Point2CAD: Reverse Engineering CAD Models from 3D Point Clouds

Yujia Liu
ETH Zürich

Anton Obukhov
ETH Zürich

Jan Dirk Wegner
University of Zürich

Konrad Schindler
ETH Zürich

Figure 1. **Point2CAD** reconstructs complex CAD models from 3D point clouds. A point cloud is segmented into clusters corresponding to CAD faces. Each face is fitted with a geometric primitive or a parametric surface using a novel neural representation. Due to the analytic representation, the surfaces can be extended and intersected such that topology emerges, which is then used to clip the surface primitives.

Abstract

Computer-Aided Design (CAD) model reconstruction from point clouds is an important problem at the intersection of computer vision, graphics, and machine learning. Recent advancements in this direction achieve rather reliable semantic segmentation but still struggle to produce an adequate topology of the CAD model. We propose a hybrid analytic-neural reconstruction scheme that bridges the gap between segmented point clouds and structured CAD models. To power the surface fitting stage, we propose a novel implicit neural representation of freeform surfaces, driving up the performance of our overall CAD reconstruction scheme. We evaluate our method on the ABC benchmark of CAD models and set a new state-of-the-art for that dataset.

1. Introduction

The task of reverse engineering CAD models from 3D point clouds has gained increasing attention in recent years due to the rapid development of 3D scanning technologies. Most approaches to reverse engineering CAD models follow a typical sequence of steps: point cloud capture, parts segmentation, and analytic representation inference. Research that aims to automate the process has focused chiefly on individual steps, whereas little work covers the complete workflow of CAD model reconstruction.

The present work addresses this gap by proposing **Point2CAD**, a method that recovers complete CAD models, including free-form surfaces, edges, and corners. Our proposed pipeline for CAD model reconstruction from point clouds is composed of several steps. First, a pre-trained neural network is employed to segment the point cloud into clusters corresponding to distinct surfaces. Second, basic primitives and a novel implicit neural representation (INR) of freeform surfaces, are fitted to the clusters. Third, adjacent surfaces are intersected to recover edges, and adjacent edges are further intersected to recover corners, thus obtaining a full B-rep. Taken together, these steps form a comprehensive and versatile pipeline for reverse engineering point clouds into CAD models, see Fig. 1. By combining modern, learning-based segmentation backbones with classical geometric primitive fitting and with recent neural field methods for freeform surfaces, we get the best of both worlds and obtain a reconstruction pipeline that sets a new state of the art on the large-scale ABC benchmark [5].

2. Related Work

3D point cloud segmentation. Neural architectures to learn feature embeddings of point clouds include: graph convolutions [14], point-voxel learning [9], and transformers [16].
Primitive fitting Several learning-based approaches were proposed to fit geometric primitives to point clouds. ParSeNet [11] finds parametric surfaces in point clouds, including basic geometric primitives as well as B-spline surfaces, but does not connect them. HPNet [15] focuses on partitioning the point cloud into segments using semantic as well as spectral features and edge information in the form of an adjacency matrix but does not fit actual primitives.
Generic CAD modeling. ComplexGen [1] reconstructs CAD models by autoregressively detecting geometric primitives. It consists of a CNN encoder, three transformer decoders for geometric primitives and topology, and post-processing with global optimization for the final refinement.
Manifold learning techniques aim at discovering a low-dimensional manifold underlying a higher-dimensional data set. Most methods focus on visualization or fidelity to particular input points, by explicitly using those points to construct the mapping [7, 10, 13]. Autoencoders provide a natural way to not only project data non-linearly to a lower-dimensional latent space, but also to decode back from the latent space to the original data space [6]. We extend this approach with the recent findings about implicit neural representations to design a fitting method for freeform surfaces.
Implicit neural representations (INR) are a generic framework to encode an arbitrary function observed in the form of sparse samples into a neural network. Research on neural rendering has led to several useful insights about neural fields, e.g., the importance of positional encoding [8] and new activation functions [12]. These findings form the basis for our freeform surface fitting method.

3. Method

Contrary to the recent trend towards generic, end-to-end deep learning pipelines, we found it advantageous to split the reconstruction process into steps and only use neural methods where necessary. Overall, our method consists of the following stages, *cf*. Fig. 1:

1. Partition the point cloud into clusters corresponding to the CAD model's topological faces. For that step, we rely on existing (pretrained) neural network methods.
2. Fit an analytical surface primitive to each cluster. Here we use a hybrid approach: First, we test a set of prevalent geometric surface primitives that admit efficient closed-form fitting. For freeform surfaces, we propose a novel fitting scheme based on implicit neural representations.
3. Find the effective area of each parametric surface and clip it, leaving enough margin to intersect adjacent surfaces.

Figure 2. Inspiration items of our INR surface fitting. UMAP [7] (**1st column, middle**) learns the underlying 2D manifold of 3D points along with the inverse mapping but cannot capture its smooth analytic representation. Early experiments with ReLU activations confirmed its low-frequency bias (**1st column, bottom**) and piecewise linearity. SiLU [2] (**2nd column**) suffers from low-frequency bias, too. We found that resolving it by adding positional encoding is challenging under our training protocol (**3rd column**). SIREN [12] (**4th column**) fits the data well but does not extrapolate.

4. Perform pairwise surface intersections to obtain a set of topologically plausible object edges. Using these edges, remove parts not supported by input points.
5. Perform pairwise edge intersection to identify a set of topological corners. Clip edges based on proximity to the remaining surface regions and inferred corners.

As a result of applying Point2CAD, we obtain a CAD model in B-rep format, which includes analytical surfaces to represent the model's faces, compatible edges and corners, and adjacency matrices that encode topology.

A cornerstone of the proposed pipeline is a novel method for fitting freeform parametric 2D-manifold surfaces in 3D to an unordered set of points. The 2D latent space allows for interpretable surface extrapolation, required in steps 3-4.

While using neural methods, INR is a pure test-time optimization. We effectively side-step learning priors from training data except for the initial low-level segmentation. We argue that simple analytical procedures like surface fitting do not necessarily benefit from learned priors and that an over-reliance may have hampered recent work on reverse engineering CAD models on data-driven learning. We empirically support this claim in our experiments, where we outperform purely learning-based methods like [1].

3.1. Parametrization of standard primitives

The geometric primitives that our current implementation handles are planes, spheres, cylinders, and cones. The exact parameterizations are taken the same as in [11]. We observed that utilizing the predicted surface type harms the reconstruction since the prediction is not always correct. Therefore, we instead rely on clustering and an exhaustive comparison: we fit each primitive type (including INR) to point clusters and select the model with the lowest reconstruction error.

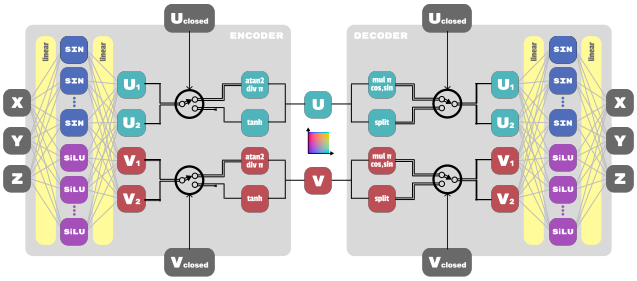

Figure 3. Block-scheme of our proposed INR for freeform surfaces fitting. We autoencode groups of 3D surface points into a latent 2D uv space. We use a mixture of activations achieve high fidelity of fitting and smooth extrapolation. Both open and closed surfaces are supported via preconfigured routing. See Sec. 3.2 for more details.

3.2. Freeform surface parametrization with INR

For the overall CAD reconstruction pipeline, this step should meet several requirements: (1) resilience to noise in the input point cloud; (2) support for inverse mapping to enable traversal of the latent space; (3) flexibility in interpolation mode to ensure high data fidelity and avoid oversmoothing; (4) strong regularization in extrapolation mode for surface intersection; (5) very low computational cost.

Existing manifold learning techniques that support the inverse transform, such as UMAP [7], could not be readily used for this task. We thus developed a neural autoencoder [6] with a single hidden layer, sufficient for simple non-linear transforms [3]. Our key finding is that different activation functions comply with subsets of requirements, and violate others, e.g., SiLU [2] leads to smooth extrapolation but does not interpolate well, whereas SIREN [12] exhibits the opposite behavior. We employ a mixture of these activations, which leads to the solution satisfying the above requirements; see Fig. 2 for ablation visualization.

Fig. 3 depicts a block diagram of the proposed autoencoder. Each surface is treated with a separate INR independently by feeding batches of its 3D points (X,Y,Z) through a 1-layer MLP encoder and a corresponding decoder, with a 2D bottleneck corresponding to manifold coordinates (u,v).

For each surface, the weights of a template INR are initialized randomly and optimized with standard mini-batch (or full-batch) descent. We run Adam [4] for 1000 steps, with 50 steps warm-up of the learning rate, followed by a linear decay that reaches zero at the last step. The complete optimization takes only a few seconds on a single GPU, and multiple surfaces can be fitted in parallel.

For extrapolation, we encode all cluster points into the latent uv space and store the bounding box parameters along with the autoencoder. To sample the surface with the margin, we extend the said bounding box by 10% in both dimensions and compute 3D points using the decoder.

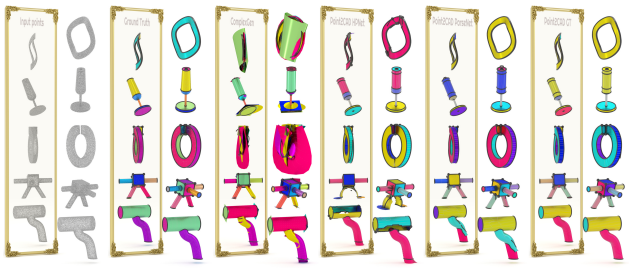

Figure 4. Results gallery. Different colors denote different surfaces. Edges and corners are depicted with black plastic. Left to right: input point cloud, ground truth mesh, and reconstruction with ComplexGen [1]. Then the proposed Point2CAD method is applied to different segmentation: with HPNet [15], ParSeNet [11], and Ground Truth. Our method reconstructs the ground truth geometry and topology from the ground truth segmentation nearly perfectly. When applied on top of pretrained segmentation modules, it outperforms the competition by a high margin.

3.3. Topology reconstruction

After instantiating all individual surfaces, it is instrumental to establish their boundaries, which amounts to finding the intersection curves between adjacent surfaces. This requirement again calls for deterministic geometric methods, as independent detection can hardly guarantee a set of edges that is complete and consistent with the surfaces. While working out the surface intersections analytically is possible, multiple freeform surfaces lead to complicated calculations.

Hence, we convert all surfaces to triangle meshes for this step, such that they can be accurately intersected with mature, numerically stable tools. For each surface, the geometric primitive is trimmed to a margin of width ϵ around the supporting points and then fed to a standard meshing algorithm. We compute pairwise mesh intersections to obtain edges as polylines and re-mesh the surfaces along those edges. Similarly, we intersect adjacent edge poly-lines to obtain corner points and use those corners to trim the edges.

4. Experiments

The flexible constraints on the input to our method permit employing it in various setups. Evaluation of the ground truth point cloud clustering or segmentation can be seen as a way to quantify the contribution of point cloud sampling sparsity and sample noise on the reconstruction quality (aliased "Point2CAD GT"). The main case of interest is the usage on top of any pretrained point cloud clustering or segmentation methods, such as ParSeNet [11] or HPNet [15].

To evaluate Point2CAD, we conduct experiments on the ABC dataset [5], a large-scale collection of CAD models ($\sim$1.000.000 models). We use a subset of the same split as ParseNet, where each model contains at least one freeform

Table 1. Fitting on freeform surfaces. In ComplexGen, a transformer-style decoder is utilized to convert the latent code of surfaces to a grid of 20x20 points on surfaces. ParseNet employs a neural network to output a 20x20 control-point grid. For us, we utilize the introduced INR.

	Open surfaces		Closed surfaces	
	Res-err ↓	P-cover ↑	Res-err ↓	P-cover ↑
ComplexGen	0.021	0.938	0.023	0.900
ParseNet	0.006	0.930	0.008	0.902
Point2CAD (INR)	**0.002**	**0.999**	**0.003**	**0.998**

Table 2. Geometric evaluation on reconstructed CADs. Segmentation denotes the method used for point cloud clustering. "GT" stands for oracle ground truth segmentation, which is also an upper bound of the performance of our method.

	Segmentation	Res-err ↓	P-cover ↑	Chamfer ↓
ComplexGen	N/A	0.020	**0.950**	0.042
Point2CAD	ParseNet	**0.018**	0.933	**0.017**
Point2CAD	HPNet	0.020	0.937	0.018
Point2CAD	GT	0.011	0.947	0.016

surface, to facilitate a fair comparison with existing methods and to better demonstrate the feasibility of our approach.

Following the evaluation protocols outlined in [11,15], we employ several geometric metrics, including: (1) **Residual error** as a measure of discrepancy between reconstructed surfaces and their corresponding ground truth counterparts determined via Hungarian matching; (2) **Chamfer** distance, a bidirectional residual measure disregarding separation into surfaces; (3) **P-coverage**, the proportion of input points in the cloud having a generated surface in close proximity; (4) [*Surface*, *Edge*, *Corner*] × [**precision**, **recall**, **F-score**] proposed in [1] to disentangle topological fidelity factors.

4.1. Evaluation on Freeform Surfaces

Both ComplexGen [1] and ParseNet [11] have successfully demonstrated the prediction of the freeform surfaces. We analyze a subset of freeform surfaces of the ABC dataset and average the geometry fitting metrics; see Tab. 1 for quantitative evaluation. INR fitting method generates freeform surfaces that represent the underlying points more faithfully than prior parameterizations. Qualitative results of Point2CAD freeform surface fitting confirm the qualitative study, also seen in the side-by-side comparison in Fig. 4.

4.2. Reconstructed CAD Evaluation

As seen in Tab. 2, Point2CAD outperforms ComplexGen in all topological metrics except P-coverage. We attribute the latter effect to the asymmetric nature of the metric favoring spurious predictions. Furthermore, Point2CAD achieves excellent results when fed with ground truth segmentations, indicating that it will likely further improve as better point cloud segmentation engines become available.

Table 3. Evaluation on CAD Reconstruction of *Surfaces*, *Edges* and *Corners* from the aspects of accuracy and completeness. The surface evaluation assesses the reconstruction performance in geometric terms, while these metrics on edges and corners reflect the reconstruction performance regarding topological properties.

Surfaces		$\theta_{\text{surface}} = 0.03$		
Method	Segmentation	precision ↑	recall ↑	F-score ↑
ComplexGen	N/A	0.370	0.388	0.379
Point2CAD	ParseNet	0.578	0.520	0.547
Point2CAD	HPNet	**0.644**	**0.540**	**0.587**
Point2CAD	GT	0.838	0.731	0.781
Edges		$\theta_{\text{edge}} = 0.02$		
Method	Segmentation	precision ↑	recall ↑	F-score ↑
ComplexGen	N/A	0.290	0.279	0.284
Point2CAD	ParseNet	0.332	**0.381**	0.355
Point2CAD	HPNet	**0.351**	0.368	**0.360**
Point2CAD	GT	0.493	0.517	0.505
Corners		$\theta_{\text{corner}} = 0.01$		
Method	Segmentation	precision ↑	recall ↑	F-score ↑
ComplexGen	N/A	0.217	0.203	0.210
Point2CAD	ParseNet	0.349	**0.401**	**0.373**
Point2CAD	HPNet	**0.353**	0.380	0.366
Point2CAD	GT	0.451	0.524	0.485

Precision, recall, and F-score of topological surfaces, edges, and corners are shown in Tab. 3. The results suggest that Point2CAD with HPNet segmentation backbone achieves the highest performance, while reconstruction based on the ground truth point cloud segmentation provides the best results, as expected. We find that our "segmentation-fit-intersect" approach generates more reliable CAD models than learned generative methods like ComplexGen.

5. Conclusion

We present Point2CAD, a comprehensive and versatile approach for reverse engineering CAD models from point clouds, which can handle various types of surfaces and yield topologically consistent reconstructions. The proposed method segments point clouds with a pre-trained segmentation backbone but then employs learning-free optimization methods to fit geometric primitives, including freeform surfaces that are optimized with a novel implicit neural representation. The recovered surfaces are analytically intersected to obtain the edges and corners of the model. Empirically, our proposed method outperforms prior art on the popular ABC dataset of CAD models.

References

[1] Haoxiang Guo, Shilin Liu, Hao Pan, Yang Liu, Xin Tong, and Baining Guo. ComplexGen: CAD reconstruction by B-rep chain complex generation. *ACM TOG*, 41(4):1–18, 2022. 2, 3, 4

[2] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 2, 3

[3] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989. 3

[4] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 3

[5] Sebastian Koch, Albert Matveev, Zhongshi Jiang, Francis Williams, Alexey Artemov, Evgeny Burnaev, Marc Alexa, Denis Zorin, and Daniele Panozzo. ABC: A big CAD model dataset for geometric deep learning. In *CVPR*, 2019. 1, 3

[6] Mark A Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*, 37(2):233–243, 1991. 2, 3

[7] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2018. 2, 3

[8] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM*, 65(1):99–106, dec 2021. 2

[9] Dario Rethage, Johanna Wald, Jurgen Sturm, Nassir Navab, and Federico Tombari. Fully-convolutional point networks for large-scale point clouds. In *ECCV*, 2018. 2

[10] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000. 2

[11] Gopal Sharma, Difan Liu, Subhransu Maji, Evangelos Kalogerakis, Siddhartha Chaudhuri, and Radomír Měch. ParSeNet: A parametric surface fitting network for 3d point clouds. In *ECCV*, 2020. 2, 3, 4

[12] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 7462–7473. Curran Associates, Inc., 2020. 2, 3

[13] Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000. 2

[14] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM TOG*, 38(5):1–12, 2019. 2

[15] Siming Yan, Zhenpei Yang, Chongyang Ma, Haibin Huang, Etienne Vouga, and Qixing Huang. HPNet: Deep primitive segmentation using hybrid representations. In *ICCV*, 2021. 2, 3, 4

[16] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *ICCV*, 2021. 2