLS) [63] combines kernel regression and statistical robustness with MLS to cope with the limitation that MLS can only reconstruct smooth surfaces. Other PSS methods share the same principle; more details can be found in [64] and [3].

C. Template-Based Priors

Template-based priors assume that a surface could be represented by combination of a group of templates, where the templates could be geometric primitives such as spheres or cubes, or complex ones from an auxiliary dataset. As such, surface reconstruction boils down as the problem of estimating/fitting the correct templates to the observed \mathcal{P} . A way to define this mathematically spells out as

$$\min_{\{w\}, \{\theta\}} D \left(\sum_{i=1}^{|M|} w_i M_i(\theta_i), \mathcal{P} \right), \quad (11)$$

where $\{M\}$ denotes the set of predefined templates, and each template M is parameterized by (the possibly learnable) θ and weighted by w ; $D(\cdot, \cdot)$ denotes a proper distance between the formed surface and \mathcal{P} . By minimizing the fitting error (11), a surface is reconstructed by the determined $\{w\}$ and/or $\{\theta\}$.

1) *Geometric Primitives*: Templates of geometric primitives are simple shapes that can be analytically represented by a certain number of parameters, e.g., cuboids, spheres, cylinders, cones, etc. Random sample consensus (RANSAC) [65], [66] is the most commonly used method to solve (11) that fits the right templates to the observed \mathcal{P} , where $D(\cdot, \cdot)$ sums up the element-wise (Euclidean) distances between randomly

sampled $\{p \in \mathcal{P}\}$ and their projections onto the fitted templates. More specifically, Schnabel et al. [65] introduce an efficient RANSAC-based algorithm with a novel sampling strategy and an efficient score evaluation scheme, where the primitive with maximum score is extracted iteratively. Nan and Wonka [66] propose to only extract planar primitives based on RANSAC, obtaining the final surface by minimizing a weighted sum of several energy terms to select the optimal set of faces. Zhu et al. [67] reconstruct the surface by fitting the segmented point clouds with different primitive patches defined by the solution of the partial differentiable equation.

2) *Retrieval-Based Templates*: Given an auxiliary set $\{M\}$ of shape models, retrieving-and-deforming methods solve (11) to find the closest shapes and deform them, via optimization of model parameters $\{\theta\}$, to fit the observed \mathcal{P} . For example, in scene reconstruction [68], [69], [70], [71], the observed scene points are segmented into semantic classes, each of which is then fit with a retrieved shape model followed by rigid or non-rigid deformation; in object reconstruction, Pauly et al. [72] warp and blend multiple retrieved object shapes to conform with the observed points, and Shen et al. [73] retrieve individual object parts to form a surface by part assembly.

D. Modeling Priors

While constraining the complexity of hypothesis space of \mathcal{H}_f or \mathcal{H}_F would promote reconstruction of smoother surfaces, as discussed in Section III-B, the choice of \mathcal{H}_f or \mathcal{H}_F itself regularizes the reconstruction given that only specific types of surface can be modeled by the choice. A prominent example is the recent trend of using deep neural networks for geometric modeling and surface reconstruction. We term the geometric priors provided by the respective designs of models themselves as *modeling priors*.

Motivated from deep image prior [74], Deep Geometric Prior (DGP) [75] verifies the efficacy of deep networks as a prior for geometric surface modeling, *even when the networks are not trained*. Later on, Point2Mesh [76] and SAIL-S3 [77] extend the global modeling adopted in [75] as local ones, where the former constructs its local, implicit functions as a weight-shared MeshCNN [78] and the later method constructs them as a weight-shared Multi-Layer Perceptron (MLP). Deep Manifold Prior [79] delivers mathematical analyses on such modeling properties for MLP as well as convolutional networks. There have also been a few works making use of modeling priors while not explicitly mentioning it. Atzmon et al. [80] theoretically prove that MLP with Rectified Linear Units (ReLUs) generate piecewise linear surfaces, and a meshing algorithm of Analytic Marching is also proposed in [81] that is able to analytically compute the piecewise linear surface mesh from such a network, both of which deliver mathematical analyses on how the structure of MLP itself acts as a regularizer. Later on, Deep Manifold Prior [79] extends such modeling properties for both MLP and convolutional networks. Implicit Geometric Regularization (IGR) [37] shows that additional regularization on the gradient of neural network would further encourage the generated surfaces to be smooth. Sign Agnostic Learning (SAL) [82] and its variants [83], [84] study reconstructing a surface from an un-oriented

point cloud via a specially initialized neural network, SSP [85] further utilizes semi-signed supervision generated by voxelizing the un-oriented point inputs to avoid bad local minimums, and Davies et al. [86] adopt the same initialization strategy for surface reconstruction from an oriented point cloud. Neural Splines [87] performs reconstruction based on random feature kernels arising from infinitely-wide shallow ReLU networks and shows that such solutions bias toward reconstruction of smooth surface.

E. Learning-Based Priors

Given the parametrization of θ_f for $f \in \mathcal{H}_f$ and θ_F for $F \in \mathcal{H}_F$, priors can be learned from an auxiliary set of training shapes as *optimized model parameters*. Note that such learning-based priors are different from modeling priors presented in the preceding section, where priors are provided by the models themselves. Let $\{\mathcal{P}_i, \mathcal{S}_i^*\}_{i=1}^N$ be the training set containing N pairs of observed point sets and their corresponding ground-truth surfaces. By adapting the objective (3), the following explicit surface reconstruction based on a learned prior is proposed in [5], [88]

$$\min_z L^{\exp} (f(z; \tilde{\theta}_f); \mathcal{P}) + \lambda R^{\exp}(z) \quad (12)$$

$$\text{s.t. } \tilde{\theta}_f = \arg \min_{\theta_f} \frac{1}{N} \sum_{i=1}^N \tilde{L}^{\exp} (\theta_f; \{\mathcal{P}_i, \mathcal{S}_i^*\}) + \tilde{\lambda} \tilde{R}^{\exp}(\theta_f), \quad (13)$$

where the constraint (13) learns the prior as the optimized model parameter $\tilde{\theta}_f$, \tilde{L}^{\exp} could be different from L^{\exp} , and an objective for smooth and/or fair surface may also be incorporated into \tilde{R}^{\exp} in addition to a simple norm constraint of θ_f ; given the learned and then fixed $\tilde{\theta}_f$, the objective (12) fits the model prediction to any observed \mathcal{P} by optimizing a latent code z , where norm of z is usually penalized to prevent overfitting. Similarly, by adapting the objective (4), the following methods [8], [9] learn a prior for implicit surface reconstruction

$$\min_z L^{\text{imp}} (F(z; \tilde{\theta}_F); \mathcal{P}) + \lambda R^{\text{imp}}(z) \quad (14)$$

$$\text{s.t. } \tilde{\theta}_F = \arg \min_{\theta_F} \frac{1}{N} \sum_{i=1}^N \tilde{L}^{\text{imp}} (\theta_F; \{\mathcal{P}_i, \mathcal{S}_i^*\}) + \tilde{\lambda} \tilde{R}^{\text{imp}}(\theta_F). \quad (15)$$

The model f can also be constructed as an auto-encoder architecture [9], [35], [89], i.e., $f = f_{\text{encoder}} \circ f_{\text{decoder}}$. In such a case, given the prior $\tilde{\theta}_f = \{\tilde{\theta}_{f_{\text{encoder}}}, \tilde{\theta}_{f_{\text{decoder}}}\}$ already learned by (13), a latent code can be directly obtained as

$$z = f_{\text{encoder}} (\mathcal{P}; \tilde{\theta}_{f_{\text{encoder}}}), \quad (16)$$

instead of optimizing the objective (12). The above applies to an auto-encoder based implicit function $F = F_{\text{encoder}} \circ F_{\text{decoder}}$ as well, and given the learned prior $\tilde{\theta}_F = \{\tilde{\theta}_{F_{\text{encoder}}}, \tilde{\theta}_{F_{\text{decoder}}}\}$, one can compute its latent code directly as

$$z = F_{\text{encoder}} (\mathcal{P}; \tilde{\theta}_{F_{\text{encoder}}}). \quad (17)$$

Note that the most recent implicit methods [36], [90] allow the priors of model parameters to be further optimized when fitting to the observed points, i.e., optimizing over both the latent code and model parameters in (12) or (14), and thus potentially bridge the gap between the learning-based priors and classical ones.

Depending on how the training shapes in $\{\mathcal{P}_i, \mathcal{S}_i^*\}_{i=1}^N$ are organized, the priors learned via (13) or (15) may be either at a global, semantic level or as local, shape primitives. For example, when pairs in $\{\mathcal{P}_i, \mathcal{S}_i^*\}_{i=1}^N$ capture surfaces of object instances belonging to a semantic category, the learned priors would encode shape patterns common to this object category. To learn priors for reconstruction of arbitrary surface shapes, one may have to prepare $\{\mathcal{P}_i, \mathcal{S}_i^*\}_{i=1}^N$ as those encoding surface patches, and expect a global surface of arbitrary shape can be better reconstructed by providing priors on its local shape primitives.

1) Learning Semantic Priors: The recent trend of deep learning surface reconstruction starts from learning deep priors at the semantic level (e.g., a category of object instances). For example, an explicit method of Deep Marching Cubes [88] proposes a novel, differentiable layer of marching cubes, which connects mesh surface generation with the learning of semantic prior via a shape encoding network that generates displaced voxel grids. The seminal deep learning-based implicit methods [8], [9], [91] model either SDF or OF via deep neural networks. Specifically, OccNet [9] and IM-Net [91] adopt an auto-encoder structure; after training, the encoder generates the latent code representing the shape via a single forward propagation in (17) for any observed point set, and the decoder predicts the probability of occupancy according to the given space point along with the latent code. Different from OccNet and IM-Net, DeepSDF [8] adopts the structure of decoder-only model, which optimizes the latent code of the given point clouds via maximum a posteriori in (14), and predicts signed distances according to the given space point along with the latent code. Curriculum DeepSDF [92] improves DeepSDF by introducing a progressive learning strategy to learn local details; in the meanwhile, Yao et al. [93] implement such learning using Graph Neural Networks, which converts the global, semantic latent code into more sophisticated local ones, before feeding into the decoder. MeshUDF [94] further extends DeepSDF to reconstruct open surfaces, which predicts unsigned distances of any given space point and generates the surface using their customized marching cubes. More recently, a few methods [95], [96] adopt networks based on transformer [97], [98] to help reconstruction. Note also that learning semantic priors enables reconstruction of surfaces from raw observations that are originally of no or less 3D shape information (e.g., as few as a single RGB image [5], [99], [100]), by training encoders that learn latent shape spaces from such observations.

2) Learning Priors as Local, Shape Primitives: To reconstruct a scene surface or surface of an object that cannot be semantically categorized, existing methods resort to modeling and learning local priors of shape primitives either at regular grids that partition the 3D space [10], [34], [35], [101], [102], [103], [104], or on local patches along the surface manifold [33], [89], [105]. Among the former methods, Implicit Feature Networks (IF-Net) [101] and Convolutional

OccNet [35] (ConvOccNet) obtain latent codes for local grids via auto-encoder structure, where IF-Net [101] adopts 3D convolution that convolves each input point with its surrounding points to get the latent code at each local grid, and ConvOccNet [35] uses PointNet [106] as its encoder to get the latent code for each point and then encapsulates all the latent codes into a volumetric feature via average pooling. Later on, Chibane et al. [102] extend IF-Net [101] to support sign-agnostic learning. Deep Local Shape (DeepLS) [34] divides the whole 3D space into regular voxels, and trains an implicit function whose parameters are shared among different local voxels. Local Implicit Grid (LIG) [10] trains an auto-encoder to extract the latent codes of local voxels during training (via (17)), while retaining the voxel-shared decoder with fixed parameters during inference only. To improve the efficiency of local encoding, Scalable Surface Reconstruction Network (SSRNet) [103] makes use of octree to partition the whole 3D space, and uses fully-convolutional U-shaped network with skip connections, which is based on modified tangent convolution [107] to get the latent code for each local grid. Neural Geometric Level of Detail (LOD) [104] adopts the structure of sparse octree to further improve the efficiency, where latent codes are computed only for local voxels that intersect with the surface. Ummenhofer et al. [108] aggregate latent codes for local grids via adaptive grid convolution on multiple levels of the octree input space. As for the methods based on local surface patches, Badki et al. [105] introduce the concept of meshlets, and train a variational auto-encoder to learn the latent space of pose-disentangled meshlets; by back-propagating the error with respect to the given points and the meshlets, the method updates the meshlets' latent codes and deforms the meshlets to fit the given points at inference time. PatchNets [33] leverages the structure of implicit auto-decoder to learn across different local surface patches, and the decoder is also trained with an elaborate loss function to ensure the smoothness of the reconstructed patches. Points2Surf [89] learns features from both local patches and the global surface, and reconstructs the surface with an implicit decoder, where the former takes a local encoder to learn the absolute distance of a queried point from the local surfaces, and the latter learns the interior/exterior of the surface with a global encoder. POCO [109] and [110] encodes each input point into a single latent code, and then perform weighted interpolation among a point and its neighboring ones to get the local latent code. Ma et al. [111] train a local context network that encodes the points in local regions to be latent codes, and then the queries of those regions are decoded by SDFs conditioned on the codes. During testing, another query network is optimized to predict the corresponding local queries of the global region and the condition codes, which are then fed into the decoder of the local context network to obtain SDFs for the surface.

F. Hybrid Priors

Methods discussed in the preceding sections are organized according to their respectively used, main priors of surface geometry. To improve the plausibility of reconstructed surfaces,

existing methods usually combine multiple priors, e.g., by combining smoothness priors with triangulation or template-based ones [112], [113], [114], by imposing additional priors on top of modeling ones [115], [116], [117], [118], [119], [120], [121], [122], or by learning priors to improve over regularization provided by pre-defined ones [6], [7], [47], [123], [124], [125]. In this section, we focus our discussion on the last case, considering that such hybrid priors are popularly used in the new era of deep learning surface reconstruction.

Given auxiliary sets of training shapes, learning-based priors are combined with triangulation-based prior in [7], [123], [126] that encourage deep networks to learn particular properties for triangulation from the observed point clouds. More specifically, PointTriNet [7] introduces a framework to generate triangles from observed point clouds directly, where a proposal network suggests candidates of triangles and a classification network predicts whether a proposed candidate should appear in the reconstructed surface or not, and two networks iteratively take effects until the final surface of triangular mesh is generated. Liu et al. [126] show that connecting those pairs of vertices whose geodesic distance approaches to their Euclidean distance can approximately reconstruct the underlying surface; they construct a network to select candidate triangles satisfying this property. Delaunay Surface Elements (DSE) [123] introduces small triangulated patches generated by combining Delaunay triangulation with learned logarithmic maps, from which candidate triangles of the output surface are then iteratively selected via their proposed adaptive voting algorithm. DeepDT [6] learns deep networks to extract geometric features from observed point clouds, and then integrates the learned features together with graph structural information to vote for the inside/outside labels of Delaunay tetrahedrons; the surface is reconstructed by extracting triangular faces between tetrahedrons of different labels. Gao et al. [127] learn a network to predict the offsets of vertices of tetrahedrons and a second network to predict the occupancy of each tetrahedron, and object surface is generated on the facet that belongs to two tetrahedrons with different occupancies.

Learning-based priors are commonly combined with smoothness priors (e.g., the Laplacian regularization in [128] and the cosine smoothness in [129]). IMLSNet [124] trains an implicit network that evaluates signed distances at grids of an octree-based 3D space, where smoothness is regularized by defining the signed distances in a way similar to implicit moving least-squares [63]. Xiao et al. [130] learn a network to implicitly model an indicator function derived from Gauss lemma, which is smooth and linearly approximates the surface; similar to Points2Surf [89], they learn both local and global features.

There have also been plenty of works [47], [125], [131], [132] combining learning-based priors with template-based ones. SPFN [125] proposes to train a deep network to fit geometric primitives according to the observed point clouds; the trained network predicts point-wise features that are fed into a differentiable estimator for algebraic computation of primitive parameters. CPFN [131] learns two SPFNs, where a global one is used for detecting bigger primitives and a local one is used for capturing smaller primitives. ParseNet [47] uses a neural decomposition module to partition an observed point cloud into

TABLE II
THE BENCHMARK COLLECTION OF SYNTHETIC OBJECT INSTANCES FROM EXISTING REPOSITORIES AND THEIR DISTRIBUTIONS IN THE THREE GROUPS OF LOW, MIDDLE, AND HIGH COMPLEXITIES

Surface complexity	Low	Middle	High	Total
Thingi10k [22]	516	230	97	845
3DNet Cat200 subset [20]	144	89	33	266
ABC [21]	312	160	6	478
Three D Scans [23]	0	7	26	31
Our benchmark	972	486	162	1,620

multiple subsets, each of which is assumed to be a primitive type modeled as an open or close B-spline by a deep network. Local Deep Implicit Function (LDIF) [132] represents a surface shape as a set of shape elements, each of which is parameterized by analytic shape variables and latent shape codes; the method learns such variables and latent codes by its SIF encoder and PointNet encoder respectively. For reconstruction of a large-scale scene surface, RetrievalFuse [133] retrieves a set of object templates from an auxiliary scene dataset, and learns a network with attention-based refinement to produce the reconstruction.

IV. A SURFACE RECONSTRUCTION BENCHMARK

As discussed in Section III, a rich set of methods exist that aim to address the ill-posed problem of surface reconstruction from point observations. These methods have their respective merits, yet it is less clear on their advantages/disadvantages under different working conditions, due to the lack of a comprehensive surface reconstruction benchmark that identifies and includes the main challenges faced by the studied problem.

In this work, we contribute such a benchmark by both *synthesizing* and *practically scanning* point clouds of object and scene surfaces. We identify the main challenges of surface reconstruction from point clouds obtained by imperfect surface scanning, including *point-wise noise*, *point outliers*, *non-uniform distribution* of points, *misalignment* among point sets obtained by scanning different but overlapped, partial surfaces of an object or scene, and *missing points* of one or several surface patches. An illustration of such challenges is given in Fig. 1. We include all the five challenges in synthetic data of the benchmark, and expect that an arbitrary combination of these challenges may appear in any sample of the real-scanned data. The benchmark is organized as the synthetic data of object surfaces, the synthetic data of scene surfaces, and the data of real-scanned surfaces. Tables III and IV summarize the statistics. We make the benchmark publicly accessible at <https://Gorilla-Lab-SCUT.github.io/SurfaceReconstructionBenchmark>, where we also release the code implementation of our synthetic scanning pipeline to facilitate future research in the community.

A. The Synthetic Data of Object Surfaces

To prepare synthetic data of object instances in the benchmark, we collect CAD models from existing repositories [20], [21], [22], [23], and pre-process them before synthetically scanning their point clouds. For each CAD model, we simulate the aforementioned five ways of imperfect scanning, in addition to

TABLE III
STATISTICS OF SYNTHETIC DATA IN THE BENCHMARK

	Object level	Scene level
normalization [r_{\min}, r_{\max}] (camera distancing)	centering+scaling [2.5, 3.5]	centering [0.75m, 2.1m]
point-wise noise σ_{noise}	0.001/0.003/0.006 [0.01, 0.1]	0.005m 0.1%/0.3%/0.6%
point outliers r_{outlier}	[$0.01, 0.1$]	[0.01m, 0.1m] 0.4%
missing points $\Delta\varphi$	[$20^\circ, 40^\circ, 60^\circ$]/[$20^\circ, 40^\circ$]/[20°] 3°	-
misalignment [$a_{\text{rotation}}, b_{\text{rotation}}$] [$a_{\text{translation}}, b_{\text{translation}}$]	[$-0.5^\circ, 0.5^\circ$]/[$-1^\circ, 1^\circ$]/[$-2^\circ, 2^\circ$] [-0.005, 0.005]/[-0.01, 0.01]/[-0.02, 0.02]	[$-1.5^\circ, 1.5^\circ$] [-0.015m, 0.015m]
#viewpoints	1000	$100/\text{m}^3$
#scanned points	low complexity middle complexity high complexity	80k 120k 160k
#surfaces	1620	50

Elements in each triple $\cdot / \cdot / \cdot$ represent the hyper-parameters that control the scanning with three levels of severity; # denotes the number. (cf. Sections 4.1 and 4.2 for specific meanings of the math notations.)

TABLE IV
STATISTICS OF REAL-SCANNED DATA

	Einscan SE	OKIO 5M
precision	100 micrometers	5 micrometers
professional operators	✗	✓
#resolution	$1300k$	$5000k$
#viewpoints	30-50	13-35
#scanned points	$210k$ - $3000k$	$330k$ - $2000k$
#surfaces	20	
#material	metal	4
	plastic	9
	ceramic	5
	cloth	2

denotes the number.

a perfect scanning that gives a clean object point cloud; we in total have six kinds of scanned point clouds for each instance. We organize the collected object instances into three levels of (algebraic) surface complexity [134], in order to study how different methods perform on surface reconstruction of varying complexities.

1) *Data Collection*: We collect CAD models of object instances from the existing repositories of Thingi10k [22], ABC [21], 3DNet Cat200 subset [20], and Three D Scans [23]. More specifically, we randomly select 3,000 objects from Thingi10k [22] (a dataset collected from online-shared 3D printing models), 3,000 industrial components from the set of Chunk 0080 in ABC [21], 3,400 commodities from 3DNet Cat200 subset [20], and 106 art sculptures from Three D Scans [23]. All these instances are represented in the form of triangular mesh. As illustrated in Fig. 2, some meshes of the collected instances could be non-watertight, with self-occlusion, and/or topologically too complex, and are thus less convenient to be synthetically scanned to simulate practical sensing conditions; we filter out these instances by determining their states of being watertight and 2D-manifold [135], checking self-occlusion [136], and calculating their surface genus [32]. We finally normalize each mesh of the remaining instances by centering it at the origin and scaling it isotropically to fit into the unit sphere. We obtain a total of 1,620 object surface meshes in the benchmark.

2) *Groups of Varying Surface Complexities*: It is possible that performance of different methods depends on the complexities of object surfaces to be reconstructed. To identify

better working conditions for different methods, we intend to divide the above processed object surface meshes into groups of varying complexities. Surface complexities can be measured under different metrics, among which algebraic complexity and topological complexity are the measures more relevant to surface reconstruction from point clouds [134]; simply put, the former measures the degree of polynomials needed to represent a surface, and the latter can be measured as the surface genus (e.g., the number of holes on the surface). Given that our instances of surface meshes have been processed to satisfy the conditions of being watertight, 2D manifold, and topologically simpler (equal to or smaller than genus 5), we divide all the 1,620 instances into three groups of *low*-, *middle*-, and *high-complexity* based on the measure of algebraic complexity.

The algebraic complexity of a surface is usually computed as the highest degree of polynomial functions that approximate/fit local patches on the surface [134]. However, given fixed budget of approximation errors, it is usually unstable to fit local surface patches with polynomials of high degrees [60], causing inaccurate prediction of polynomial degrees and thus that of algebraic complexity. Instead, we take the strategy of fixing the highest function degree and measuring the averaged approximation errors of local surface patches; technical details are given in Appendix A, available online. We finally obtain the three groups respectively of 972 instances, 486 instances, and 162 instances (at the ratio of around 6 : 3 : 1 for low-, middle-, and high-complexity groups). Table II summarizes the group statistics.

3) *Synthetic Point Cloud Scanning*: We use Blender Sensor Simulation Toolbox (BlenSor) [137] to synthetically scan our collected surface meshes of object instances. We describe in this section our scanning pipeline, including how we implement different ways of imperfect scanning that simulate point cloud sensing happening in practical conditions.

Perfect scanning – We first use the case of perfect scanning to give a whole picture of how we conduct our virtual scanning pipeline as shown in Fig. 2. As described in Section IV-A1, our instances of surface meshes have been normalized at the center of a unit sphere in the simulator. To scan an instance, we place a virtual time-of-flight (TOF) camera [138] on viewing spheres whose radii range from $r_{\min} = 2.5$ to $r_{\max} = 3.5$; a viewpoint on any of the spheres can be specified as the camera extrinsic $\mathbf{K} = [\mathbf{R}|\mathbf{t}] \in \mathbb{R}^{4 \times 4}$, where the rotation $\mathbf{R} \in \mathbb{R}^{4 \times 3}$ and translation $\mathbf{t} \in \mathbb{R}^{4 \times 1}$ together specify how the camera is positioned. Denote as $\mathcal{P}_{\mathbf{K}}$ the point cloud obtained by scanning from the viewpoint \mathbf{K} ; one may transform it into the world coordinate system as $\mathbf{K} \circ \mathcal{P}_{\mathbf{K}}$, where \circ denotes an operator.² In our setting, we sample 1,000 viewpoints on the spheres of different radii, resulting in a collection of scanned point clouds $\{\mathbf{K}_1 \circ \mathcal{P}_{\mathbf{K}_1}, \dots, \mathbf{K}_{1000} \circ \mathcal{P}_{\mathbf{K}_{1000}}\}$, each of which partially covers the surface. Then we can register and fuse different partial point clouds to cover the complete surface

²Note that a point $\mathbf{p} \in \mathcal{P}_{\mathbf{K}}$ in the camera coordinate system can be transformed to point $\mathbf{p}^{\text{world}} \in \mathcal{P}_{\mathbf{K}}^{\text{world}}$ in the world coordinate system in terms of a homogeneous equation as $[\mathbf{p}^{\text{world}}; 1]^{\top} = [\mathbf{p}; 1]^{\top} \mathbf{K}^{-1}$; here we transform $\mathcal{P}_{\mathbf{K}}$ to $\mathcal{P}_{\mathbf{K}}^{\text{world}}$ and write collectively as $\mathcal{P}_{\mathbf{K}}^{\text{world}} = \mathbf{K} \circ \mathcal{P}_{\mathbf{K}}$.

$\bigcup_{i=1}^{1000} \mathbf{K}_i \circ \mathcal{P}_{\mathbf{K}_i}$. We obtain the final, uniformly distributed point cloud scanning \mathcal{P} by applying Farthest Point Sampling (FPS) [106] to $\bigcup_{i=1}^{1000} \mathbf{K}_i \circ \mathcal{P}_{\mathbf{K}_i}$; \mathcal{P} is set to contain 80 k points for low-complexity surfaces, 120 k points for middle-complexity surfaces, and 160 k points for high-complexity surfaces.

For existing methods that require surface normals to perform reconstruction, we compute the oriented surface normals as follows. For any point $\mathbf{p} \in \mathcal{P}$, we first compute its un-oriented normal $\bar{\mathbf{n}}_{\mathbf{p}}$ by performing PCA on the local neighborhood constructed by $k = 40$ nearest neighbors of the point; orientation of $\bar{\mathbf{n}}_{\mathbf{p}}$ can be simply determined by comparing \mathbf{p} with the camera position, giving rise to the oriented normal $\mathbf{n}_{\mathbf{p}}$.

Point-wise noise – Due to sensor noise, ambient noise, reflective nature of the surface, and the incapable precision of the scanning devices, point clouds from practical scanning are inevitably noisy. In this case, each scanned point is not exactly on the underlying surface, deviating away from the surface in a point-wise, independent manner; and to simulate such noise, we add point-wise perturbations to the points obtained by the aforementioned perfect scanning. Specifically, for any surface point \mathbf{q} , we generate $\Delta \mathbf{q}_{\text{noise}} \in \mathbb{R}^3$ by randomly sampling its element values from a Gaussian distribution $\mathcal{N}(0, \sigma_{\text{noise}}^2)$ with a truncated values $[-2\sigma_{\text{noise}}, 2\sigma_{\text{noise}}]$. Given a point $\mathbf{p} \in \mathcal{P}$, the corresponding noisy point from noisy scanning is obtained as $\mathbf{p} = \mathbf{q} + \Delta \mathbf{q}_{\text{noise}}$. We set σ_{noise} respectively as 0.001, 0.003, and 0.006 in our benchmark to simulate different severity levels of point-wise noise. Note that the truncation above is to prevent individual $\{\mathbf{p} \in \mathcal{P}\}$ from deviating too far away from the surface, which would become point outliers to be discussed shortly.

Non-uniform distribution of points – Practical scanning often produces a point cloud whose points are not uniformly distributed over the surface. For example, the surface patches that are scanned for multiple times (possibly from different viewpoints) would have more points, and a closer scanning position would produce denser points as well. To simulate such phenomena, we replace the final step of uniformity-promoting FPS in perfect scanning with Random Sampling (RS), and local point densities of the resulting \mathcal{P} would be less uniform over the surface.

Point outliers – As mentioned above, outliers of a surface point cloud are defined as those deviating far away from the surface. They are often caused by impulsive noise of practical scanning. We simulate such outliers as follows. Given a point cloud obtained by perfect scanning, we first randomly sample a ratio r_{outlier} of its points, and for any sampled point \mathbf{q} that is on the surface, we generate $\Delta \mathbf{q}_{\text{outlier}} \in \mathbb{R}^3$ by randomly sampling its element values from a uniform distribution $\mathcal{U}[a_{\text{outlier}}, b_{\text{outlier}}]$; the corresponding point outlier $\mathbf{p}_{\text{outlier}} \in \mathcal{P}$ is then obtained as $\mathbf{p} = \mathbf{q} \pm \Delta \mathbf{q}_{\text{outlier}}$. We set $a_{\text{outlier}} = 0.01$ to distinguish point outliers from noisy points and set $b_{\text{outlier}} = 0.1$ to prevent the outliers from being less relevantly distancing. The ratio r_{outlier} is respectively set as 0.1%, 0.3%, and 0.6% for varying numbers of outliers in each obtained \mathcal{P} .

Misalignment – As described for perfect scanning, a scanning viewpoint is specified by the camera extrinsic $\mathbf{K} = [\mathbf{R}|\mathbf{t}]$; scanning from the viewpoint \mathbf{K} would produce a point cloud $\mathcal{P}_{\mathbf{K}}$ that covers the surface partially; a complete point cloud is

obtained by scanning from 1,000 viewpoints and then registering and fusing the obtained point clouds as $\bigcup_{i=1}^{1000} \mathbf{K}_i \circ \mathcal{P}_{\mathbf{K}_i}$. However, misalignment would happen when the camera extrinsics are less accurate. To simulate such a misalignment, for each viewpoint \mathbf{K} , we generate the perturbation $\Delta\mathbf{K} = [\Delta\mathbf{R}|\Delta\mathbf{t}] \in \mathbb{R}^{4 \times 4}$ where $\Delta\mathbf{R} \in \mathbb{R}^{4 \times 3}$ is obtained by the XYZ Euler angle convention [139], i.e., $\Delta\mathbf{R} = [\mathbf{R}_x(\alpha)\mathbf{R}_y(\beta)\mathbf{R}_z(\gamma); \mathbf{0}_3^\top]$ with α , β , and γ uniformly sampled from $\mathcal{U}[a_{\text{rotation}}, b_{\text{rotation}}]$ and those of the translation perturbation $\Delta\mathbf{t} \in \mathbb{R}^{4 \times 1}$ uniformly sampled from $\mathcal{U}[a_{\text{translation}}, b_{\text{translation}}]$. The final point cloud \mathcal{P} with misalignment is obtained by applying FPS to $\bigcup_{i=1}^{1000} (\mathbf{K}_i + \Delta\mathbf{K}_i) \circ \mathcal{P}_{\mathbf{K}_i}$. We set $[a_{\text{rotation}}, b_{\text{rotation}}]$ as $[-0.5^\circ, 0.5^\circ]$, $[-1^\circ, 1^\circ]$, and $[-2^\circ, 2^\circ]$, and set $[a_{\text{translation}}, b_{\text{translation}}]$ as $[-0.005, 0.005]$, $[-0.01, 0.01]$, and $[-0.02, 0.02]$, which are respectively for different severities of misalignment.

Missing points – Due to surface reflection, self-occlusion, and/or simply insufficient covering of the surface, practical scanning often produces a point cloud that does not cover the whole object surface of interest. Surface reflection depends on a mixed effect of lighting and surface material, and the latter is not included in our collected CAD models. Instead, we take the following simple approach in our benchmark to simulate missing surface points. Rather than allowing the scanning viewpoint \mathbf{K}_i to locate on the whole viewing spheres as in perfect scanning (shown as the blue points in Fig. 2), we only allow it to locate on a limited number of viewing positions to simulate missing points. Specifically, we define the viewing positions in a few narrow bands of trajectories with the polar angle of $\varphi \pm \varphi_\Delta$ (shown as the yellow points in Fig. 2). For different severities, we respectively set the number of trajectories to be 3, 2, and 1, with the polar angle φ set to be $[20^\circ, 40^\circ, 60^\circ]$, $[20^\circ, 40^\circ]$, and $[20^\circ]$ respectively, and with $\varphi_\Delta = 3^\circ$. The scanned surface areas approximately cover 99%, 94%, and 86% of the whole surface for different severities.

B. The Synthetic Data of Scene Surfaces

We adopt a pipeline similar to that presented in Section IV-A for synthetic scanning of the scene-level data. Since the scale of a scene surface is larger and practical scanning usually produces a point cloud that includes multiple types of imperfections. As such, we include all the five challenges of point-wise noise, point outliers, non-uniform distribution of points, misalignment, and missing points into our synthetic scanning of a single scene. Fig. 3 gives the illustration.

Data collection – We choose to collect CAD surface models of indoor scenes in the benchmark. To form the collection, we randomly select 28 indoor scenes from SceneNet [24], 14 from 3D-FRONT [25], and 8 from Replica [29]. The scenes from the three datasets are diverse in terms of varying room types, varying room sizes, and the contained different furniture and furnishings. The collected surfaces are represented in the form of triangular mesh as well. We normalize each scene mesh by centering it at the origin but keeping its original size.

Synthetic scanning – We still use BlenSor [137] to perform our synthetic scene-level scanning. In practical scanning of indoor scenes, the scanner is usually placed at a medium distancing

from the scene surface; we thus choose the sensor of Kinect V2 [140] whose working distance ranges from 0.75 m to 2.1 m. We prepare the scanning by placing each surface mesh of indoor scene in a bounded 3D space, where the scene center has been aligned at the origin and the space size is set to be just enclosing the scene surface (cf. Fig. 3). We first partition the 3D space into the volume of 1 m³-sized, 0.5 m³-overlapped cubes; some of the cubes would be empty while others contain certain patches of the scene surface. We choose the centers of those empty cubes as the positions from which the virtual camera views the scene, and abandon others to ensure that the working distance between the camera and scene surface satisfy the aforementioned Kinect V2 requirements. From each position, the camera could view a certain patch of the scene surface towards arbitrary directions on a viewing sphere originated at the cube center. More specifically, for a center of empty cube positioned at $\mathbf{x} \in \mathbb{R}^3$, we make it homogeneous in the form of $\mathbf{t}_i = [\mathbf{x}; 1] \in \mathbb{R}^4$ and randomly sample 100 directions to get the camera extrinsics $\{\mathbf{K}_{ij} = [\mathbf{R}_j|\mathbf{t}_i]\}_{j=1}^{100}$, each of which would be used to generate a point cloud $\mathcal{P}_{\mathbf{K}_{ij}}$ covering a certain patch of the scene surface. Similar to object scanning, a final point cloud \mathcal{P} is obtained by registering and fusing all the point clouds obtained by the preceding scanning $\{\mathbf{K}_{ij} \circ \mathcal{P}_{\mathbf{K}_{ij}}\}_{i,j}$. Note that we do not conduct FPS as object-level synthetic scanning does; and consequently, the challenge of non-uniform distribution of points naturally appears here. Since some methods require surface normals, we compute the oriented normal \mathbf{n}_p for all the point $p \in \mathcal{P}$ in the same way as object scanning does. Apart from the natural challenges of non-uniform distribution and missing points (due to self-occlusion), we also simulate the other challenges of point-wise noise, point outliers, and misalignment in the same way as described in Section IV-A3. Specifically, following the settings of Kinect V2 camera, we set $\sigma_{\text{noise}} = 0.005$ m to control the level of point-wise noise, set $a_{\text{outlier}} = 0.01$ m, $b_{\text{outlier}} = 0.1$ m, and the ratio $r_{\text{outlier}} = 0.4\%$ to control the level of outliers, and set $[a_{\text{rotation}}, b_{\text{rotation}}]$ as $[-1.5^\circ, 1.5^\circ]$ and $[a_{\text{translation}}, b_{\text{translation}}]$ as $[-0.015$ m, 0.015 m] to control the level of misalignment. Such a point cloud \mathcal{P} contains around one million points containing all the five challenges.

C. The Real-Scanned Data

We provide real-scanned data of the benchmark by scanning real object instances via two depth cameras of varying precisions. To scan an object, we use SHINING 3D Einscan SE³ whose precision is of 100 micrometers to get the input point cloud, and use SHINING 3D OKIO 5 M⁴ whose precision is of 5 micrometers to get the approximate ground-truth.⁵

Data collection – We collect 20 object instances of varying surface complexities, including commodities, instruments, and artworks, and also of varying materials, including metal, plastic,

³<https://www.einscan.com/desktop-3d-scanners/einscan-se/>

⁴<https://www.shining3d.com/solutions/optimscan-5m>

⁵Our surface reconstruction evaluation is based on metrics of point set distances (cf. Section V-C), for which we obtain the ground-truth point clouds by sampling from the corresponding surface meshes. We thus choose to directly use the raw point clouds scanned by SHINING 3D OKIO 5 M, instead of converting the scanned point clouds as surface meshes using its in-built software.



Fig. 6. Examples of real-scanned object pairs.

ceramic, and cloth; this is to ensure that various sensing imperfections would appear in the obtained point clouds. Fig. 5 shows these objects.

Real scanning – In general, better scanning results could be obtained by increasing the numbers of scanning shots from multiple viewpoints, as verified in Fig. 4, where the Chamfer Distances (CD, cf. Appendix B for its definition, available online) decrease with more scan shots.

Since qualities of the scanned point clouds start to saturate at 40 shots for Einscan SE and 20 shots for OKIO 5 M, we conduct around 40 shots for Einscan SE and 20 shots for OKIO 5 M shots when scanning an object. After scanning an object, we use CloudCompare [141] to align different point clouds obtained from different shots. Table IV gives the statistics of our real-scanned data in the benchmark. Some scanned pairs from the two scanners are visualized in Fig. 6.

V. EXPERIMENTAL SET-UP FOR BENCHMARKING EXISTING SURFACE RECONSTRUCTION METHODS

With the benchmark prepared in Section IV, we aim to empirically compare existing surface reconstruction methods, and identify their advantages and disadvantages in different working conditions (e.g., various scanning imperfections considered in our benchmark). We expect such studies would both provide insights for future research in this area, and guide the use of appropriate methods for practical surface reconstruction from point observations. Such a comprehensive investigation could be timely for the community, given that a plethora of new methods have been proposed recently, in particular those based on deep learning.

A. Data

We conduct the empirical studies using a subset of our collected benchmark, while publicly releasing the whole benchmark to facilitate the community research. More specifically, we randomly sample 22 synthetic surfaces of object instances, whose distribution in the three groups of low-, middle-, and high-complexity is 12 : 6 : 4; as described in Section IV-A, for each instance we conduct six ways of synthetic scanning, which produce either a clean point cloud or point clouds with various

imperfections, giving rise to a total of 308 input-output pairs for benchmarking algorithms. We also use all the 50 synthetic scene surfaces and all the 20 real-scanned surfaces for the studies. For existing methods that use learning-based priors, we prepare an auxiliary set of training data consisting of surfaces from ShapeNet [18] and ABC [21], and also the remaining synthetic data of object instances from our benchmark. Specifically, we retrain OccNet [9] and DeepSDF [8] on the auxiliary set of data; the details are in Appendix C, available online.

B. Pre-Processing

While existing surface reconstruction methods can be compared using the data prepared in Section V-A, for most of them, their performance could be greatly improved via some standard pipeline of point cloud pre-processing. In this work, we compare existing methods both without and with such a pre-processing pipeline. For synthetic data, the pipeline is in the order of outlier removal, de-noising, and point re-sampling; details are given as follows. For real-scanned data, we use the inbuilt pre-processing of different scanners, and use a final step of FPS to re-sample 200,000 points for each point cloud.

Outlier removal – Performance of surface reconstruction degrades severely when extreme outliers exist in a point cloud; fortunately, these outliers are easy to be removed. We use a statistical method [142] to remove extreme point outliers. For a point cloud \mathcal{P} , it regards any $p \in \mathcal{P}$ as an outlier and remove it when p is very far away from its local neighborhood. More precisely, for any $p \in \mathcal{P}$, we first compute the averaged distance \bar{d}_p between p and its k nearest neighbors in \mathcal{P} ; we then compute the mean $m_{\bar{d}}$ and standard deviation $\sigma_{\bar{d}}$ of such distances for all $\{p \in \mathcal{P}\}$; a point p is regarded as an outlier when its corresponding $\bar{d}_p > 5 \cdot \sigma_{\bar{d}}$. We set $k = 35$ in this work for outlier removal.

De-noising – The inevitable existence of point-wise noise influences surface reconstruction as well. For an input point cloud \mathcal{P} , we choose to suppress such noise using Jets smoothing [143], which smoothes out the point cloud without sacrificing its surface curvatures. It works by first fitting a parametric surface patch to a local neighborhood \mathcal{N} of k points in \mathcal{P} , and then projecting $\{p \in \mathcal{N}\}$ onto the fitted surface patch. We set $k = 18$ in this work for point-wise de-noising.

Point re-sampling – Empirical results show that surface reconstruction benefits from more uniform distribution of points, even when reducing the number of points contained in \mathcal{P} [144]. For the synthetic data of object or scene surfaces, we simply use farthest point sampling [106] as the method to re-sample a more uniform distribution of points; we preserve 40% of original points during re-sampling.

C. Evaluation Metrics

We quantitatively compare reconstruction results from different methods using the popular metrics of Chamfer Distance (CD) [145], F-score [146], and Normal Consistency Score (NCS) [9]. Their computations are specified in Appendix B, available online. We also propose a neural metric, termed Neural Feature Similarity (NFS), focusing on perceptual consistency

between each reconstruction and the ground-truth; intuitively speaking, NFS compares the similarity of two shapes in the deep feature space, and thus depends more on the high-level semantic information consistent with human perception [147]. We present the details of NFS in Appendix B, available online, and compare it with a classical metric called Light Field Distance (LFD) [148]. Similar to LFD, NFS can capture the coarse-to-fine details of shapes; different to LFD, NFS can be used to evaluate all the three types of datasets in the benchmark, while LFD can only be used for the synthetic object-level surfaces (cf. Appendix B, available online).

CD and F-score are used for measuring the overall similarity between two shapes; NCS is more useful for measuring the nuance of two similar shapes by measuring their consistency of surface normals; NFS measures semantic difference related to human perception.

D. Methods and Implementation Details

In Section III-E, we have categorized existing methods according to what geometric priors they have respectively used to regularize the reconstruction. It is less feasible to study and empirically compare all the existing methods; instead, we take the strategy of selecting representative ones from each method group of geometric priors, assuming that our studies and conclusions would generalize in the same groups of existing methods. More specifically, we adopt the most representative GD [41] and BPA [42] as the methods to be studied for triangulation-based prior; for priors of surface smoothness, we adopt SPSR [39] using the first strategy of surface smoothness (cf. (10)) and RIMLS [63] using both two strategies of surface smoothness (cf. (9) and constraining \mathcal{H}_f); for modeling priors, we adopt SALD [83] that is able to reconstruct surfaces from un-oriented point clouds, and IGR [37] that can do so from oriented ones; for learning-based priors, we adopt the global, semantic learning methods of OccNet [9] and DeepSDF [8], and also the local learning methods of LIG [10] and Points2Surf [89]; we consider three methods that use hybrid priors, including DSE [123] that combines triangulation-based prior with learning-based prior, IMLSNet [124] that combines surface smoothness prior with learning-based prior, and ParseNet [47] that combines template-based prior with learning-based prior; we do not consider methods using template-based priors *only*, given that their performance largely depends on whether there would exist a good match between a surface to be reconstructed and the assumed templates (e.g., the assumed geometric primitives or the templates that can be retrieved in an auxiliary dataset), and that even the one with learning-based templates [47] fail to reconstruct surfaces of certain complexities.

Our implementations of the more classical, learning-free methods are based on established libraries; for example, we implement GD using the CGAL library [149], and directly execute BPA [42], SPSR [39], and RIMLS [63] with proper parameter tunings in MeshLab [150]. For those learning-based methods, we use the implementation codes publicly released by the authors when they are available, again with proper tuning of hyper-parameters for individual surfaces to be reconstructed.

For those without code releasing, we re-implement their algorithms and tune the respective hyper-parameters as the optimal ones.

In Section VI, we report and discuss the results *that are obtained with the pre-processing mentioned in Section V-B*. We note that those without pre-processing are of similar comparative qualities, and we put them in Appendix F, available online.

VI. MAIN RESULTS

We first summarize our key empirical findings, before presenting details of our series of experiments; insights are drawn subsequently.

- While many challenges of surface reconstruction from point clouds can be more or less tackled by using different regularization/priors of surface geometry, the challenges of *misalignment*, *missing points*, and *outliers* have been less addressed and remain unsolved.
- Data-driven solutions using deep learning have recently shown great promise for surface modeling and reconstruction, including their potential to deal with various data imperfections, however, our systematic experiments suggest that they struggle in generalizing to reconstruction of complex shapes; it is surprising that some classical methods such as SPSR [39] perform even better in terms of generalization and robustness.
- Use of surface normals is a key to success of surface reconstruction from raw, observed point clouds, even when the normals are estimated less accurately; in many cases, the reconstruction result improves as long as the interior and exterior of a surface can be identified in the 3D space.
- There exist inconsistencies between different evaluation metrics, and in many cases, good quantitative results do not translate as visually pleasant ones. For example, quantitative results measured by CD and F-score are not much affected by the challenge of misalignment; however, the reduced scores of NCS and NFS suggest that the recovered surfaces might be less pleasant to human perception.

Quantitative results of the comparative methods are given in Tables V, VI, and VII, which are respectively for synthetic data of object surfaces, synthetic data of scene surfaces, and real-scanned data. Qualitative results are presented in the following sections accompanying our discussions.

A. The Remaining Challenges

For ease of analysis, we plot in Fig. 7 the quantitative results in Table V under the metrics of CD, F-score, NCS, and NFS. As presented in Section V-C, the four metrics focus on different measure perspectives. By diagnosing the comparative methods using these measures and investigating their capabilities to cope with different challenges of imperfect scanning, Fig. 7 helps in identifying the remaining challenges.

Fig. 7 shows that, under all the metrics, the challenge of *non-uniform distribution of points* is relatively easy to be tackled by almost all the methods, except those learning semantics or geometric primitives (i.e., DeepSDF [8], OccNet [9], and ParseNet [47]), achieving similar results as those on data of

TABLE V
QUANTITATIVE RESULTS ON THE TESTING SYNTHETIC DATA OF OBJECT SURFACES

Prior	Method	CD ($\times 10^{-4}$) ↓						F-score (%) ↑					
		Perfect scanning	Non-uniform distribution	Point-wise noise	Point outliers	Missing points	Mis-alignment	Perfect scanning	Non-uniform distribution	Point-wise noise	Point outliers	Missing points	Mis-alignment
Triangulation-based	GD [41]	14.24	14.87	18.20	123.19	64.15	21.15	99.66	99.07	98.91	54.59	87.72	97.41
	BPA [42]	14.89	17.13	18.58	14.66	62.55	20.88	98.70	97.02	98.51	99.31	87.85	97.24
Smoothness	SPSR [39]	14.47	15.36	16.05	14.71	225.66	17.24	99.59	99.02	99.46	99.65	76.91	99.27
	RIMLS [63]	15.73	16.74	17.17	126.40	65.36	21.12	99.27	98.76	99.24	57.78	87.92	97.53
Modeling	SALD [83]	15.10	14.96	18.77	53.65	55.63	20.09	99.45	99.10	98.94	88.05	85.78	98.09
	IGR [37]	18.40	18.33	18.57	43.60	151.53	20.72	97.54	97.79	97.75	91.11	70.49	96.46
Learning Semantics	OccNet [9]	201.96	210.80	205.21	225.85	231.65	212.51	31.03	29.90	29.77	23.75	24.52	29.19
	DeepSDF [8]	229.18	227.42	230.40	511.36	378.58	232.16	17.79	18.81	17.08	4.15	14.06	17.05
Local Learning	LIG [10]	23.09	24.03	22.05	115.38	70.98	24.30	96.20	95.32	96.99	76.69	82.43	95.01
	Points2Surf [89]	17.18	18.81	18.48	83.91	102.18	20.36	98.14	97.72	97.39	72.94	76.46	96.49
Hybrid	DSE [123]	14.26	15.34	17.89	100.37	68.88	20.06	99.64	98.84	99.17	52.21	87.71	98.20
	IMLSNet [124]	22.56	23.17	22.67	99.95	74.35	23.77	94.82	94.51	94.36	64.55	80.01	94.14
	ParseNet [47]	162.94	161.14	135.84	176.38	195.98	136.86	40.52	38.82	44.13	37.21	41.28	46.60
Prior	Method	NCS ($\times 10^{-2}$) ↑						NFS ($\times 10^{-2}$) ↑					
Perfect scanning	Non-uniform distribution	Point-wise noise	Point outliers	Missing points	Mis-alignment	Perfect scanning	Non-uniform distribution	Point-wise noise	Point outliers	Missing points	Mis-alignment	Perfect scanning	Non-uniform distribution
Triangulation-based	GD [41]	98.57	98.05	87.58	92.17	95.17	84.96	95.22	95.19	84.63	37.47	82.93	80.69
	BPA [42]	98.37	97.89	91.68	98.07	94.42	90.12	94.10	93.60	87.49	93.24	80.36	81.96
Smoothness	SPSR [39]	98.58	98.38	97.03	98.56	89.99	96.24	96.38	96.22	94.98	96.31	69.34	94.28
	RIMLS [63]	98.19	97.77	95.23	83.42	93.27	92.48	95.01	94.02	92.67	62.01	79.23	87.12
Modeling	SALD [83]	98.67	98.52	96.42	95.32	95.19	94.73	96.11	96.65	89.72	51.74	84.01	88.26
	IGR [37]	97.62	97.59	97.52	96.47	90.61	97.04	94.71	94.22	94.52	84.63	68.14	92.54
Learning Semantics	OccNet [9]	79.55	79.30	79.64	78.55	78.43	79.58	47.33	46.55	46.03	42.11	42.46	45.80
	DeepSDF [8]	78.65	79.11	77.78	73.40	74.52	78.12	39.94	40.91	39.29	16.65	31.59	39.26
Local Learning	LIG [10]	96.49	95.79	94.22	91.66	89.56	92.70	91.58	90.66	89.55	66.21	74.98	84.34
	Points2Surf [89]	95.24	95.09	94.62	87.87	86.36	94.48	93.45	93.23	92.59	63.30	68.53	91.59
Hybrid	DSE [123]	98.60	97.86	87.79	77.34	94.40	86.20	94.50	94.75	83.53	42.32	79.62	76.63
	IMLSNet [124]	96.13	95.98	96.02	87.45	90.48	95.87	90.61	90.20	89.97	52.82	74.59	89.19
	ParseNet [47]	77.71	76.89	80.31	75.46	75.83	80.48	38.54	37.71	49.30	35.98	38.40	45.73

Comparisons are made on data of perfect scanning and those of all the five challenges of imperfect scanning specified in Section 4.1.3; for those challenges with varying levels of severity, we use the data of middle-level severity for the comparison (cf. Appendix D for the overall results). Results of the best and second best methods are highlighted in each column. Comparative methods are also grouped according to what priors of surface geometry they have used (cf. Section 3 for the grouping and Section 5.4 for how these representative methods are selected). “-” indicates the method cannot produce reasonable results due to their limited generalization.

TABLE VI
QUANTITATIVE RESULTS ON THE TESTING SYNTHETIC DATA OF SCENE SURFACES

Prior	Method	CD ($\times 10^{-3}$) ↓	F-score (%) ↑	NCS ($\times 10^{-2}$) ↑	NFS ($\times 10^{-2}$) ↑
Triangulation-based	GD [41]	51.54	58.89	57.81	37.74
	BPA [42]	77.97	33.36	54.41	33.28
Smoothness	SPSR [39]	49.46	64.41	74.23	47.82
	RIMLS [63]	67.46	53.61	58.16	27.37
Modeling	SALD [83]	50.42	61.14	86.47	45.98
	IGR [37]	50.06	62.67	84.36	46.31
Learning Semantics	OccNet [9]	108.29	29.16	79.48	35.08
	DeepSDF [8]	-	-	-	-
Local Learning	LIG [10]	69.38	57.57	79.24	43.92
	Points2Surf [89]	66.22	56.27	75.53	45.64
Hybrid	DSE [123]	51.72	57.49	55.44	37.64
	IMLSNet [124]	59.95	59.88	80.80	44.86
	ParseNet [47]	-	-	-	-

Results of the best and second best methods are highlighted in each column. Comparative methods are grouped according to what priors of surface geometry they have used (cf. Section 3 for the grouping and Section 5.4 for how these representative methods are selected). “-” indicates the method cannot produce reasonable results due to their limited generalization.

TABLE VII
QUANTITATIVE RESULTS ON THE REAL-SCANNED DATA

Prior	Method	CD ($\times 10^{-2}$) ↓	F-score (%) ↑	NCS ($\times 10^{-2}$) ↑	NFS ($\times 10^{-2}$) ↑
Triangulation-based	GD [41]	31.72	87.51	88.86	82.20
	BPA [42]	40.37	80.95	87.56	68.69
Smoothness	SPSR [39]	31.05	87.74	94.94	89.38
	RIMLS [63]	32.80	87.05	91.97	85.19
Modeling	SALD [83]	31.13	87.72	94.68	86.86
	IGR [37]	32.70	87.18	95.99	89.10
Learning Semantics	OccNet [9]	232.71	17.11	80.96	39.70
	DeepSDF [8]	263.92	19.83	77.95	40.95
Local Learning	LIG [10]	48.75	83.76	92.57	81.48
	Points2Surf [89]	48.93	80.89	89.52	81.83
Hybrid	DSE [123]	32.16	86.88	87.20	76.81
	IMLSNet [124]	38.46	82.44	93.31	85.30
	ParseNet [47]	149.96	38.92	81.51	45.67

Results of the best and second best methods are highlighted in each column. Comparative methods are grouped according to what priors of surface geometry they have used (cf. Section 3 for the grouping and Section 5.4 for how these representative methods are selected).

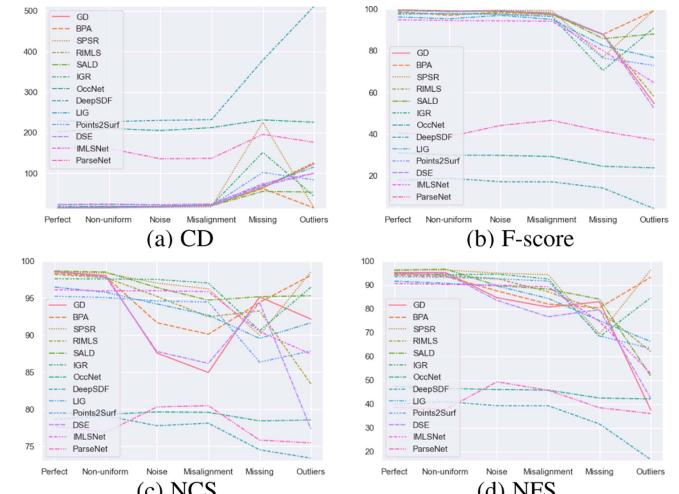


Fig. 7. Plotting of quantitative results in Table V.

verified by the reduced scores of NCS and NFS, especially for triangulation-based methods such as GD [41], BPA [42], and the hybrid one of DSE [123] that combines the triangulation-based prior. This is intuitive since methods based on triangulation of points rely heavily on cleanliness of input points.

There exists a similar but severer phenomenon for the challenge of *misalignment*. As indicated by Fig. 7, when measured by CD and F-score, most of the methods give reasonably good results, suggesting that the overall shape structures have been recovered. However, the reduced scores of NCS and NFS suggest that some of the recovered surfaces might be less pleasant to human perception. In fact, as shown by the example in Fig. 8, most methods fail in reconstructing the surface on the misaligned areas, generating thickened or even multiple layers of the local surface. Misalignment is a practical issue in 3D scanning, especially when using hand-held, consumer scanners. Figs. 7

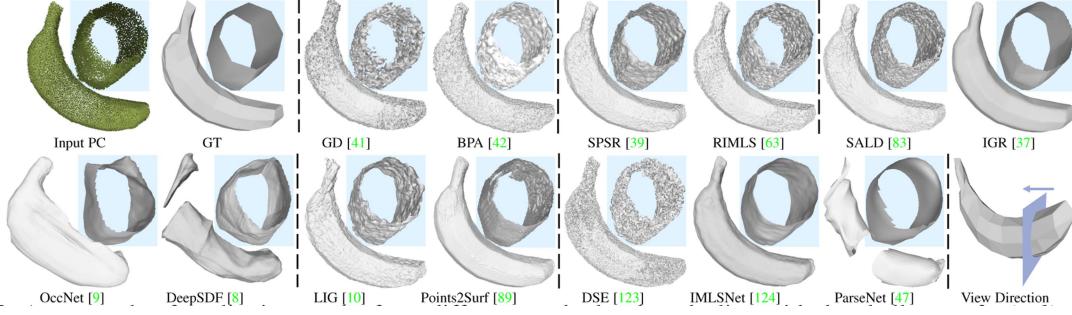


Fig. 8. An example of qualitative results from different methods when dealing with the challenge of *misalignment*.

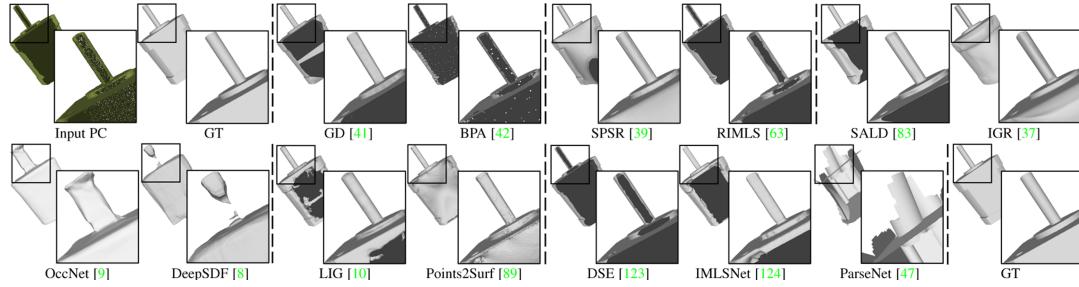


Fig. 9. An example of qualitative results from different methods when dealing with the challenge of *missing points*.

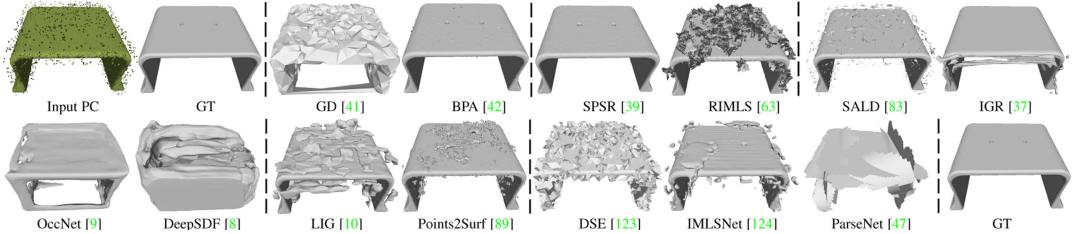


Fig. 10. An example of qualitative results from different methods when dealing with the challenge of *point outliers*.

and 8 show that methods using smoothness and/or modeling priors have the advantage in handling misalignment.

Fig. 7 also shows that the two challenges of *missing points* and *point outliers* are much more difficult to be handled. When an input point cloud has missing points, as shown in Fig. 9, most of the methods ignore reconstruction of the missing surface areas, resulting in incomplete surfaces. The implicit methods (e.g., IGR [37], OccNet [9], and Points2Surf [89]) tend to generate watertight surfaces by filling the holes with concave/convex hulls; however, such envelopes may not represent the surface correctly, possibly making the reconstruction even poorer under all the evaluation metrics (cf. Fig. 7).

As for *point outliers*, although a pre-processing step of outlier removal has already been adopted for all the comparative methods (cf. Section V-B), performance of different methods still varies drastically. Fig. 10 gives an example; the results depend on whether the respective methods have their inbuilt mechanisms of outlier removal. For example, BPA [42] requires the sizes of its triangular faces to satisfy certain conditions, making it naturally suitable for handling outliers; methods using a global implicit field (e.g., SPSR [39] and IGR [37]) ignore the outliers

implicitly, and are thus capable of handling point outliers as well.

Summarizing the above analyses gives us the following empirical findings: 1) the challenge of *missing points* remains unsolved by all the comparative methods; 2) for the challenges of *misalignment* and *point outliers*, most of the methods (except few ones such as SPSR [39]) give unsatisfactory results; 3) there exist inconsistencies between different evaluation metrics, and in many cases, good quantitative results do not translate as visually pleasant ones; 4) methods that learn semantics or pre-defined shape patterns may fail to generalize even on clean data of perfect scanning, when the testing point clouds do not fall in the learned data domains; we will discuss more on this issue shortly.

B. Optimization-Based, Learning-Free Methods versus Learning-Based, Data-Driven Ones

In this section, we investigate the behaviors of comparative methods by organizing them into two groups of *optimization-based*, *learning-free methods* and *learning-based, data-driven ones*. The former group includes SPSR [39], RIMLS [63],

TABLE VIII
COMPARISON BETWEEN OPTIMIZATION-BASED, LEARNING-FREE METHODS AND LEARNING-BASED, DATA-DRIVEN METHODS ON THE TESTING SYNTHETIC DATA OF OBJECT SURFACES

Algorithms	Priors	CD ($\times 10^{-4}$) \downarrow	F-score (%) \uparrow	NCS ($\times 10^{-2}$) \uparrow	NFS ($\times 10^{-2}$) \uparrow
		low- /middle- / high-level	low- /middle- / high-level	low- /middle- / high-level	low- /middle- / high-level
GD [41]	learning-free	16.52 / 1.68 / 13.21	99.09 / -0.18 / -7.44	94.00 / -6.42 / -25.68	91.57 / -6.94 / -35.27
		16.59 / 1.99 / 12.71	98.63 / -0.12 / -9.04	95.36 / -3.68 / -17.39	90.71 / -3.22 / -25.56
		15.50 / 0.55 / 2.66	99.51 / -0.05 / -0.35	97.84 / -0.81 / -3.95	95.60 / -0.62 / -3.61
		16.13 / 1.04 / 9.75	99.36 / -0.12 / -4.67	97.36 / -2.13 / -11.23	94.19 / -1.52 / -14.43
		15.33 / 3.44 / 12.26	99.54 / -0.60 / -7.06	98.07 / -1.65 / -9.16	95.66 / -5.94 / -28.48
		18.21 / 0.36 / 0.99	97.87 / -0.12 / -0.35	97.64 / -0.12 / -0.54	94.37 / 0.15 / -0.70
		209.04 / -3.83 / 0.97	29.85 / -0.08 / -1.93	79.43 / 0.21 / -0.38	46.17 / -0.14 / -1.13
DeepSDF [8]	learning-based	241.28 / -10.88 / -1.65	16.70 / 0.38 / -0.71	78.01 / -0.23 / 0.51	38.55 / 0.74 / 0.15
		23.96 / -1.91 / 2.31	94.50 / 2.49 / -0.70	93.96 / 0.26 / -5.70	86.71 / 2.84 / -5.68
		17.74 / 0.74 / 4.89	98.02 / -0.63 / -3.81	94.97 / -0.35 / -1.53	92.83 / -0.24 / -2.62
		16.07 / 1.82 / 11.54	99.40 / -0.23 / -6.65	94.43 / -6.64 / -23.04	90.05 / -6.52 / -32.97
		22.64 / 0.03 / 1.09	94.41 / -0.05 / -0.01	96.08 / -0.06 / -0.46	90.13 / -0.16 / -0.47
		154.11 / -18.27 / -14.14	44.80 / -0.67 / -2.14	78.55 / 1.76 / 2.60	46.21 / 3.09 / 2.79

We report quantitative results for the imperfect scanning of point-wise noise at three levels of severity; results are in the format of “ $\cdot\cdot\cdot / \cdot\cdot\cdot / \cdot\cdot\cdot$ ”, where the most left one is the absolute value under each evaluation metric, and the right two ones are those relative to the most left one. The best and second best methods are highlighted in each column.

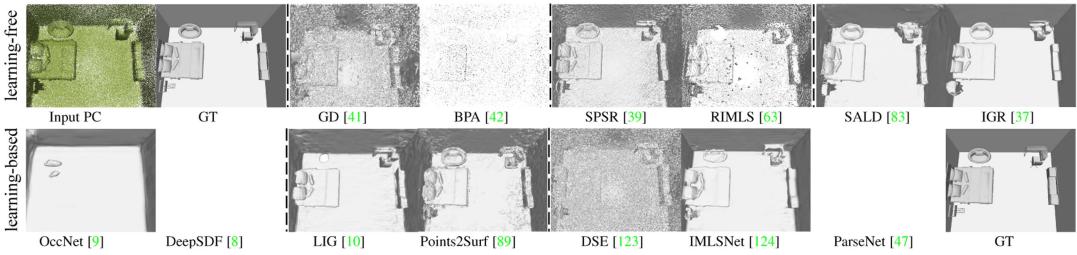


Fig. 11. Qualitative results respectively from optimization-based, learning-free methods and learning-based, data-driven methods on the testing synthetic data of scene surfaces. The scene is a bedroom instance from the 3D-FRONT [25].

SALD [83], IGR [37], and we also include GD [41] and BPA [42] into the group for a complete coverage of the studied methods; the latter group includes OccNet [9], DeepSDF [8], LIG [10], Points2Surf [89], DSE [123], IMLSNet [124], and ParseNet [47]. By doing so, we aim to investigate how the two groups perform in terms of generalizing to complex shapes, where we pay special attention to methods of global, semantic learning (i.e., OccNet [9] and DeepSDF [8]) whose advantages may only be manifested when the categories of object surfaces exist in the auxiliary training set. Note that our testing data of synthetic object surfaces include 22 randomly selected objects, as described in Section V-A, which contain both those belonging to popular semantic categories (e.g., *chair*) and those without clearly defined semantics; for the former case, our auxiliary training set contains rich shape instances of same categories from ShapeNet [18]. In this section, we also study robustness of the two groups of methods against imperfect scanning at varying levels of severity.

Quantitative results in Table VIII show that on reconstruction of synthetic object surfaces, optimization-based, learning-free methods generalize better under different evaluation metrics, since our testing shapes contain both semantic ones and non-semantic, complex ones; we have consistent observations from examples in Fig. 12 — while learning-based, data-driven methods are good at reconstructing an *chair* surface, they fail in generalizing to non-semantic shapes. Table VIII also shows

that learning-based methods are good in terms of robustness against higher levels of data imperfections. This suggests that learning-based methods perform better when the input provides less information, e.g., when the input cloud of points is getting sparser, as discussed in [11]. We observe similar phenomena on reconstruction of real-scanned data (cf. Appendix G for more details, available online), by re-organizing Table VII according to the two method groups.

To further investigate the behaviors of learning from auxiliary data, we conduct experiments of synthetic scene surface reconstruction. Results in Table VI (after re-organization of method groups) and Fig. 11 show that among the learning-based methods, LIG [10], Points2Surf [89] and IMLSNet [124] are capable of handling reconstruction of scene-level surfaces via local modeling and aggregation, while semantic learning methods of OccNet and DeepSDF fail, as expected.

We summarize the above analyses as follows: 1) learning-based, data-driven methods are able to reconstruct object surfaces when the training set contains object instances of the same semantic categories, and they show a certain degree of robustness against data imperfections; however, these methods fail to generalize when the condition is not satisfied; 2) for reconstruction of scene-level surfaces, local learning methods succeed by local modeling and aggregation, and in contrast, global, semantic learning methods fail to do so; 3) some optimization-based, learning-free methods (e.g., SPSR [39]) perform surprisingly

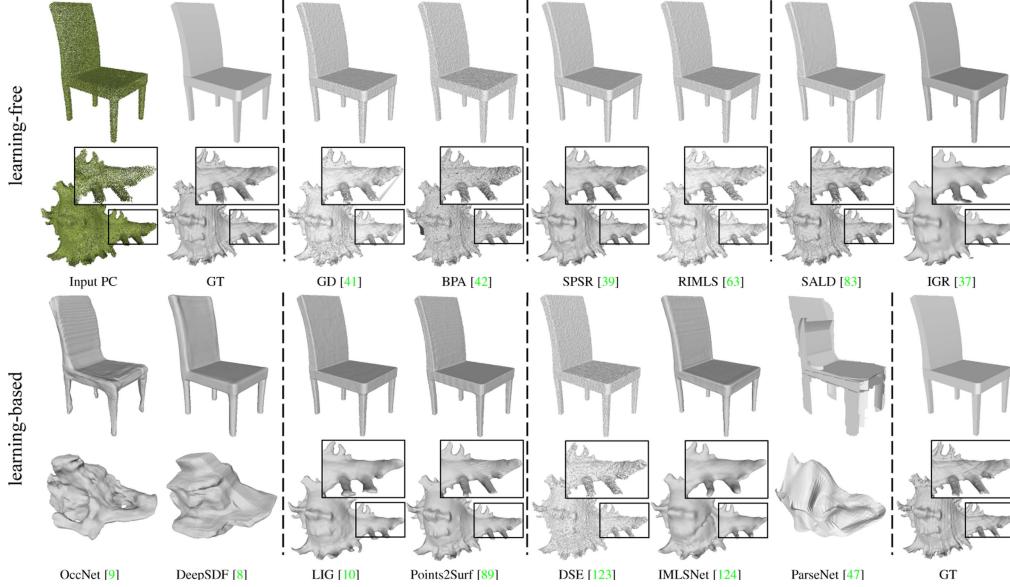


Fig. 12. Qualitative results respectively from optimization-based, learning-free methods and learning-based, data-driven methods on the testing synthetic data of object surfaces. The input point clouds have imperfect scanning of *point-wise noise*. The example of *chair* is of popular semantic categories, for which our auxiliary training set contains rich instances of the same category from ShapeNet [18]; the other one is non-semantic.

well in robustness and generalization for both object-level and scene-level surfaces.

C. The Importance of Orientations of Surface Normals

Some of our studied methods compute surface normals from observed point clouds, and use the computed normals for surface reconstruction, including BPA [42], SPSR [39], RIMLS [63], IGR [37], OccNet [9], DeepSDF [8], LIG [10], and ParseNet [47]; a few other methods (e.g., Points2Surf [89] and IMLSNet [124]) compute surface normals on training data, and then train models to estimate surface normals when reconstructing testing surfaces. To investigate how these methods benefit from surface normal computation/estimation, we conduct experiments on our testing data with scanning imperfections, since robustness of these methods would be tested when less accurate surface normals are computed from imperfectly scanned data. Assume that the relative pose of a camera w.r.t. an observed point cloud is given; for any observed point, we compute its *oriented* surface normal by performing PCA on its local neighborhood of points (cf. Section IV-A3 for the details); as such, the computed surface normals may not be precise but their *inward or outward orientations* must be correct.

Table IX gives the quantitative results for synthetic data of object surfaces; under different evaluation metrics, the additional computation or estimation of surface normals does help in improving surface reconstruction. When the relative pose of camera w.r.t. observed surface points is not available, the computed surface normals would be *wrongly oriented* at some local surface neighborhoods, since principal directions of PCA on local neighborhoods of noisy points are less reliable. To investigate the importance of inward or outward orientations of surface normals, we also report experiments in Table IX where

TABLE IX
COMPARISON ON TESTING SYNTHETIC DATA OF OBJECT SURFACES AMONG METHODS WITHOUT USING SURFACE NORMALS (\times), METHODS USING SURFACES NORMALS (\checkmark), AND METHODS USING SURFACE NORMALS ONLY DURING LEARNING (*)

Algorithms	Normals	CD ($\times 10^{-4}$) \downarrow	F-score (%) \uparrow	NCS ($\times 10^{-2}$) \uparrow	NFS ($\times 10^{-2}$) \uparrow
GD [41]	\times	18.20 / 18.61	98.91 / 98.56	87.58 / 87.58	84.63 / 84.63
SALD [83]	\times	18.77 / 17.89	98.94 / 99.17	96.42 / 87.79	89.72 / 83.53
BPA [42]		18.58 / 18.61	98.51 / 98.56	91.68 / 91.79	87.49 / 85.08
SPSR [39]		16.05 / 60.42	99.46 / 91.45	97.03 / 94.68	94.98 / 84.34
RIMLS [63]		17.17 / 18.58	99.24 / 97.47	95.23 / 94.06	92.67 / 86.63
IGR [37]	\checkmark	18.57 / 262.40	97.75 / 85.98	97.52 / 95.28	94.52 / 79.61
OccNet [9]		205.21 / 214.12	29.77 / 28.83	79.64 / 79.01	46.03 / 45.12
DeepSDF [8]		230.40 / 569.41	17.08 / 15.64	77.78 / 77.09	39.29 / 33.78
LIG [10]		22.05 / 34.78	96.99 / 89.60	94.22 / 91.36	89.55 / 80.02
ParseNet [47]		135.84 / 197.28	44.13 / 35.67	80.31 / 76.19	49.30 / 41.65
Points2Surf [89]	\ast	18.48 / 22.67	97.39 / 94.36	94.62 / 96.02	92.59 / 89.97
IMLSNet [124]					

We report quantitative results for the imperfect scanning of point-wise noise at the middle level of severity; results are in the format of “/”, where the left one is obtained assuming the availability of ground-truth camera poses, and the right one is obtained without knowing the camera poses. The best and second best methods are highlighted in each column.

each result on the right of the “/” symbol is obtained without knowing the relative camera poses; compared with results on the left side of “/” that are obtained assuming the ground-truth camera poses, results on the right side drops drastically. We also conduct experiments on our synthetic data of scene surfaces and real-scanned data, and observe similar phenomena; these results are presented in Appendix E, available online.

We have the following empirical findings based on the above analyses: 1) computation or estimation of oriented surface normals help in surface reconstruction from point clouds; 2) compared with precisions of surface normals, it is more important to have the correct inward or outward orientations of surface normals.

VII. CONCLUSION

In this paper, we have reviewed both the classical and the more recent deep learning-based methods for surface reconstruction from point clouds; we have organized our reviews by

categorizing these methods according to what priors of surface geometry they had used to regularize their solutions. To better understand the respective strengths and limitations of existing methods, we contribute a large-scale benchmarking dataset consisting of both synthetic and real-scanned data, which provides various sensing imperfections that are commonly encountered in practical 3D scanning. We conduct thorough empirical studies on the constructed benchmark, evaluating the robustness and generalization of different methods. Our studies help identify the remaining challenges faced by existing methods, and we expect that our studies would be useful for guiding the directions in future research.

REFERENCES

- [1] R. Bolle and B. Vemuri, "On three-dimensional surface reconstruction methods," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 1, pp. 1–13, Jan. 1991.
- [2] S. P. Lim and H. Haron, "Surface reconstruction techniques: A review," *Artif. Intell. Rev.*, vol. 42, no. 1, pp. 59–78, 2014.
- [3] M. Berger et al., "A survey of surface reconstruction from point clouds," *Comput. Graph. Forum*, vol. 36, no. 1, pp. 301–329, 2017.
- [4] 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 *Proc. IEEE Conf. Open Syst.*, 2020, pp. 37–42.
- [5] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry, "A Papier-Mâché approach to learning 3D surface generation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 216–224.
- [6] Y. Luo, Z. Mi, and W. Tao, "DeepDT: Learning geometry from delaunay triangulation for surface reconstruction," in *Proc. 35th AAAI Conf. Artif. Intell.*, 2021, pp. 2277–2285.
- [7] N. Sharp and M. Ovsjanikov, "PointTriNet: Learned triangulation of 3D point sets," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2020, pp. 762–778.
- [8] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "DeepSDF: Learning continuous signed distance functions for shape representation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 165–174.
- [9] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy networks: Learning 3D reconstruction in function space," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4460–4470.
- [10] C. Jiang, A. Sud, A. Makadia, J. Huang, M. Nießner, and T. Funkhouser, "Local implicit grid representations for 3D scenes," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 6001–6010.
- [11] R. Sulzer, L. Landrieu, R. Marlet, and B. Vallet, "A survey and benchmark of automatic surface reconstruction from point clouds," 2023, *arXiv:2301.13656*.
- [12] G. Slabaugh, R. Schafer, T. Malzbender, and B. Culbertson, "A survey of methods for volumetric scene reconstruction from photographs," in *Proc. Volume Graph.*, Springer, 2001, pp. 81–100.
- [13] F. Remondino and S. El-Hakim, "Image-based 3D modelling: A review," *Photogrammetric Rec.*, vol. 21, no. 115, pp. 269–291, 2006.
- [14] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, "A comparison and evaluation of multi-view stereo reconstruction algorithms," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2006, pp. 519–528.
- [15] X. Han, H. Laga, and M. Bennamoun, "Image-based 3D object reconstruction: State-of-the-art and trends in the deep learning era," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 5, pp. 1578–1604, May 2021.
- [16] G. Fahim, K. Amin, and S. Zarif, "Single-view 3D reconstruction: A survey of deep learning methods," *Comput. Graph.*, vol. 94, pp. 164–190, 2021.
- [17] H. Zhu, Y. Nie, T. Yue, and X. Cao, "The role of prior in image based 3D modeling: A survey," *Front. Comput. Sci.*, vol. 11, no. 2, pp. 175–191, 2017.
- [18] A. X. Chang et al., "ShapeNet: An information-rich 3D model repository," 2015, *arXiv:1512.03012*.
- [19] Z. Wu et al., "3D ShapeNets: A deep representation for volumetric shapes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1912–1920.
- [20] W. Wohlkinger, A. Aldoma, R. B. Rusu, and M. Vincze, "3DNet: Large-scale object class recognition from CAD models," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2012, pp. 5384–5391.
- [21] S. Koch et al., "ABC: A big CAD model dataset for geometric deep learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 9601–9611.
- [22] Q. Zhou and A. Jacobson, "Thingi10K: A dataset of 10, 000 3D-printing models," 2016, *arXiv:1605.04797*.
- [23] V. Albertina et al., "Three D scans," 2012. [Online]. Available: <https://threescans.com/>
- [24] A. Handa, V. Pătrăucean, S. Stent, and R. Cipolla, "SceneNet: An annotated model generator for indoor scene understanding," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 5737–5743.
- [25] H. Fu et al., "3D-front: 3D furnished rooms with layouts and semantics," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 10933–10942.
- [26] A. Singh, J. Sha, K. S. Narayan, T. Achim, and P. Abbeel, "BigBIRD: A large-scale 3D database of object instances," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2014, pp. 509–516.
- [27] B. Calli et al., "Yale-CMU-Berkeley dataset for robotic manipulation research," *Int. J. Robot. Res.*, vol. 36, no. 3, pp. 261–268, 2017.
- [28] A. Kasper, Z. Xue, and R. Dillmann, "The kit object models database: An object model database for object recognition, localization and manipulation in service robotics," *Int. J. Robot. Res.*, vol. 31, no. 8, pp. 927–934, 2012.
- [29] J. Straub et al., "The replica dataset: A digital replica of indoor spaces," 2019, *arXiv:1906.05797*.
- [30] A. Chang et al., "Matterport3D: Learning from RGB-D data in indoor environments," *Int. Conf. 3D Vis.*, 2017, pp. 667–676.
- [31] M. Berger, J. A. Levine, L. G. Nonato, G. Taubin, and C. T. Silva, "A benchmark for surface reconstruction," *ACM Trans. Graph.*, vol. 32, no. 2, pp. 1–17, 2013.
- [32] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, and B. Lévy, *Polygon Mesh Processing*. Boca Raton, FL, USA: AK Peters / CRC Press, Sep. 2010.
- [33] E. Tretschk, A. Tewari, V. Golyanik, M. Zollhöfer, C. Stoll, and C. Theobalt, "PatchNets: Patch-based generalizable deep implicit 3D shape representations," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2020, pp. 293–309.
- [34] R. Chabra et al., "Deep local shapes: Learning local SDF priors for detailed 3D reconstruction," in *Proc. Eur. Conf. Comput. Vis.*, Cham: Springer International Publishing, 2020, pp. 608–625.
- [35] S. Peng, M. Niemeyer, L. Mescheder, M. Pollefeys, and A. Geiger, "Convolutional occupancy networks," in *Proc. Eur. Conf. Comput. Vis.*, Springer International Publishing, 2020, pp. 523–540.
- [36] J. Tang, J. Lei, D. Xu, F. Ma, K. Jia, and L. Zhang, "SA-ConvONet: Sign-agnostic optimization of convolutional occupancy networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 6504–6513.
- [37] A. Gropp, L. Yariv, N. Haim, M. Atzmon, and Y. Lipman, "Implicit geometric regularization for learning shapes," in *Proc. Mach. Learn. Syst.*, 2020, pp. 3569–3579.
- [38] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson surface reconstruction," in *Proc. 4th Eurograph. Symp. Geometry Process.*, 2006, pp. 61–70.
- [39] M. Kazhdan and H. Hoppe, "Screened Poisson surface reconstruction," *ACM Trans. Graph.*, vol. 32, no. 3, pp. 1–13, 2013.
- [40] H. Edelsbrunner and N. R. Shah, "Triangulating topological spaces," in *Proc. 10th Annu. Symp. Comput. Geometry*, 1994, pp. 285–292.
- [41] D. Cohen-Steiner and F. Da, "A greedy delaunay-based surface reconstruction algorithm," *Vis. Comput.*, vol. 20, no. 1, pp. 4–16, 2004.
- [42] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin, "The ball-pivoting algorithm for surface reconstruction," *IEEE Trans. Vis. Comput. Graphics*, vol. 5, no. 4, pp. 349–359, Fourth Quarter 1999.
- [43] D. Levin, "Mesh-independent surface interpolation," in *Proc. Geometric Model. Sci. Vis.*, Berlin, Germany: Springer, 2004, pp. 37–49.
- [44] S. Peng, C. M. Jiang, Y. Liao, M. Niemeyer, M. Pollefeys, and A. Geiger, "Shape as points: A differentiable poisson solver," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 13032–13044.
- [45] M. Eck and H. Hoppe, "Automatic reconstruction of b-spline surfaces of arbitrary topological type," in *Proc. 23rd Annu. Conf. Comput. Graph. Interactive Techn.*, 1996, pp. 325–334.
- [46] Y. He and H. Qin, "Surface reconstruction with triangular b-splines," in *Proc. IEEE Geometric Model. Process.*, 2004, pp. 279–287.
- [47] G. Sharma, D. Liu, S. Maji, E. Kalogerakis, S. Chaudhuri, and R. Měch, "ParSeNet: A parametric surface fitting network for 3D point clouds," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 261–276.
- [48] N. Leal, E. Leal, and J. W. Branch, "Simple method for constructing NURBS surfaces from unorganized points," in *Proc. 19th Int. Meshing Roundtable*, Springer, 2010, pp. 161–175.

- [49] A. Hashemian and S. F. Hosseini, "An integrated fitting and fairing approach for object reconstruction using smooth NURBS curves and surfaces," *Comput. Math. Appl.*, vol. 76, no. 7, pp. 1555–1575, 2018.
- [50] R. Brüel-Gabrielsson, V. Ganapathi-Subramanian, P. Skraba, and L. J. Guibas, "Topology-aware surface reconstruction for point clouds," *Comput. Graph. Forum*, vol. 39, no. 5, pp. 197–207, 2020.
- [51] D. Wang, "An efficient iterative method for reconstructing surface from point clouds," *J. Sci. Comput.*, vol. 87, no. 1, 2021, Art. no. 38.
- [52] S. Lin, D. Xiao, Z. Shi, and B. Wang, "Surface reconstruction from point clouds without normals by parametrizing the Gauss formula," *ACM Trans. Graph.*, vol. 42, no. 2, pp. 1–19, 2022.
- [53] J. C. Carr et al., "Reconstruction and representation of 3D objects with radial basis functions," in *Proc. 28th Annu. Conf. Comput. Graph. Interactive Techn.*, 2001, pp. 67–76.
- [54] M. Kazhdan, "Reconstruction of solid models from oriented point sets," in *Proc. 3rd Eurographics Symp. Geometry Process.*, 2005, pp. 73–es.
- [55] J. Manson, G. Petrova, and S. Schaefer, "Streaming surface reconstruction using wavelets," *Comput. Graph. Forum*, vol. 27, no. 5, pp. 1411–1420, 2008.
- [56] Y. He, S. H. Kang, and H. Liu, "Curvature regularized surface reconstruction from point clouds," *SIAM J. Imag. Sci.*, vol. 13, no. 4, pp. 1834–1859, 2020.
- [57] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva, "Point set surfaces," in *Proc. Conf. Visual.*, 2001, pp. 21–28.
- [58] A. Adamson and M. Alexa, "Approximating and intersecting surfaces from points," in *Proc. Eurographics/ACM SIGGRAPH Symp. Geometry Process.*, 2003, pp. 230–239.
- [59] G. Guennebaud and M. Gross, "Algebraic point set surfaces," *ACM Trans. Graph.*, vol. 26, no. 3, pp. 23–es, Jul. 2007.
- [60] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva, "Computing and rendering point set surfaces," *IEEE Trans. Vis. Comput. Graphics*, vol. 9, no. 1, pp. 3–15, First Quarter, 2003.
- [61] S. Fleishman, D. Cohen-Or, and C. T. Silva, "Robust moving least-squares fitting with sharp features," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 544–552, 2005.
- [62] R. Kolluri, "Provably good moving least squares," *ACM Trans. Algorithms*, vol. 4, no. 2, pp. 1–25, 2008.
- [63] A. C. Öztireli, G. Guennebaud, and M. Gross, "Feature preserving point set surfaces based on non-linear kernel regression," *Comput. Graph. Forum*, vol. 28, no. 2, pp. 493–501, 2009.
- [64] Z.-Q. Cheng, Y.-Z. Wang, B. Li, K. Xu, G. Dang, and S.-Y. Jin, "A survey of methods for moving least squares surfaces," in *Proc. 5th Eurograph/IEEE VGTC Conf. Point-Based Graph.*, 2008, pp. 9–23.
- [65] R. Schnabel, R. Wahl, and R. Klein, "Efficient RANSAC for point-cloud shape detection," *Comput. Graph. Forum*, vol. 26, no. 2, pp. 214–226, 2007.
- [66] L. Nan and P. Wonka, "PolyFit: Polygonal surface reconstruction from point clouds," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2353–2361.
- [67] Z. Zhu et al., "PDE patch-based surface reconstruction from point clouds," *J. Comput. Sci.*, vol. 61, 2022, Art. no. 101647.
- [68] Y. Li, A. Dai, L. Guibas, and M. Nießner, "Database-assisted object retrieval for real-time 3D reconstruction," *Comput. Graph. Forum*, vol. 34, no. 2, pp. 435–446, 2015.
- [69] Y. M. Kim, N. J. Mitra, Q. Huang, and L. Guibas, "Guided real-time scanning of indoor objects," *Comput. Graph. Forum*, vol. 32, no. 7, pp. 177–186, 2013.
- [70] L. Nan, K. Xie, and A. Sharf, "A search-classify approach for cluttered indoor scene understanding," *ACM Trans. Graph.*, vol. 31, no. 6, pp. 1–10, 2012.
- [71] Y. M. Kim, N. J. Mitra, D.-M. Yan, and L. Guibas, "Acquiring 3D indoor environments with variability and repetition," *ACM Trans. Graph.*, vol. 31, no. 6, pp. 1–11, 2012.
- [72] M. Pauly, N. J. Mitra, J. Giesen, M. H. Gross, and L. J. Guibas, "Example-based 3D scan completion," in *Proc. Symp. Geometry Process.*, 2005, pp. 23–32.
- [73] C.-H. Shen, H. Fu, K. Chen, and S.-M. Hu, "Structure recovery by part assembly," *ACM Trans. Graph.*, vol. 31, no. 6, pp. 1–11, 2012.
- [74] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 9446–9454.
- [75] F. Williams, T. Schneider, C. Silva, D. Zorin, J. Bruna, and D. Panozzo, "Deep geometric prior for surface reconstruction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 10130–10139.
- [76] R. Hanocka, G. Metzer, R. Giryes, and D. Cohen-Or, "Point2Mesh: A self-prior for deformable meshes," *ACM Trans. Graph.*, vol. 39, no. 4, Jul. 2020, Art. no. 126.
- [77] W. Zhao, J. Lei, Y. Wen, J. Zhang, and K. Jia, "Sign-agnostic implicit learning of surface self-similarities for shape modeling and reconstruction from raw point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 10256–10265.
- [78] R. Hanocka et al., "MeshCNN: A network with an edge," *ACM Trans. Graph.*, vol. 38, no. 4, pp. 1–12, 2019.
- [79] M. Gadelha, R. Wang, and S. Maji, "Deep manifold prior," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops*, 2021, pp. 1107–1116.
- [80] M. Atzmon, N. Haim, L. Yariv, O. Israelov, H. Maron, and Y. Lipman, "Controlling neural level sets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, Art. no. 182.
- [81] J. Lei and K. Jia, "Analytic marching: An analytic meshing solution from deep implicit surface networks," in *Proc. Int. Conf. Mach. Learn.*, 2020, Art. no. 537.
- [82] M. Atzmon and Y. Lipman, "SAL: Sign agnostic learning of shapes from raw data," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 2565–2574.
- [83] M. Atzmon and Y. Lipman, "SALD: Sign agnostic learning with derivatives," in *Proc. Int. Conf. Learn. Representations*, 2021, pp. 1–14.
- [84] A. Basher, M. Sarmad, and J. Boutellier, "LightSAL: Lightweight sign agnostic learning for implicit surface representation," 2021, *arXiv:2103.14273*.
- [85] R. Zhu et al., "Semi-signed neural fitting for surface reconstruction from unoriented point clouds," 2022, *arXiv:2206.06715*.
- [86] T. Davies, D. Nowrouzezahrai, and A. Jacobson, "On the effectiveness of weight-encoded neural implicit 3D shapes," 2021, *arXiv: 2009.09808*.
- [87] F. Williams, M. Trager, J. Bruna, and D. Zorin, "Neural splines: Fitting 3D surfaces with infinitely-wide neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 9949–9958.
- [88] Y. Liao, S. Donne, and A. Geiger, "Deep marching cubes: Learning explicit surface representations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2916–2925.
- [89] P. Erler, P. Guerrero, S. Ohrhallinger, N. J. Mitra, and M. Wimmer, "Points2Surf learning implicit surfaces from point clouds," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2020, pp. 108–124.
- [90] M. Yang, Y. Wen, W. Chen, Y. Chen, and K. Jia, "Deep optimized priors for 3D shape modeling and reconstruction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 3269–3278.
- [91] Z. Chen and H. Zhang, "Learning implicit fields for generative shape modeling," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5939–5948.
- [92] Y. Duan, H. Zhu, H. Wang, L. Yi, R. Nevatia, and L. J. Guibas, "Curriculum DeepSDF," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2020, pp. 51–67.
- [93] S. Yao, F. Yang, Y. Cheng, and M. G. Mozerov, "3D shapes local geometry codes learning with SDF," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 2110–2117.
- [94] B. Guillard, F. Stella, and P. Fua, "Meshuflf: Fast and differentiable meshing of unsigned distance field networks," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 576–592.
- [95] S. Giebenhain and B. Goldluecke, "AIR-nets: An attention-based framework for locally conditioned implicit representations," in *Proc. Int. Conf. 3D Vis.*, 2021, pp. 1054–1064.
- [96] X. Yan, L. Lin, N. J. Mitra, D. Lischinski, D. Cohen-Or, and H. Huang, "ShapeFormer: Transformer-based shape completion via sparse representation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 6229–6239.
- [97] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.
- [98] H. Zhao, L. Jiang, J. Jia, P. H. Torr, and V. Koltun, "Point transformer," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 16259–16268.
- [99] J. Tang, X. Han, M. Tan, X. Tong, and K. Jia, "SkeletonNet: A topology-preserving solution for learning mesh reconstruction of object surfaces from RGB images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 10, pp. 6454–6471, Oct. 2022.
- [100] J. Pan, X. Han, W. Chen, J. Tang, and K. Jia, "Deep mesh reconstruction from single RGB images via topology modification networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 9963–9972.
- [101] J. Chibane, T. Alldieck, and G. Pons-Moll, "Implicit functions in feature space for 3D shape reconstruction and completion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 6970–6981.

- [102] J. Chibane, A. Mir, and G. Pons-Moll, “Neural unsigned distance fields for implicit function learning,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, Art. no. 1816.
- [103] Z. Mi, Y. Luo, and W. Tao, “SSRNet: Scalable 3D surface reconstruction network,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 970–979.
- [104] T. Takikawa et al., “Neural geometric level of detail: Real-time rendering with implicit 3D shapes,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 11358–11367.
- [105] A. Badki, O. Gallo, J. Kautz, and P. Sen, “Meshlet priors for 3D mesh reconstruction,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 2849–2858.
- [106] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “PointNet: Deep learning on point sets for 3D classification and segmentation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 652–660.
- [107] M. Tatarchenko, J. Park, V. Koltun, and Q.-Y. Zhou, “Tangent convolutions for dense prediction in 3D,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3887–3896.
- [108] B. Ummenhofer and V. Koltun, “Adaptive surface reconstruction with multiscale convolutional kernels,” in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 5651–5660.
- [109] A. Boulch and R. Marlet, “POCO: Point convolution for surface reconstruction,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 6302–6314.
- [110] H. Jiang, J. Cai, J. Zheng, and J. Xiao, “Neighborhood-based neural implicit reconstruction from point clouds,” in *Proc. Int. Conf. 3D Vis.*, 2021, pp. 1259–1268.
- [111] B. Ma, Y.-S. Liu, M. Zwicker, and Z. Han, “Surface reconstruction from point clouds by learning predictive context priors,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 6326–6337.
- [112] D. Attali, J.-D. Boissonnat, and A. Lieutier, “Complexity of the delaunay triangulation of points on surfaces the smooth case,” in *Proc. 19th Annu. Symp. Comput. Geometry*, 2003, pp. 201–210.
- [113] J.-D. Boissonnat and F. Cazals, “Smooth surface reconstruction via natural neighbour interpolation of distance functions,” *Comput. Geometry*, vol. 22, no. 1/3, pp. 185–203, 2002.
- [114] F. Lafarge and P. Alliez, “Surface reconstruction through point set structuring,” *Comput. Graph. Forum*, vol. 32, no. 2pt2, pp. 225–234, 2013.
- [115] R. R. Paulsen, J. A. Bærentzen, and R. Larsen, “Markov random field surface reconstruction,” *IEEE Trans. Vis. Comput. Graphics*, vol. 16, no. 4, pp. 636–646, Jul./Aug. 2010.
- [116] J. Giesen, S. Spalinger, and B. Schölkopf, “Kernel methods for implicit surface modeling,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2005, pp. 1193–1200.
- [117] F. Williams et al., “Neural fields as learnable kernels for 3D reconstruction,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 18500–18510.
- [118] M. Baorui, H. Zhizhong, L. Yu-Shen, and Z. Matthias, “Neural-pull: Learning signed distance functions from point clouds by learning to pull space onto surfaces,” in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 1–12.
- [119] Y. Lipman, “Phase transitions, distance functions, and implicit neural representations,” in *Proc. 38th Int. Conf. Mach. Learn.*, 2021, 2021, pp. 6702–6712.
- [120] P.-S. Wang, Y. Liu, Y.-Q. Yang, and X. Tong, “Spline positional encoding for learning 3D implicit signed distance fields,” in *Proc. 30th Int. Joint Conf. Artif. Intell.*, 2021, pp. 1091–1097.
- [121] Z. Wang et al., “Neural-IMLS: Learning implicit moving least-squares for surface reconstruction from unoriented point clouds,” 2021, *arXiv:2109.04398*.
- [122] B. Ma, Y.-S. Liu, and Z. Han, “Reconstructing surfaces for sparse point clouds with on-surface priors,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 6315–6325.
- [123] M.-J. Rakotosaona, P. Guerrero, N. Aigerman, N. J. Mitra, and M. Ovsjanikov, “Learning delaunay surface elements for mesh reconstruction,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 22–31.
- [124] S.-L. Liu, H.-X. Guo, H. Pan, P.-S. Wang, X. Tong, and Y. Liu, “Deep implicit moving least-squares functions for 3D reconstruction,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 1788–1797.
- [125] L. Li, M. Sung, A. Dubrovina, L. Yi, and L. J. Guibas, “Supervised fitting of geometric primitives to 3D point clouds,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2652–2660.
- [126] M. Liu, X. Zhang, and H. Su, “Meshing point clouds with predicted intrinsic-extrinsic ratio guidance,” in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2020, pp. 68–84.
- [127] J. Gao, W. Chen, T. Xiang, A. Jacobson, M. McGuire, and S. Fidler, “Learning deformable tetrahedral meshes for 3D reconstruction,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, Art. no. 833.
- [128] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang, “Pixel2Mesh: Generating 3D mesh models from single RGB images,” in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 52–67.
- [129] H. Kato, Y. Ushiku, and T. Harada, “Neural 3D mesh renderer,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3907–3916.
- [130] D. Xiao, S. Lin, Z. Shi, and B. Wang, “Learning modified indicator functions for surface reconstruction,” *Comput. Graph.*, vol. 102, pp. 309–319, 2022.
- [131] E.-T. Lê, M. Sung, D. Ceylan, R. Mech, T. Boubekeur, and N. J. Mitra, “CPFN: Cascaded primitive fitting networks for high-resolution point clouds,” in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 7457–7466.
- [132] K. Genova, F. Cole, A. Sud, A. Sarna, and T. Funkhouser, “Local deep implicit functions for 3D shape,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4857–4866.
- [133] Y. Siddiqui, J. Thies, F. Ma, Q. Shan, M. Nießner, and A. Dai, “Retrieval-fuse: Neural 3D scene reconstruction with a database,” in *Proc. Int. Conf. Comput. Vis.*, 2021, pp. 12548–12557.
- [134] J. Rossignac, “Shape complexity,” *Vis. Comput.*, vol. 21, no. 12, pp. 985–996, 2005.
- [135] D. Stutz, “A formal definition of watertight meshes,” 2018. Accessed: Dec. 2021. [Online]. Available: <https://davidstutz.de/a-formal-definition-of-watertight-meshes>
- [136] J. Kontkanen and S. Laine, “Ambient occlusion fields,” in *Proc. Symp. Interactive 3D Graph. Games*, 2005, pp. 41–48.
- [137] M. Gschwandtner, R. Kwitt, A. Uhl, and W. Pree, “BlenSor: Blender sensor simulation toolbox,” in *Proc. Int. Symp. Vis. Comput.*, Springer, 2011, pp. 199–208.
- [138] L. Li, “Time-of-flight camera—An introduction,” May 2014. [Online]. Available: <https://www.ti.com/lit/wp/sloa190b/sloa190b.pdf>
- [139] E. W. Weisstein, “Euler angles,” 2009. [Online]. Available: <https://mathworld.wolfram.com>
- [140] J. Jiao, L. Yuan, W. Tang, Z. Deng, and Q. Wu, “A post-rectification approach of depth images of kinect v2 for 3D reconstruction of indoor scenes,” *ISPRS Int. J. Geo-Inf.*, vol. 6, no. 11, p. 349, 2017.
- [141] D. Girardeau-Montaut, “Cloudcompare (version 2.12.0) [GPL software],” Mar 2022. [Online]. Available: <http://www.cloudcompare.org>
- [142] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz, “Towards 3D point cloud based object maps for household environments,” *Robot. Auton. Syst.*, vol. 56, no. 11, pp. 927–941, 2008.
- [143] F. Cazals and M. Pouget, “Estimating differential quantities using polynomial fitting of osculating jets,” *Comput. Aided Geometric Des.*, vol. 22, no. 2, pp. 121–146, 2005.
- [144] J. Yang, R. Li, Y. Xiao, and Z. Cao, “3D reconstruction from non-uniform point clouds via local hierarchical clustering,” in *Proc. 9th Int. Conf. Digit. Image Process.*, SPIE, 2017, pp. 685–689.
- [145] H. Fan, H. Su, and L. J. Guibas, “A point set generation network for 3D object reconstruction from a single image,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 605–613.
- [146] A. Knapsch, J. Park, Q.-Y. Zhou, and V. Koltun, “Tanks and temples: Benchmarking large-scale scene reconstruction,” *ACM Trans. Graph.*, vol. 36, no. 4, pp. 1–13, 2017.
- [147] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, “The unreasonable effectiveness of deep features as a perceptual metric,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 586–595.
- [148] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung, “On visual similarity based 3D model retrieval,” *Comput. Graph. Forum*, vol. 22, no. 3, pp. 223–232, 2003.
- [149] A. Fabri and S. Pion, “CGAL: The computational geometry algorithms library,” in *Proc. 17th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, 2009, pp. 538–539.
- [150] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia, “MeshLab: An open-source mesh processing tool,” in *Proc. Eurographics Italian Chapter Conf.*, V. Scarano, R. D. Chiara, and U. Erra, Eds., The Eurographics Association, 2008, pp. 129–136.



Zhang Jin Huang received the graduate degree from the School of Electronic and Information Engineering, South China University of Technology, Guangzhou, China, in 2021. He is currently working toward the master's degree in electronic information with the South China University of Technology. His research interests include computer vision and machine learning.



Yuxin Wen is currently working toward the PhD degree in information and communication engineering with the School of Electronic and Information Engineering, South China University of Technology. His research interests are in computer vision and machine learning, with focus on deep learning shape modeling and reconstruction of objects and scenes.



ZiHao Wang is currently working toward the master's degree in electronic information with the School of Electronic and Information Engineering, South China University of Technology. His research interests include computer vision, depth sensor, and machine learning.



Jinjuan Ren received the BA degree from Fudan University, in 2001, the MPhil degree from the University of Hong Kong, in 2004, and the PhD degree from the University of Hong Kong, in 2009. She has been an associate professor with the University of Macau since 2015. She was an assistant professor with the University of Macau from 2009 to 2015. Her research areas include fintech, machine learning, and behavioral finance. She published papers in top journals such as *Management Science*. She is a CFA charter-holder.



Kui Jia received the BEng degree in marine engineering from Northwestern Polytechnical University, China, in 2001, the MEng degree in electrical and computer engineering from the National University of Singapore, in 2003, and the PhD degree in computer science from the Queen Mary University of London, London, U.K., in 2007. He is currently a professor with the School of Electronic and Information Engineering, South China University of Technology. His research interests are in computer vision and machine learning. His recent research focuses on theoretical deep learning and its applications to 3D vision. He has been serving as an associate editor for *IEEE Trans. on Image Processing* and *Transaction on Machine Learning Research*.