

Registered and Segmented Deformable Object Reconstruction from a Single View Point Cloud

Pit Henrich¹, Balázs Gyenes², Paul Maria Scheikl¹,
Gerhard Neumann², Franziska Mathis-Ullrich¹ +

¹ Dep. Artificial Intelligence in Biomedical Engineering - FAU Erlangen-Nürnberg, Erlangen, Germany

² Institute for Anthropomatics and Robotics - Karlsruhe Institute of Technology, Karlsruhe, Germany

{pit.henrich, paul.m.scheikl, franziska.mathis-ullrich}@fau.de

{balazs.gyenes, gerhard.neumann}@kit.edu

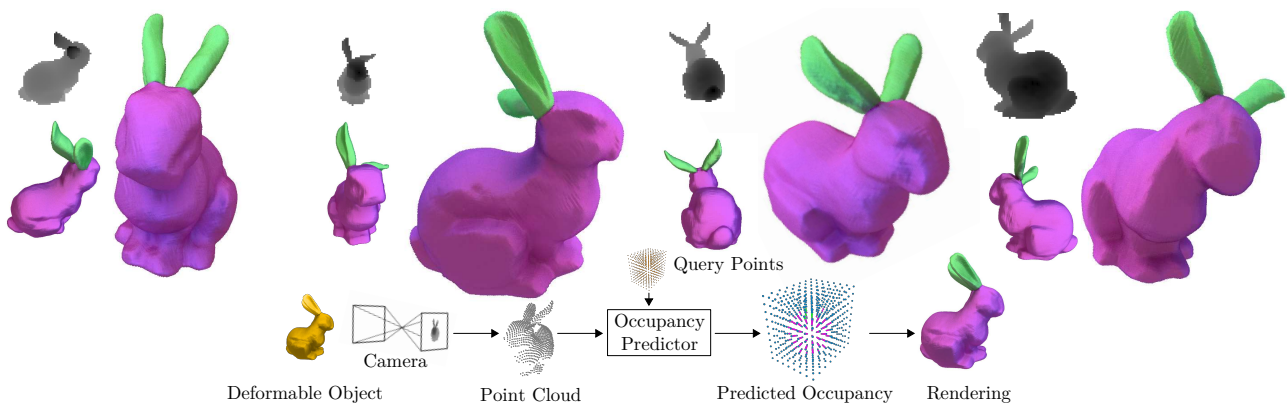


Figure 1. A Point Cloud and Query Points from a regular grid are passed to the Occupancy Predictor to infer the volumetric model of a deformed object.

Abstract

In deformable object manipulation, we often want to interact with specific segments of an object that are only defined in non-deformed models of the object. We thus require a system that can recognize and locate these segments in sensor data of deformed real world objects. This is normally done using deformable object registration, which is problem specific and complex to tune. Recent methods utilize neural occupancy functions to improve deformable object registration by registering to an object reconstruction. Going one step further, we propose a system that in addition to reconstruction learns segmentation of the reconstructed object. As the resulting output already contains the information about the segments, we can skip the registration process. Tested on a variety of deformable objects in simulation and the real world, we demonstrate that our method learns to robustly find these segments. We also introduce a sim-

ple sampling algorithm to generate better training data for occupancy learning.

1. Introduction

Our research is driven by the demands of robot-assisted surgical applications. These include the ability to find and interact with specific segments, for example to grasp or cut them. Soft organs are highly deformable, which makes these interactions challenging. To find segments of interest, current work focuses on registering a known segmented model to sensor data. The known models can be obtained from a medical CT, where the segments are then defined manually.

Instead of using deformable registration methods, we propose a method to directly reconstruct deformable objects with all their segments from depth images. The segmented reconstruction can then be used instead of a registered model for planning the interactions.

+Corresponding author

Emerging object reconstruction methods learn a continuous object representation to create voxel or surface meshes at any resolution. For example, neural occupancy functions [21, 28] or Signed Distance Function (SDF)s [30, 36]. Labeled points in 3D space are used to train such continuous representation. However, the impact of generation and distribution of these points is rarely detailed or investigated.

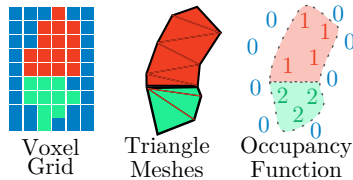


Figure 2. Three ways to represent objects: using a regular grid of voxels, using two Triangle Meshes, and using an Occupancy Function. In this example, the Occupancy Function outputs values of 0, 1, or 2 for every point depending on its location. The shape of the object is implicitly defined by the boundaries between regions of the same value.

Binary neural occupancy is usually defined as a function in 3D space that takes on discrete values 0 and 1 to indicate if the point is inside or outside of a solid object, *i.e.* $o_{\text{binary}} : \mathbb{R}^3 \rightarrow \{0, 1\}$. We consider an extension to this definition to include multiple class labels $o_{\text{multi}} : \mathbb{R}^3 \rightarrow \{0, 1, \dots, n\}$, where n is the number of segments. Fig. 2 illustrates how an occupancy function encodes an object in comparison to voxel and triangle models. We consider 0 to be the label for empty space. Objects are implicitly defined by the boundaries between the clusters of points with the same occupancy value. A multi-class occupancy function with $0, \dots, n$ labels can be used to represent an object with n segments. By conditioning this multi-class occupancy function on sensor data, it can be used to reconstruct objects in different deformation states.

Contribution We train a neural system end-to-end for 3D reconstruction and segmentation using and demonstrate its ability to reconstruct objects accurately from a single-view point cloud, despite deformations and occlusions. Our contributions include:

- 1) Extending binary occupancy functions over continuous domains to multi-class occupancy.
- 2) Developing an algorithm with few hyperparameters for generating high-quality training data for occupancy learning.
- 3) Provide an alternative to traditional deformable object registration methods.

2. Related Work

2.1. Semantic Scene Completion

Semantic Scene Completion (SSC) provides a labeled 3D environment model based on sensor data. This model can be used for tasks such as autonomous navigation. Based on a single depth image, Song et al. [37] predict semantic labels for a voxel grid.

To improve SSC, multi-modal (color and depth images) sensor data can be used [24]. Cai et al. [3] further improve SSC by extracting object instances from the scene and propagating the object details back up to the scene.

When labeling a voxel grid using convolutional architectures, there is a trade-off between spatial resolution and memory requirements. For example, based on a depth image, Song et al. [37] use a limited resolution of $(240 \times 144 \times 240)$ to approximate the structure of a room. Rist et al. [33] use a continuous representation that is not based on voxelization for semantic scene completion. Thereby, surpassing previous voxel-based methods in geometric accuracy.

2.2. Shape Reconstruction by Learning Implicit Functions

A central challenge in utilizing neural networks for 3D reconstruction is choosing a shape representation. Several representations, including voxel grids [7, 14, 39], dense point clouds [11, 25, 27, 43, 45], polygonal meshes [5, 8, 13, 15, 17–20, 26, 35], and manifold atlases [2, 9, 12, 16, 41] have been proposed. Each of these have unique advantages and disadvantages.

Voxel-based methods suffer from memory constraints. Point cloud methods do not define any surfaces, so they have to be approximated. Polygonal mesh and manifold atlas methods often output meshes with degenerate geometry or holes. Mesh learning is complex as it involves defining problem specific loss functions and strong regularizers.

Recent work investigates continuous reconstruction methods that circumvent these challenges. Continuous methods represent surfaces as decision boundaries. There is no direct output of a mesh or model. The object is encoded implicitly in the weights of a neural network. For any point in space, the network predicts if it is inside or outside of an object. Therefore, the object can be retrieved at any resolution. By using equidistant query points, a voxel model can be obtained. Marching cubes along the decision boundary can be used to create a triangle mesh.

Popular continuous representations are occupancy and SDF networks. Occupancy networks [28] divide the object space into clusters. The decision boundaries between the clusters represent the surfaces of the object.

Alternatively, a SDF can be used [30]. A SDF provides the distance to an object’s surface from any point in space. The sign indicates if the point is inside or outside of an ob-

ject. All points with a distance of 0 define the surface of the object. Meta-learning allows faster adaptation to a specific object instances [36]. SDF have also been used to represent deformable objects [26].

A SDF, like an occupancy function, divides a space into insides and outsides. There are scenarios where this is not possible. For example, walls scanned by a LiDAR sensor have a thicknesses that can not be inferred. Chebane et al. [6] learn an Unsigned Distance Function (UDF). This UDF can represent thin surfaces such as a plane.

Lamb et al. [23] propose a hybrid approach that combines occupancy, signed distance field, and normal estimation methods, leading to improved reconstruction accuracy.

Williams et al. [40] propose predicting an implicit surface using a trained kernel. They demonstrate that neural kernel fields can effectively reconstruct shapes when the kernel possesses an appropriate inductive bias.

2.3. Point Cloud Encoders

Object reconstruction usually depends on sensor information, including RGB images, depth images, or point clouds. Image data, RGB and depth, obtained from cameras contain camera dependent distortion, such as perspective distortion. Objects further away from the camera occupy less pixels in the depth image. Projecting a depth image back into 3D space creates a point cloud. In this point cloud, the scale of equally sized objects in the fore- and background are the same.

Working with point clouds presents unique challenges, as any function applied to a point cloud should remain invariant to the point order [31].

PointNet++ [32] is a popular hierarchical architecture that is capable of recognizing structures at various size scales. PCNN [1] extend convolutional operations to point clouds, outperforming PointNet++ in segmentation and classification tasks. Zhao et al. [44] adapt self-attention networks for point cloud processing. Their Point Transformer network and its successor Point Transformer v2 by Wu et al. [42] consistently surpasses PointNet++ in segmentation and classification tasks.

Test-time optimization with autodecoders has gained traction in this field [4,30,36] as an alternative to point order invariant encoders. In test-time optimization, the encoder is replaced with an optimization of a commutative loss function acting on the points. The commutative property ensures the point order invariance.

Point Cloud Encoders are also investigated for applications in occupancy learning Jia et al. [21] successfully utilize an altered PCNN with an additional sampling operator to learn occupancy functions from point clouds. The method was applied to improve deformable liver registration by registering to a reconstruction instead of the raw sensor data [22].

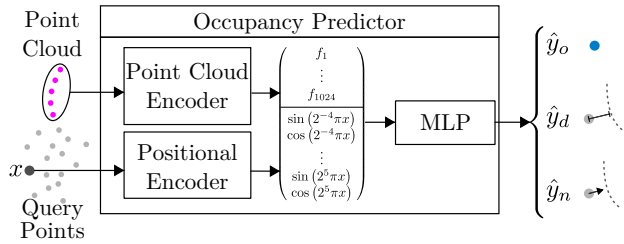


Figure 3. The Occupancy Predictor estimates the occupancy value \hat{y}_o , signed distance \hat{y}_d and direction \hat{y}_n to the nearest surface. The Point Cloud is passed to a Point Cloud Encoder to obtain the latent representation (f_1, \dots, f_{1024}) . From Query Points a random point x is taken and encoded by a Positional Encoder. The encoded x and the latent representation of the Point Cloud are concatenated and passed to the MLP which estimates \hat{y}_o , \hat{y}_d , and \hat{y}_n .

3. Method

3.1. Occupancy Predictor

The Occupancy Predictor (OP) is an approximator for an occupancy function as illustrated in Fig. 2. Receiving the position of a query point, the OP is trained on estimating the occupancy value of the point. To train the OP, pairs of position and occupancy are used.

By using only pairs of position and occupancy the OP is limited to learning only a single shape. To learn multiple shapes or objects, the OP requires information about which object to estimate. A simple solution is to pass an additional value to the OP to identify the object by. The training data then consists of (position, occupancy, identifier) triplets.

Instead of manually provided identifiers, depth images from a camera may be used for an end-to-end solution. A depth image is transformed into a Point Cloud so the OP does not need to compensate the perspective distortion of the camera. We use a Point Cloud Encoder (PCE) to produce a latent representation that is used as the identifier. The PCE handles varying Point Cloud sizes while also being invariant to point order.

Sitzmann et al. [36] suggest that learning auxiliary tasks can be beneficial to representing objects implicitly. Therefore, instead of only estimating the occupancy of a point, the OP predicts the distance and direction to the nearest surface as an auxiliary task. The resulting loss function is:

$$Loss = CrossEntropy(Y_o, \hat{Y}_o) + \lambda \cdot L1(Y_d, \hat{Y}_d) - MeanCosineSimilarity(Y_n, \hat{Y}_n) \quad (1)$$

$\lambda = 100$ was chosen heuristically such that all loss components are in the same order of magnitude. We found that clamping the distance values as suggested by DeepSDF [30] did not yield any significant benefits during training.

The latent representation of the observed Point Cloud is concatenated with the encoded position of a query point to

create a query vector. The query vector is then passed to an MLP to predict occupancy, distance, and direction to the nearest surface. The MLP has 7 fully-connected layers with 512 neurons each. Batch normalization is applied after every layer, and a skip connection concatenates the query vector with the activations after 4 layers.

Neural networks tend to learn low-frequency representations [29, 38]. We therefore use the positional encoding β as proposed in NeRF [29] to transform the query point positions into a higher dimensional space

$$\beta(x) = (\sin(2^0\pi x), \cos(2^0\pi x), \dots, \sin(2^{L-1}\pi x), \cos(2^{L-1}\pi x)) \quad (2)$$

to improve the reconstruction of high-frequency detail. Frequency encoding increases sensitivity to small changes in position, as the high-frequency components will still produce a significant signal change. For example, compare the L1 distance of original values $|0.07 - 0.09| = 0.02$ with the distance for encoded values $|\sin(2^5 \cdot 0.07) - \sin(2^5 \cdot 0.09)| \approx 0.53$.

The inputs to the OP are normalized based on the sensor data. As the sensor data does not encompass the complete object, but all parts of the object will be queried, there will be points outside of the normalization range (see output of Joint Normalizer in Fig. 5). Positional encoding uses sinusoidal functions that are 2π -periodic. To ensure that queried points can be encoded uniquely, we also use negative exponents for β . We chose frequency encoding because it allows this simple modification to encode points outside of the normalization range.

The architecture of the OP is illustrated in Fig. 3. All hyperparameters are defined in Appendix B.

3.2. Data Generation

We create a synthetic dataset of 10 deformable objects with a varying number of segments. The key features that are represented in the data are 1) high-frequency detail, 2) self-occlusion, and 3) deformations. For a detailed description of the objects, see Appendix B.

Each example in our dataset consists of a Point Cloud and Query Points. Examples are created by first randomly sampling a camera perspective and a deformed state of the object. Camera perspectives are sampled inside a spherical workspace and are always aimed at the object. The camera captures a depth image that is converted into a Point Cloud using the camera’s intrinsic parameters. Query points are generated with a sampler that scans over the deformed object and transforms them into camera space. As both the Point Cloud and the Query Points are in camera space, all reconstructions are in the camera space and therefore already registered to the camera. Each query point has a position (x, y, z) , an occupancy value o , and a distance d and

a direction (n_x, n_y, n_z) to the nearest surface. As the last step, the Point Cloud and Query Points are jointly centered and scaled. The translation t and the scaling factor s are computed using only the Point Cloud. Only the Point Cloud will be known at inference time with real-world sensor data. With c being the center of the Point Cloud:

$$t = -\frac{1}{2} (\max x_i + \min x_i, \max y_i + \min y_i, \dots) \\ s = \frac{1}{\max(|\max x_i - c_x|, |\max y_i - c_y|, \dots)} \quad (3)$$

An overview of the data generation process is given in Fig. 5.

Sampling Method for Query Points To improve surface reconstruction quality, it is desirable to concentrate query points near segment boundaries. Jia et al. [21] achieves this by sampling points on all object surfaces and adding a small random offsets along the surface normal. This method requires accurate normals in the model, which are not always available. The magnitude of the random offset also introduces an object-specific hyperparameter that needs manual tuning. Additionally, this method introduces local density biases in locally convex (over-sampled) and concave (under-sampled) regions.

We propose an algorithm, SortSample, for generating Query Points that benefit the learning of 3D reconstructions of multi-segment objects, regardless of segment thickness or surface area-to-volume ratio. SortSample samples points uniformly within each segment’s bounding box, extended by 50% in all dimensions. We separate points inside the segment (added to S_{inside}) from points in empty space (added to S_{outside}), discarding points that are outside the sampled segment but within other segments. Once S_{inside} and S_{outside} contain at least n points, we sort the lists based on their distance to the segment’s surface and select the nearest k points.

Setting $n = k$ typically includes all inside points and outside points up to a distance determined by the geometry of the object. Choosing $n = 2k$ leads to points that are closer to the surface. Using $n = 2k$ can increase the computation time needed to generate the dataset noticeably as more samples are drawn and discarded.

SortSample ensures that the generated query points are concentrated around decision boundaries. SortSample does not require hyperparameter adjustments based on scale or segment surface area-to-volume ratio and is straightforward to implement. While watertight meshes are necessary for occupancy testing, well-behaved normals are not required for SortSample. SortSample is neither locally nor globally bias for single segment objects. If there are many segments, there will still be higher point density for smaller objects, introducing a local bias towards them.

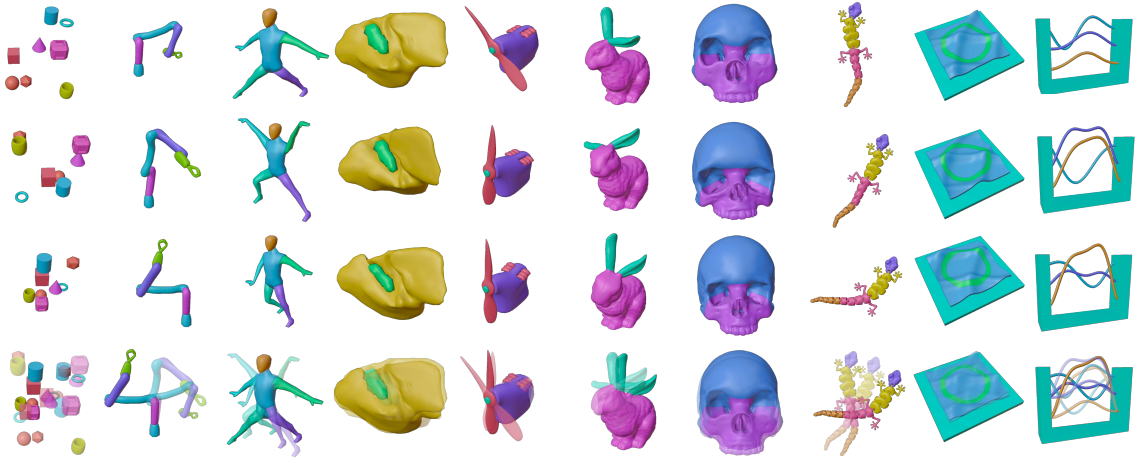


Figure 4. All scenes used to create the training and test data. Each scene contains objects with at least two moving or deforming segments, as indicated by color. In this illustration the camera is fixed. In the last row, the images of the previous rows are overlaid.

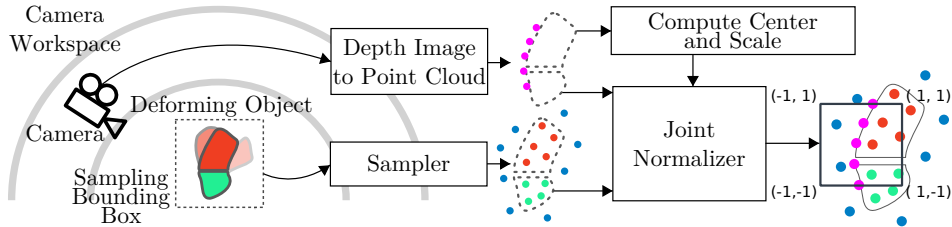


Figure 5. 2D simplification of the dataset generation process, where different camera and object configurations are captured. For each sample, the Camera is positioned randomly within a spherical workspace around the deformable object. The depth image is transformed into a Point Cloud using the Camera’s intrinsic parameters. Simultaneously, the Sampler takes points from the Sampling Bounding Box of the deformable object to create the Query Points. Each query point has a position (x, y) , occupancy value o (indicating in which segment the point resides), signed distance d and direction (n_x, n_y) to the nearest surface. Point Cloud and Query Points are then jointly scaled and re-centered to fit inside the square $[-1, 1]^2$ that encloses the observed Point Cloud.

In our final dataset, we randomly sample an additional $n_{uniform}$ points within the joint boundaries of all segments to ensure the network learns to classify regions far from the decision boundary.

4. Experiments

4.1. Overview

We evaluate different Point Cloud Encoders, loss functions, and the impact of positional encoding to find the best performing variant of our system. Performance is evaluated by Intersection over Union (IoU) to quantify reconstruction and Mean Intersection over Union (mIoU) to quantify segmentation quality. For Point Cloud Encoders, we compare PointNet++, the autoencoder of DeepSDF, and Point Transformer. For this, we extended DeepSDF to perform segmentation, see Appendix B. For the loss function, we investigate the importance of the individual components (CE, L1, and cosine) of our composite loss function. We show that PointNet++ as the Point Cloud Encoder, the loss func-

tion without the cosine component, and position encoding is the best-performing variant. Using our this best-performing variant, we evaluate 1) the reconstruction accuracy on synthetic and real-world objects, 2) the value of SortSampling, 3) the importance of frequency based positional encoding. Experiments that investigate the effect of noise on reconstruction accuracy and the metric used for measuring the reconstruction quality can be found in Appendix C and B, respectively.

4.2. Reconstructions and Segmentation of Synthetic Objects

The object reconstructions and segmentations on synthetic data are visualized in Fig. 6. Most objects are reconstructed with an mIoU of between 0.93 and 0.73, with the exception of the Rope object, where the relatively high IoU of the base segment is contrasted by the low IoU of the rope segments. IoU values are typically close to mIoU values, which indicates that the reconstruction task is more























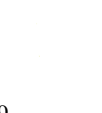


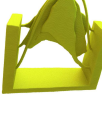
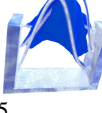
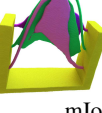

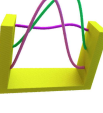
Object	Point Cloud	Reconstruction	Reconst. Error	Segmentation	Seg. Error	Reference
Human		 IoU: 0.894		 mIoU: 0.837		
Robot		 IoU: 0.882		 mIoU: 0.864		
Stanford Bunny		 IoU: 0.971		 mIoU: 0.917		
Lizard		 IoU: 0.752		 mIoU: 0.749		
Rope		 IoU: 0.815		 mIoU: 0.375		

Figure 6. Examples of reconstructions generated by conditioning on the input **Point Cloud**. The **Reconstruction** is the sum of all points that are not classified as empty space. **Reconstruction Error** identifies **over-reconstruction** and **under-reconstruction** when compared with the reference. **Segmentation** colors each predicted class. **Segmentation Error** between predicted **Segmentation** and **Reference**. IoU and mIoU values are averaged over all examples in the test data, not just the rendered examples. Remaining objects are listed in Appendix A.

challenging than the segmentation. Despite deformations, occlusions, and complex geometries, the system produces highly accurate 3D reconstructions.

4.3. Reconstruction and Segmentation of Real-World Objects

The object reconstructions and segmentations on real-world data are visualized in Fig. 7. We test the system’s capability to reconstruct real-world objects using the Lizard object (all attributions in Appendix B). Lizard was chosen due to its ability to be 3D printed and deformed while also allowing manual registration of its rigid segments. We capture Point Clouds of 15 deformed states with a Zivid One+ (Zivid, Norway) camera. Background points that are not part of the lizard are removed and the point density is reduced from approximately 600,000 to around 600 points with the Subsampling method of the open-source application CloudCompare. For IoU calculation, we manually register the original 3D model to the full-resolution Point Cloud, aligning each rigid segment individually. Our system is trained on synthetic data and tested on real world data, which yielded a mIoU of 0.526 and an IoU of 0.530. These scores were considerably lower than the 0.752 and

0.759 on synthetic data. Although the network accurately managed the overall reconstruction and segmentation, it struggled to precisely reconstruct the segments.

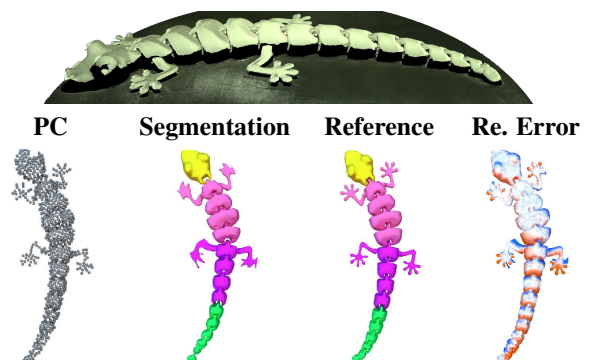


Figure 7. Example of a reconstruction generated by using a sub-sampled version **PC** of a real-world point cloud (top) as input to the network trained on synthetic data. The lizard is approximately 30 cm long. **Re. Error** identifies **over-reconstruction** and **under-reconstruction** when compared with the reference. All reconstructions are shown in Appendix D.

4.4. Variants and Ablations

Point Cloud Encoder Architecture and Loss Function

The mIoU and IoU of different variants of PCEs and loss functions are shown in Figure 8.

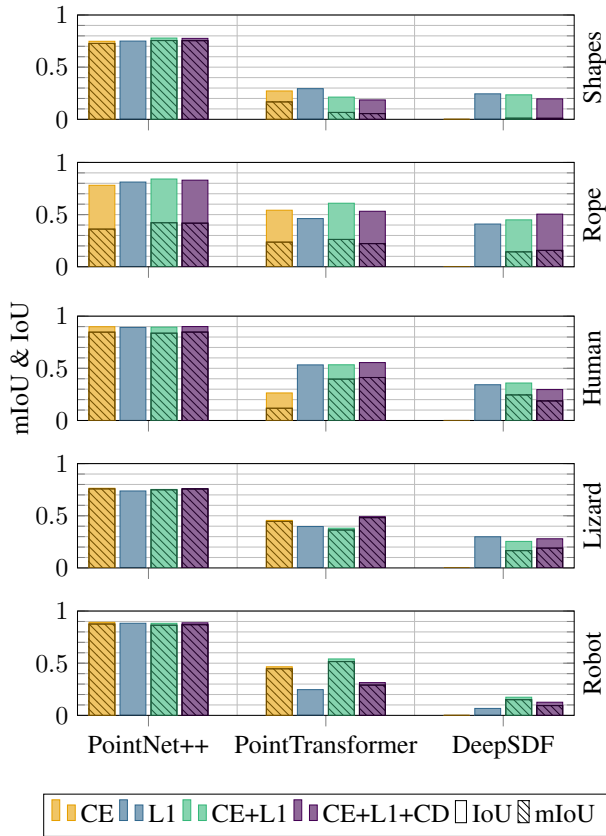


Figure 8. IoU and mIoU for each model architecture and loss function on selected scenes. It is not possible to train the DeepSDF with only CE loss, so these values are omitted. The L1 loss does not optimize class probabilities required for mIoU, so these values are also omitted.

The factor with the strongest influence on both mIoU and IoU is the Point Cloud Encoder. PointNet++ significantly outperformed both Point Transformer and DeepSDF’s autodecoder. PointNet++ produces high-frequency details present in the training data. The autodecoder of DeepSDF [30] proved to be sensitive to both the type of object and the hyperparameters, and required considerably more epochs to train (1000 vs. 300).

Figure 9 shows that all of these variants are able to infer the overall structure of objects.

The loss function has a smaller effect on the reconstruction quality. For PointNet++, adding the L1 component to the loss increases the mIoU to 0.422 compared to 0.361 with CE loss alone for the Rope object. The benefit is less

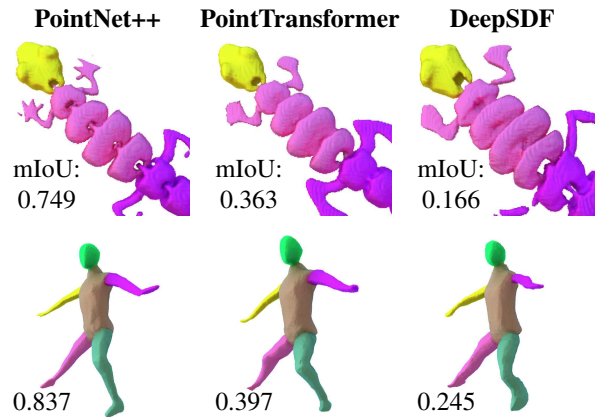


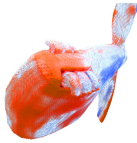
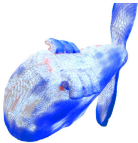
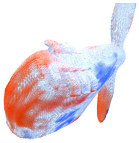
Figure 9. Comparison of reconstructions of the Lizard (top row) and Human (bottom row) scenes provided by different models trained with CE+L1 loss. The significant difference in mIoU between the models is due to the high-frequency details, which PointNet++ is better able to capture. All models are able to infer the overall structure of the scenes.

pronounced for other objects. When using only CE loss in combination with positional encoding, floating periodic artifacts far from the actual reconstruction can be observed. CE loss operates on class label probabilities and does not inherently enforce boundedness. In contrast, L1 loss penalizes large prediction errors and thus tends to encourage more bounded outputs. Our results of adding a cosine distance (CD) term to the loss also align with those of Lamb et al. [23] that report a small benefit. However, we argue that the minor advantage in quality is outweighed by the additional computation and memory requirements. When using only the L1 loss, the reconstruction is not segmented. DeepSDF [30], see DeepSDF with L1 in Figure 9, was outperformed by a large margin by PointNet++ with L1 loss in this task. Overall, CE+L1 loss produces more consistent results compared to either CE or L1 loss in isolation. In summary, the best variant uses a PointNet++ as Point Cloud Encoder and is trained on the CE+L1 loss with enabled Position Encoding.

Query Point Generation We investigate the effect of different sampling methods to generate Query Points. *Volume Uniform* sampling draws points uniformly within the extended bounding box of each segment. 256 samples are drawn for each segment. *Label Uniform* sampling generates points uniformly inside the segment, and then again uniformly outside the segment within the extended bounding box. 256 samples are drawn for each segment, 128 samples inside and 128 samples outside.

The different methods are illustrated in Fig. 10. In contrast to Volume and Label Uniform, SortSample does not

Table 1. Reconstruction error, IoU, and mIoU using 3 sampling methods. IoU and mIoU are averaged over all examples in the test data. The first row shows **over-reconstruction** and **under-reconstruction** for the Engine object.

Method →	Volume Unif.	Label Unif.	SortSample		
Engine					
Engine	0.926	0.698	0.915	0.686	0.931 0.729
Human	0.849	0.778	0.800	0.713	0.894 0.837
Lizard	0.672	0.676	0.679	0.681	0.752 0.749
Object ↑	IoU	mIoU	IoU	mIoU	IoU mIoU

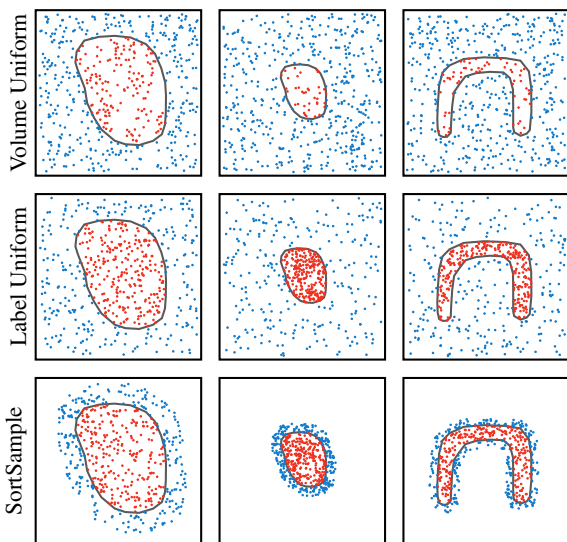


Figure 10. Sample distributions using different sampling methods on 2D toy scenes. Only SortSample results in a dataset that is unbiased between red and blue classes both globally and locally near object boundaries.

cause local or global biases.

Tab. 1 shows IoU and mIoU for Engine, Human, and Lizard using all 3 sampling methods. SortSample results in datasets that promote improved reconstruction quality. The improvement is most noticeable for more complex objects such as the Lizard. Volume Uniform sampling outperforms Label Uniform sampling as it avoids introducing a local bias in point density at the boundaries. The rendered reconstruction shows that Label Uniform sampling results in a *ballooning* effect (over-reconstruction) of structures.

Positional Encoding The results of using a frequency-based encoding are shown in Tab. 2. The use of positional encoding improves the quality of reconstructions by an av-

erage of 2%. The results indicate that objects with more intricate details such as Lizard benefit from the positional encoding, while less detailed objects such as Human and Robot do not.

Table 2. mIoU with positional encoding (PE) and without.

Scene	Shapes	Rope	Human	Lizard	Robot
PE	0.791	0.375	0.837	0.749	0.864
Without	0.772	0.354	0.835	0.737	0.860

5. Discussion and Conclusion

In this work, we present a system for registered and segmented 3D reconstruction of deformable objects from single-view point clouds. We introduce a simple sampling method for generating Query Points without hyperparameter tuning. This sampling method produces suitable training data regardless of the surface-area-to-volume ratio of object segments. Furthermore, we present a system for learning segmented objects implicitly through a multi-class occupancy function. The system supports using an arbitrary neural network as a point cloud encoder. The system is evaluated on synthetic data of a novel suite of 10 deformable objects and data from a real-world experiment. The system is able to reconstruct occluded and high-frequency features. Segments are clearly defined and there are only minimal segmentation errors. Trained on synthetic data, the system is able to reconstruct real-world objects with some degradation in performance (mIoU of 0.526 versus 0.752). PointNet++ considerably outperforms Point Transformer and the autoencoder of DeepSDF as Point Cloud Encoders. Combining cross-entropy loss (Occupancy) with L1 distance loss (SDF) is superior to either loss function in isolation. Positional encoding benefits objects with high frequency details, but we argue that the evaluated objects did not contain enough details to make full use of it.

A major limitation of the proposed method is the requirement for watertight meshes for occupancy testing using ray casting. This prohibits the use of some publicly available mesh datasets. In addition, realistic deformations must be modeled by hand or simulated. Future work will address the ability of a single model to generalize across many objects.

Acknowledgements This contribution is supported by the Helmholtz Association under the joint research school "HIDSS4Health – Helmholtz Information and Data Science School for Health" and the Helmholtz Association's Initiative and Networking Fund on the HAICORE@KIT partition. We wish to acknowledge Vaisakh Shaj and Jan-Philipp Kaiser from the Karlsruhe Institute of Technology for their support.

References

- [1] Matan Atzmon, Haggai Maron, and Yaron Lipman. Point convolutional neural networks by extension operators. *ACM Trans. Graph.*, 37(4), July 2018.
- [2] Abhishek Badki, Orazio Gallo, Jan Kautz, and Pradeep Sen. Meshlet priors for 3d mesh reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2849–2858, 2020.
- [3] Yingjie Cai, Xuesong Chen, Chao Zhang, Kwan-Yee Lin, Xiaogang Wang, and Hongsheng Li. Semantic Scene Completion via Integrating Instances and Scene In-the-Loop. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 324–333, 2021.
- [4] Rohan Chabra, Jan E Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard Newcombe. Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. In *Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16*, pages 608–625. Springer, 2020.
- [5] Zhiqin Chen, Andrea Tagliasacchi, and Hao Zhang. Bsp-net: Generating compact meshes via binary space partitioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 45–54, 2020.
- [6] Julian Chibane, Mohamad Aymen mir, and Gerard Pons-Moll. Neural Unsigned Distance Fields for Implicit Function Learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 21638–21652. Curran Associates, Inc., 2020.
- [7] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *Computer Vision—ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part VIII 14*, pages 628–644. Springer, 2016.
- [8] Boyang Deng, Kyle Genova, Soroosh Yazdani, Sofien Bouaziz, Geoffrey Hinton, and Andrea Tagliasacchi. Cvxnet: Learnable convex decomposition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 31–44, 2020.
- [9] Theo Deprelle, Thibault Groueix, Matthew Fisher, Vladimir Kim, Bryan Russell, and Mathieu Aubry. Learning elementary structures for 3d shape generation and matching. *Advances in Neural Information Processing Systems*, 32, 2019.
- [10] Mark Everingham, S. M. Ali Eslami, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The Pascal Visual Object Classes Challenge: A Retrospective. *International Journal of Computer Vision*, 111(1):98–136, Jan. 2015.
- [11] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017.
- [12] Matheus Gadelha, Rui Wang, and Subhransu Maji. Deep manifold prior. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1107–1116, 2021.
- [13] Jun Gao, Wenzheng Chen, Tommy Xiang, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Learning deformable tetrahedral meshes for 3d reconstruction. *Advances in Neural Information Processing Systems*, 33:9936–9947, 2020.
- [14] Rohit Girdhar, David F Fouhey, Mikel Rodriguez, and Abhinav Gupta. Learning a predictable and generative vector representation for objects. In *Computer Vision—ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part VI 14*, pages 484–499. Springer, 2016.
- [15] Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh r-cnn. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9785–9795, 2019.
- [16] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3d surface generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 216–224, 2018.
- [17] Oshri Halimi, Ido Imanuel, Or Litany, Giovanni Trappolini, Emanuele Rodolà, Leonidas Guibas, and Ron Kimmel. The whole is greater than the sum of its nonrigid parts. *arXiv preprint arXiv:2001.09650*, 2020.
- [18] Rana Hanocka, Amir Hertz, Noa Fish, Raja Giryes, Shachar Fleishman, and Daniel Cohen-Or. Meshcnn: a network with an edge. *ACM Transactions on Graphics (TOG)*, 38(4):1–12, 2019.
- [19] Rana Hanocka, Gal Metzer, Raja Giryes, and Daniel Cohen-Or. Point2mesh: A self-prior for deformable meshes. *arXiv preprint arXiv:2005.11084*, 2020.
- [20] Ka-Hei Hui, Ruihui Li, Jingyu Hu, and Chi-Wing Fu. Neural template: Topology-aware reconstruction and disentangled generation of 3d meshes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18572–18582, 2022.
- [21] Meng Jia and Matthew Kyan. Learning Occupancy Function from Point Clouds for Surface Reconstruction. *arXiv:2010.11378 [cs]*, Oct. 2020.
- [22] Meng Jia and Matthew Kyan. Improving Intraoperative Liver Registration in Image-Guided Surgery with Learning-Based Reconstruction. *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1230–1234, June 2021.
- [23] Nikolas Lamb, Sean Banerjee, and Natasha Kholgade Banerjee. Deepjoin: Learning a joint occupancy, signed distance, and normal field function for shape repair. *ACM Transactions on Graphics (TOG)*, 41(6):1–10, 2022.
- [24] Jie Li, Yu Liu, Dong Gong, Qinfeng Shi, Xia Yuan, Chunxia Zhao, and Ian Reid. RGBD Based Dimensional Decomposition Residual Network for 3D Semantic Scene Completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7693–7702, 2019.
- [25] Chen-Hsuan Lin, Chen Kong, and Simon Lucey. Learning efficient point cloud generation for dense 3d object reconstruction. In *proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.
- [26] Or Litany, Alex Bronstein, Michael Bronstein, and Ameesh Makadia. Deformable shape completion with graph con-

- volutional autoencoders. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1886–1895, 2018.
- [27] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2837–2845, 2021.
- [28] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy Networks: Learning 3D Reconstruction in Function Space. *arXiv:1812.03828 [cs]*, Apr. 2019.
- [29] 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. In *Proceedings of European Conference on Computer Vision (ECCV)*, Aug. 2020.
- [30] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019.
- [31] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.
- [32] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
- [33] Christoph B Rist, David Emmerichs, Markus Enzweiler, and Dariu M Gavrilă. Semantic scene completion using local deep implicit functions on lidar data. *IEEE transactions on pattern analysis and machine intelligence*, 44(10):7205–7218, 2021.
- [34] Luis Roldão, Raoul de Charette, and Anne Verroust-Blondet. 3D Semantic Scene Completion: A Survey. *International Journal of Computer Vision*, 130(8):1978–2005, Aug. 2022.
- [35] Tianchang Shen, Jun Gao, Kangxue Yin, Ming-Yu Liu, and Sanja Fidler. Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis. *Advances in Neural Information Processing Systems*, 34:6087–6101, 2021.
- [36] Vincent Sitzmann, Eric Chan, Richard Tucker, Noah Snavely, and Gordon Wetzstein. Metasdf: Meta-learning signed distance functions. *Advances in Neural Information Processing Systems*, 33:10136–10147, 2020.
- [37] Shuran Song, Fisher Yu, Andy Zeng, Angel X. Chang, Manolis Savva, and Thomas Funkhouser. Semantic Scene Completion From a Single Depth Image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1746–1754, 2017.
- [38] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. *arXiv:2006.10739 [cs]*, June 2020.
- [39] Shubham Tulsiani, Tinghui Zhou, Alexei A Efros, and Jitendra Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2626–2634, 2017.
- [40] Francis Williams, Zan Gojcic, Sameh Khamis, Denis Zorin, Joan Bruna, Sanja Fidler, and Or Litany. Neural fields as learnable kernels for 3d reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18500–18510, 2022.
- [41] Francis Williams, Teseo Schneider, Claudio Silva, Denis Zorin, Joan Bruna, and Daniele Panozzo. Deep geometric prior for surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10130–10139, 2019.
- [42] Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. Point transformer v2: Grouped vector attention and partition-based pooling. *arXiv preprint arXiv:2210.05666*, 2022.
- [43] Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Pu-net: Point cloud upsampling network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2790–2799, 2018.
- [44] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 16259–16268, 2021.
- [45] Linqi Zhou, Yilun Du, and Jiajun Wu. 3d shape generation and completion through point-voxel diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5826–5835, 2021.