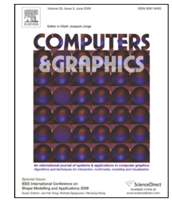
Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag



3DCascade-GAN: Shape Completion from Single-View Depth Images

Fahd Alhamazani[a], Yu-Kun Lai[a], Paul L. Rosin[a]

[a] *Cardiff University, Abacws Building, Cardiff, CF24 4AG, UK*

ARTICLE INFO

Article history:
Received July 19, 2023

Keywords: Depth images, Shape completion, Generative adversarial network, Self-attention, Multi-task learning

ABSTRACT

Depth images can be easily acquired using depth cameras. However, these images only contain partial information about the shape due to unavoidable self-occlusion. Thanks to the availability of large datasets of shapes, it is possible to use a learning-based approach to produce complete shapes from single depth images. State-of-the-art generative adversarial network (GAN) architectures can produce reasonable results. However, the use of relatively local convolutions restricts GAN architectures from producing globally plausible shapes. In this study, we develop a novel dynamic latent code selection mechanism in which the model learns to select only important codes from the latent space. Furthermore, a novel 3D self-attention (3DSA) layer is introduced that is able to capture non-local relationships across the 3D space. We further design a GAN architecture that uses a multistage encoder-decoder to recover the shape, where our 3DSA layer is introduced to the discriminator to help attend to global features, which stabilizes the model learning and encourages shape refinement, making our reconstruction more structurally plausible. Through extensive experiments, we demonstrate that our method outperforms other state-of-the-art methods for single depth image 3D reconstruction.

1. Introduction

Many tasks of modern technology, such as robotic vision and obstacle avoidance, rely heavily on 3D reconstruction for which depth images are a common source of data. Until recently, capturing depth information was challenging, but with the availability of low-cost depth cameras, depth images can now be quite easily obtained, allowing datasets to be created [1] that make possible novel applications such as virtual reality (VR) [2, 3]. However, estimating the full 3D shape from a depth image, which only represents one viewpoint, is still challenging. Since a depth image only contains partial information about the shape due to unavoidable self-occlusion [4], shape completion is naturally present as part of many 3D application pipelines, e.g., SLAM [5], robot grasping [6] or autonomous driving [7]. A single depth image may not be sufficiently descriptive to fully reconstruct a shape, causing holes and spurious surfaces in the reconstruction. Ideally a system should be able to cope with such difficult or unusual viewpoints. The alternative, capturing sufficient depth maps to form complete 3D data, is not feasible for many real-world applications due to the increase in cost and time. For example, in indoor scene modeling, capturing complete furniture would be near-impossible due to substantial occlusion.

Our work focuses on reconstructing a 3D shape from a single depth image using a 3D convolution neural network (CNN). The CNN approach shows impressive results compared to other non-learning-based models [8, 9, 10] where the bounding ray cone or voxel hashing are used. Non-learning models usually require multiple viewpoints of the shape, while the learning-based models can learn from existing full shapes to reconstruct complete shapes from single depth images [11, 12], or single RGB images [13, 14, 15].

In this work, we present a model capable of producing a com-

e-mail: fahda@cardif.ac.uk (Fahd Alhamazani), laiy4@cardiff.ac.uk (Yu-Kun Lai), rosinpl@cardiff.ac.uk (Paul L. Rosin)

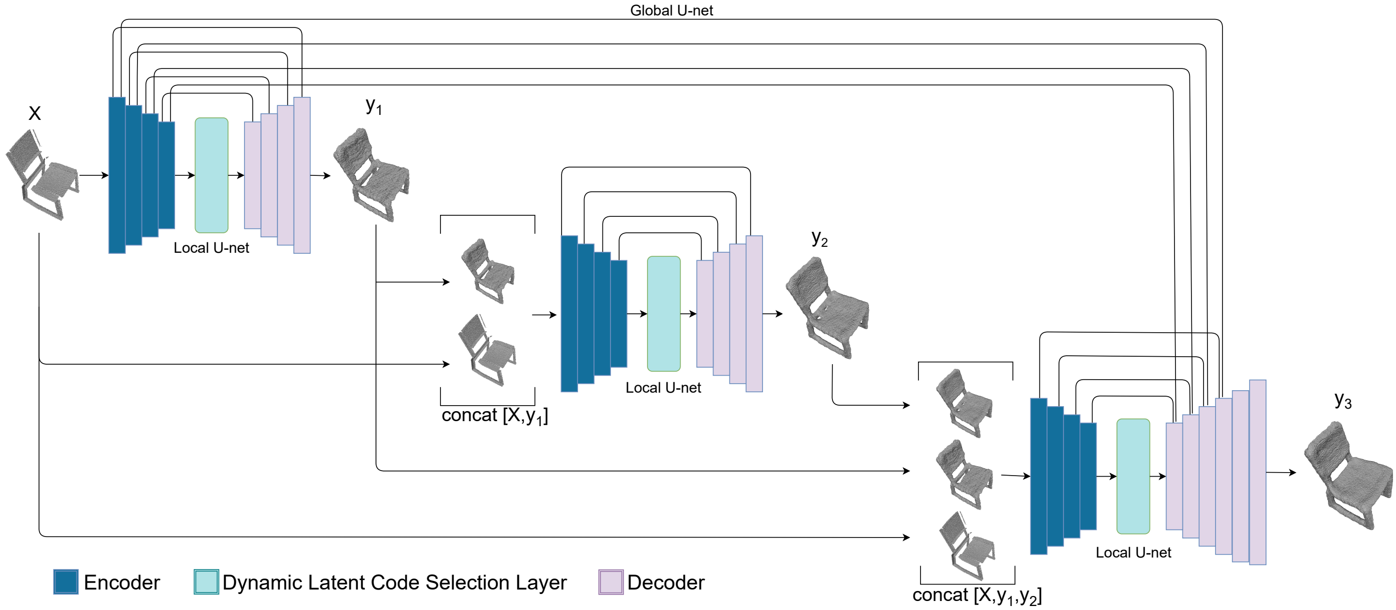


Fig. 1. The generator turns an input volume from a depth image to a high-resolution 3D volumetric output.

plete shape from a single depth image. Given a 2.5D depth image as input, the model can learn to reconstruct a high resolution shape. As shown in Figures 1 and 2, an end-to-end learning model containing a sequence of multiple encoder-decoders with global and local skip links is trained to complete the volumetric shape, where the later stages take both the input and outputs from previous stages to further improve completion. We also introduce a self-attention layer that helps refine the 3D shapes, mimicking the human ability to focus on a region of interest in the volumetric space. In addition, if a 3D shape is missing certain features (e.g., due to occlusion), self-attention aids in improving its details by exploiting clues from non-local regions. Such non-local information is useful as only partial single-view depth is given. For example, the geometry of one table leg gives a useful clue for reconstructing the other table legs. We further introduce a dynamic latent space where the model has the ability to select only relevant codes to estimate 3D shapes. As we will later demonstrate, this strategy provides a strong sparse regularization that improves the robustness. Furthermore, we extend the shape completion to a multi-task setting, where the generated shape is further classified into one of the object categories, as shown in Figure 3. As properly completed shapes are easier to classify, these two tasks help with each other, contributing to improved shape completion results.

Our contributions are:

- We propose a cascade architecture consisting of multiple encoder-decoder blocks with additional skip links, which provides better 3D reconstruction than a single encoder-decoder.
- We incorporate a self-attention layer to refine the 3D shapes, mimicking human ability to focus on a region of interest in the volumetric space.
- We introduce a dynamic latent space where the model has the ability to select only relevant latent codes to estimate 3D shape. This provides a strong sparse regularization that enhances the robustness of the network.
- A classifier network is introduced as an auxiliary task to provide additional guidance to the reconstruction model.

Extensive experiments show that our method outperforms state-of-the-art methods.

2. Related Work

Our work reconstructs a complete 3D shape from a single depth image, so we review related papers which use either a single RGB or depth image as input to reconstruct a 3D object. This is a challenging problem, and has received significant attention in recent years. Reconstructing 3D shapes from single RGB images requires addressing the domain differences, as it can be difficult to obtain training data in both domains. Yan et al. [16] built a model that uses RGB images as input. The authors generate the dataset inputs by using projection (i.e. rendering). The projection was made from 24 different angles. Furthermore, the network model contains a 2D encoder and a 3D decoder, and the authors add a transformer layer to get target projection. However, the model results in shapes that are of low resolution. Yu et al. [17] took multiview images as input. They estimated a depth image for each input image. After that, they reconstructed a coarse volumetric shape by fusing multiview depth images, and then utilized a refinement model to reconstruct a high-resolution shape.

Xie et al. [18] took a similar approach, but the model has a fused network where high-quality parts are selected and fused. By applying a differentiable renderer on the reconstructed shape, Huang et al. [19] found nearest neighbor images from the dataset to semantically enhance the reconstructions. Wu et al. [13] employed synthetic data as ground truth to disentangle unwanted features like color and texture. After that, the model is fine-tuned on realistic appearance images to improve

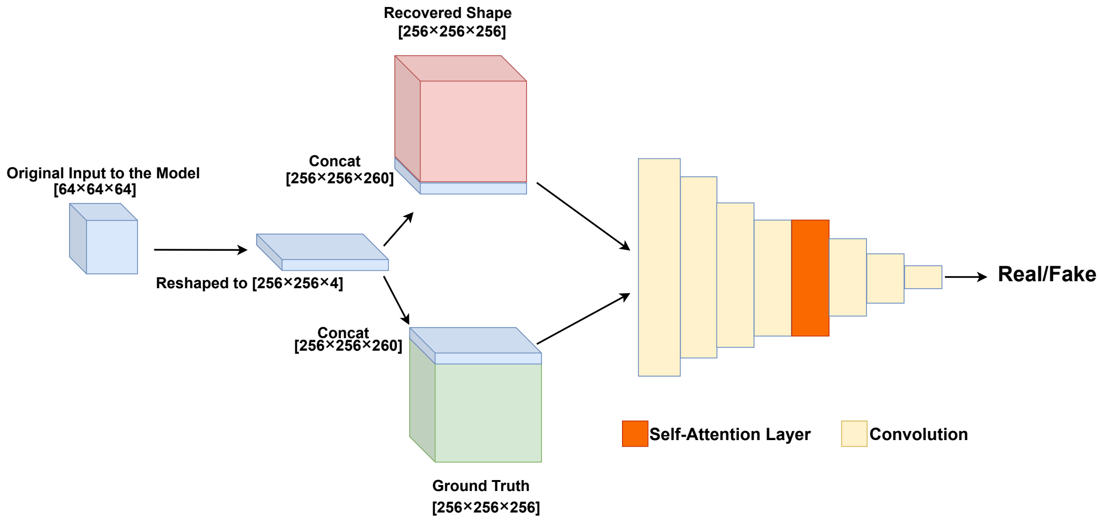


Fig. 2. The discriminator takes the concatenation of the original single view volume and either the ground truth or the reconstructed shape as its input. We also introduce a 3D self-attention layer to the discriminator to improve the generated shape.

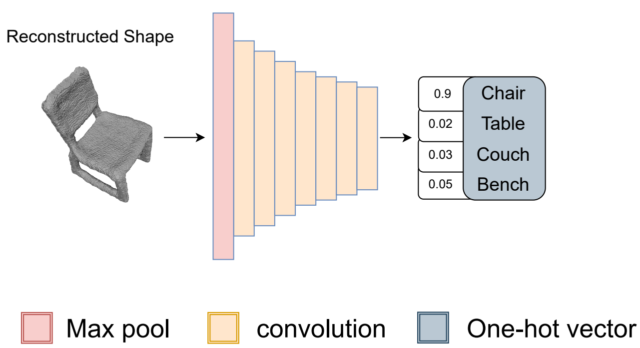


Fig. 3. The classifier that classifies the type of the shape helps the generator to produce shapes with proper structure and details to improve the chance of correct classification.

its performance. Zhang et al. [14], on the other hand, used a depth estimator as a middle step before generating a 3D shape, in a way similar to [13] but with skip links used for shape refinement. Wu et al. [15] estimated the 2.5D depth image from a given 2D image before reconstructing a full 3D shape. They proposed to penalize the reconstructed shape according to the lack of realism of its appearance. Xian et al. [20] also estimated multi-depth images as an intermediate step, and then projected the depth images to a point cloud followed by voxelization. Hui et al. [21] estimated topology as a step before predicting a mesh. Hafiz et al. [22] took a different approach, using a single encoder and multiple decoders to predict point clouds from multiple viewpoints, which were then fused to obtain a complete shape.

The approach Kurenkov et al. [23] investigated was reconstruction through deformation. The authors suggested retrieving the closest shape from the dataset to the given input image. Then both the image and the retrieved shape are used as input for the model. The output of the model is a vector containing an offset of control points for free-form deformation (FFD). Kanazawa et al. [24] demonstrated deforming a mesh shape based on an image collection as ground truth rather than a 3D shape. Their model also learns to find the keypoints used for mapping the input texture. Wang et al. [25] worked on deforming a mesh driven by a single image; the model consists of three blocks: the first block deforms an ellipsoid mesh and each following block completes the deformation by increasing the number of vertices.

Wen et al. [26] also deformed an ellipsoid. However, features extracted were split to edge features and local features, and the edge features were used to deform the ellipsoid to a coarse shape while local features were for refining the shape.

Richter and Roth [27] built a 3D shape from a single image where the method reconstructs a low resolution model, along with depth images for each higher resolution. The shape is then obtained through the fusion of those images. Peng et al. [28] utilized a transformer for each view's latent codes before fusing. Lin et al. [29] generated 3D data from multiple viewpoints of an image by using an image encoder and a 3D decoder, which are then combined to produce a complete shape. In addition to using a 2D-encoder and a 3D-decoder, Gao et al. [30] also trained a 3D autoencoder to concatenate the latent codes for enhanced reconstruction. In the works of [31, 32] a single image is used as input and a mixed dataset of labeled and unlabeled samples for training. Robert et al. [32] employed two models, each one is responsible for reconstructing a partial shape, while Jiang et al. [33] introduced two losses: a geometric loss that forces each view of the reconstructed shape to be close to the ground truth, and an adversarial loss that is responsible for finding the differences in the output and the ground truth. Gwak et al. [34] addressed an ill-posed problem which takes one or more views of the shape as input, and through adversarial learning, it aims to make the shape more plausible rather than with fine details. To produce higher resolution shapes, some works utilize space partitioning data structures such as octrees. Given an input image, Tatarchenko et al. [35] used an octree as the output of a CNN, which is able to reconstruct high-resolution (up to 512^3)

voxel grids.

Hane et al. [11] used an octree to represent the boundaries of the shape, which they first reconstructed at a low resolution and then refined using a “block octree”. Wang et al. [36] took as input an incomplete point cloud represented by an octree. Due to incomplete input and the nature of octree representation, the authors add dynamic skip connections, which leads to improved performance. The work [12] instead reconstructed a shape by giving a single depth image to the model with an adversarial component for the purpose of refinement. These methods are capable of generating high-resolution 3D shapes. However, the generated shapes may still suffer from incorrect structure and/or geometry, because these methods largely depend on convolution layers which only capture local information. To produce appropriate reconstruction from partial single-view information, non-local relationships between locations are essential. This however is not considered in previous single depth image 3D reconstruction works.

Some works address 3D shape completion with more general partial input, although they can also be applied to cope with single-depth input as a special case. Hu et al. [37] leveraged a generator to complete shapes where the model renders multi-view depth images and pools across all outputs. Wang et al. [38] proposed to use a GAN model to reconstruct coarse shapes, followed by refinement to match the ground truth while Huang et al. [39] completed shapes implicitly by generating latent vectors of depth shapes. However, both [38] and [39] suffer from geometric inconsistency. Wen et al. [40] addressed the issue by adding folding-block and skip attention where the features' locations are matched against the input.

In the work [41] they implemented parallel models for complete and incomplete shapes where the models share weights during training to preserve geometric consistency. However, the models may not work well for unseen objects. ForkNet [42] addresses this issue, and the model consists of three parallel generators with shared latent features. Two branches reconstruct the SDF (Signed Distance Field) representation and complete the surface respectively, while the third branch concatenates features from both previous reconstruction branches to semantically complete the volume scene. Park et al. [43] also suggested using an SDF, where the input is a latent code concatenated with 3D point locations to elevate a high dimensional representation. At first, the model optimizes the weights and the latent code to generate plausible SDF values while during inference, the model optimizes latent code to generate an appropriate SDF.

Wu et al. [44] claimed that the Chamfer Distance is not sensitive to outliers, and queries for nearest points could make the model unaware of the shape density. So they also added a discriminator that separates the points to form groups based on the shape surface. Alliegro et al. [45] introduced a contrastive model. They utilized pretrained encoders to capture semantic information and geometry features. The model naturally completes the missing parts, Li et al. [46] leveraged a transformer to extract meaningful features. The model generates features for both partial and complete shapes, and learns to complete a shape by matching partial to complete features. Chen et al. [47] proposed to locate anchor points instead of generating them. The network learns to locate sparse points that capture global features. Wang et al. [48] sorted generated latent features based on activation scores, and the sorted features were then utilized to reconstruct a complete shape. Zhang et al. [49] suggested using k-nearest neighbor points to capture local features before using an MLP (Multi-layer Perceptron) to generate the latent features.

Some methods achieve 3D reconstruction by locally deforming 2D planar patches to provide local structures. Yang et al. [50] suggested extracting features of a point cloud to guide the model to deform 2D planes. On the other hand, Wei et al. [51] believed the randomness of the 2D plane generation could introduce noise to the complete shape. To address this, they added rules for generating the planes, which could enhance the deformation and reconstruction. Xiao et al. [52] proposed to use folding blocks on latent features to enhance the reconstruction for regions with missing points.

Previously described methods require paired data of incomplete/complete shapes for supervision during training. Alternatively, some unsupervised models try to avoid such explicit supervision. Zhang et al. [53] generated full 3D shapes in an unsupervised manner through Generative Adversarial Network (GAN) inversion.

Given a pre-trained GAN for complete shape generation, the method tries to optimize the latent code for the GAN such that it produces a complete shape that matches the partial input. To achieve this, the generated complete shape goes through a degradation function to retain partial points that match the input based on k-nearest neighbors, and both Chamfer Distance and Feature Distance are used to measure the differences between the degraded and the input shapes, which in turn optimizes the latent code through gradient descent. The method can achieve similar performance as supervised approaches.

In this paper, we address the problem of 3D completion from single-view depth input. We introduce a 3D self-attention (3DSA) layer and develop a GAN-based framework including the 3DSA layer in the discriminator which effectively improves the performance of 3D reconstruction. We also present a novel dynamic latent space, that can learn to weight latent features and select important latent dimensions. Furthermore, the model consists of multiple stages where the next stage further refines prediction from the previous stage.

3. 3DCascade-GAN

Our model addresses the problem of reconstructing a 3D shape from a single depth image where the 3D space is voxelized. The voxel representation provides flexibility for topological change, which is required when turning the depth image into a complete 3D shape. A cascade approach was adopted in which shape estimation was enhanced at each stage of the model. In addition, instead of passing the entire latent vector, we suggest a selection process to dynamically select appropriate latent codes. Furthermore, self-attention has the ability to find links between features; the self-attention layer works globally on the whole space while convolution works on the local region with the volume occupancy represented by 1 for occupied and 0 for unoccupied.

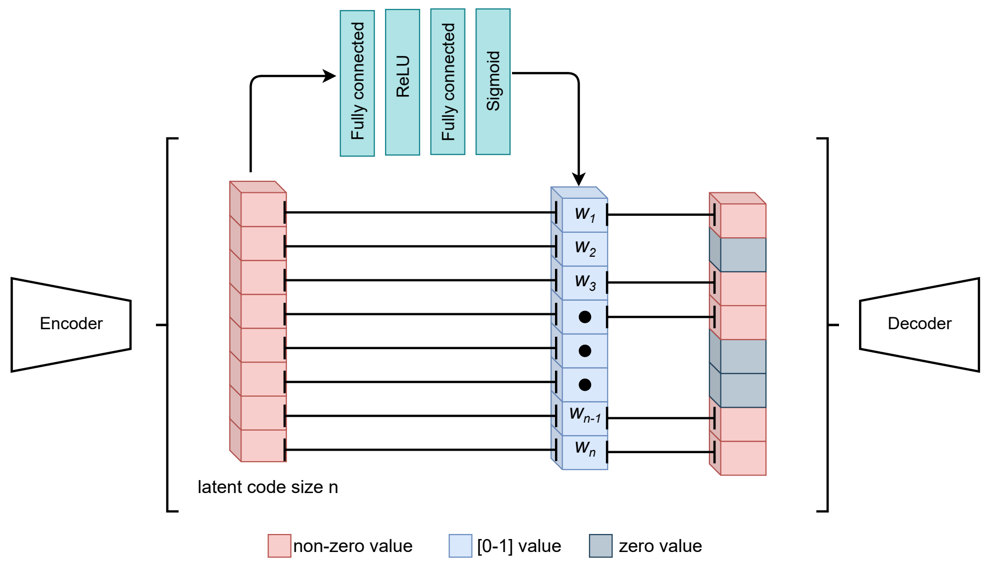


Fig. 4. The n-dimensional latent code is first processed by two fully connected layers to predict an n-dimensional weight vector. Then the top K codes are selected according to the weight vector and values in the remaining dimensions are set to zero, leading to a sparsified latent space.

Our model takes 64^3 voxels representing the input depth image and reconstructs the 3D shape sampled to 256^3 voxels to retain more details.

3.1. Network architecture

Our 3DCascade-GAN consists of two components: the generator and discriminator. Figures 1, 2 and 3 show the complete network architecture where Figure 1 is the multistage encoder-decoder (generator), Figure 3 is the classifier and Figure 2 is the discriminator.

Generator. The generator is multistage (three stages), and each stage is an identical encoder-decoder-like network (except the last stage where we add two up-sampling layers). The encoder contains four 3D CNN layers starting with an input X that is 64^3 in size (the depth view of the shape); the kernel size for each layer of $4 \times 4 \times 4$, and $1 \times 1 \times 1$ strides. Each layer uses a leaky ReLU activation function, and after each convolution layer, a max pooling layer with a kernel size of $2\times2\times2$ follows $2\times2\times2$ strides; the size of the feature maps for each layer is 64, 128, 256 and 512, respectively, followed by a fully connected layer to map the higher abstraction of the shape and generate a 1000-dimensional latent code. Before the decoder runs, a selector layer processes the latent vector to select the top K codes, where K is set to 100 (for different K values, see the Dynamic Latent Code and the ablation sections). Another fully connected layer is then introduced which generates a 512-dimensional feature map. The decoder consists of four layers of transpose convolution with each layer followed by a ReLU. Skip links are used between the encoder and decoder where feature maps are concatenated; skip links enhance the shape details, as the latent code appears to preserve the general structure of shape without any fine details. No max pooling is used in the decoder; however, a kernel size of $4 \times 4 \times 4$ and $2 \times 2 \times 2$ strides is used, and each layer is followed by a ReLU except for the last layer where we used sigmoid. Note, the third stage has extra up-sampling layers so as to reconstruct to 64^3.

We concatenate both the output y_1 and the original input X at the feature channel to form $64^3 \times 2$, which will be the input for stage two. The process is also repeated for stage three, where the input is a concatenation of stage one y_1 and stage two y_2 and the original input X, the concatenated input size is $64^3 \times 3$. We found that the model tends to rely heavily on stages two and three, and consequently the output at stage one could be fragmented and not useful. To address this issue, we added global skip links between the encoder in stage one and the decoder in stage three.

Discriminator. The discriminator is useful to ensure the completion of the partial input shape. The input for the discriminator is either a fake pair (2.5D and the recovered shape) or a real pair (2.5D and ground truth). Again, the component contains seven 3D convolution layers. Each layer has a kernel size of $4 \times 4 \times 4$ and strides of $2 \times 2 \times 2$. At the end of each layer, a ReLU activation function is used; however, the last layer consists of a sigmoid to generate a semantic representation of the shapes. Finally, we applied the strategy of [12] by outputting the mean of a vector feature rather than a scalar in order to stabilize training because the discriminator cannot discriminate high dimension data (the input concatenated with either ground truth or the reconstructed shape) and the model usually collapses at an early stage. Our 3DSA layer is introduced to capture non-local relationships.

Classifier. The classifier network consists of 7 CNN layers each with kernel size of $4\times4\times4$ and $1\times1\times1$ strides. Each layer is followed by max pooling layers with kernel size of $2 \times 2 \times 2$ follows $2 \times 2 \times 2$. For the activation function, we use Leaky ReLU. The resulting output is reshaped to form a 4 element vector representing the categories {chair, bench, table, couch}, followed by a softmax layer to reconstruct the one-hot vector. It was not necessary to use the full 256^3 resolution as input to the classifier, and so we applied max pooling to reduce the input dimensions to 64^3.

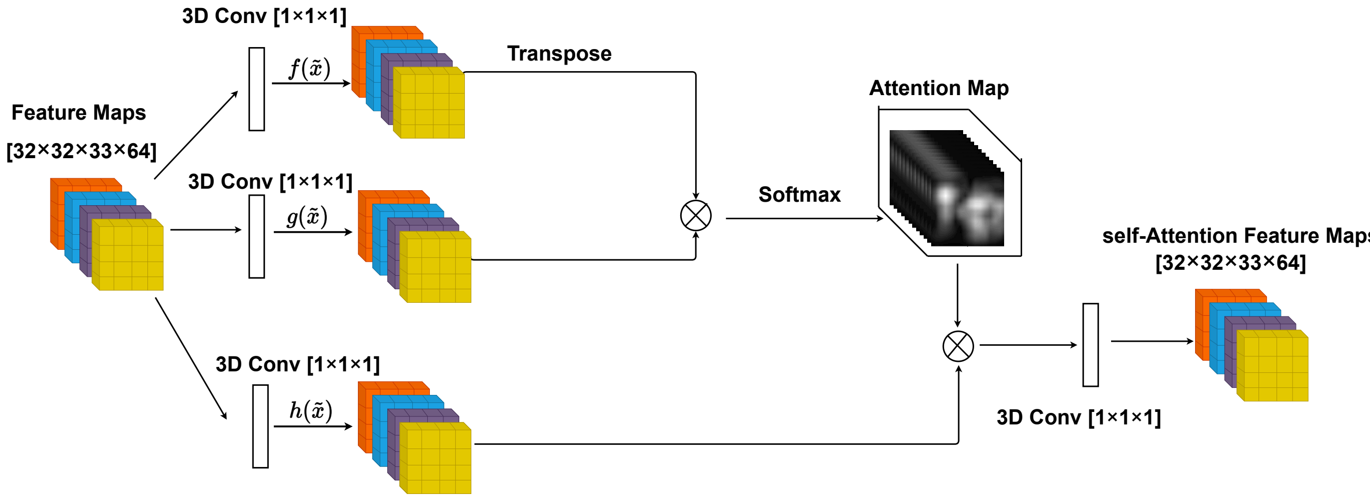


Fig. 5. Network architecture of our 3D self-attention layer.

3.2. Dynamic Latent Code Selection

In a typical encoder-decoder architecture, the latent space is fixed $l \in \mathbb{R}^n$, where n is the latent dimension. However, for a given shape, not all the latent dimensions are relevant. Responses from such irrelevant dimensions may have negative impact on the reconstruction quality. To address this, as shown in Figure 4, we introduce a selection process such that only selected latent dimensions are retained, with the remaining components in the latent code set to zero. Specifically, the model first learns to predict the weight for each latent dimension, collectively as a latent weight vector $w \in (0, 1)^n$, denoted as $w = \omega(l)$, where $\omega(\cdot)$ is the weight prediction network, and in practice, it is achieved by passing the latent code l through two fully connected (FC) layers each with n units, and ReLU and sigmoid activation functions are used after the two FC layers respectively. This makes the output w to be in the range $(0, 1)$ for each dimension. Then, we use the predicted weights to determine which latent components should be retained, namely, only those with the weights in the top K weights (where K is a hyper-parameter) are kept. Then the i-th component of the output latent code $\tilde{l}$ satisfies:

$$\tilde{l}_i = l_i \cdot \mathbf{1}(w_i \in W_K), \tag{1}$$

where $\mathbf{1}(\cdot)$ is 1 if the predicate is true, and 0 otherwise. W_K is the set containing the top K weights. This approach achieves two effects. On the one hand, by suppressing low-weight (i.e., recognized as unimportant) components, this avoids their negative impacts. On the other hand, the network strives to reconstruct high-quality complete 3D shapes with at most K latent components, essentially serving as a strong sparse regularization, that helps improve the robustness of the network. Note that while selecting K latent components, we maintain their positions in the latent space, rather than removing zero components. This makes the follow-up FC layers more efficient to learn.

3.3. 3D Self-Attention Layer

A limitation of convolutions is that they can only capture local features, and so convolution tends to distort the shapes when attempting to recover non-local features. To overcome this issue, we introduce a self-attention layer in this task. Self-attention has been shown to be effective in the GAN framework for improving *image* generation [54] and due to the nature of the input in our problem (i.e., single-view depth images), significant information is missing. The self-attention mechanism focuses attention on the most important global features, which helps to reduce distortion in the reconstruction. The paper [54] incorporates a self-attention mechanism for both the generator and the discriminator. However, in our 3D reconstruction setting, self-attention can only be applied to feature maps with relatively low resolution (e.g. around 16^3) since the relationships between every pair of locations need to be considered. This is still useful to help recover more global structures. As we will later show, incorporating such a 3D self-attention (3DSA) layer in the generator is unable to capture meaningful non-local relationships and actually leads to worse performance. We therefore only consider incorporating the 3DSA layer in the discriminator network.

The network architecture for the 3DSA layer is illustrated in Figure 5. The input feature map $\tilde{x}$ has a spatial resolution of $32 \times 32 \times 33$ with 64 channels. It passes through two different $1 \times 1 \times 1$ convolutions to obtain $f(\tilde{x})$ and $g(\tilde{x})$. The contribution $\beta_{j,i}$ of the jth location from the feature map at the ith location is calculated as follows

$$\beta_{j,i} = \frac{\exp\left(f(\tilde{x}_i)^T g(\tilde{x}_j)\right)}{\sum_{i=1}^{\tilde{N}} \exp\left(f(\tilde{x}_i)^T g(\tilde{x}_j)\right)} \tag{2}$$

where $\tilde{N}$ is the number of spatial locations. β is then used as weights to combine feature maps $h(\tilde{x})$, obtained through $1 \times 1 \times 1$ convolution, and then the final output of the 3DSA layer is obtained through another $1 \times 1 \times 1$ convolution $v(\cdot)$.

3.4. Loss Function

The model has three loss functions: reconstruction loss, GAN loss and classifier loss, and the GAN has generator and discriminator losses.

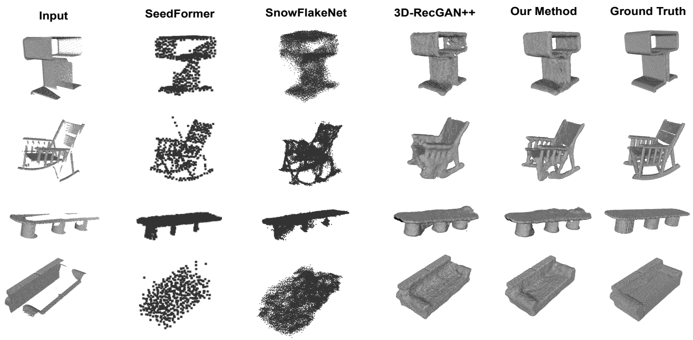


Fig. 6. Visual comparison of completed single categories on same view samples.

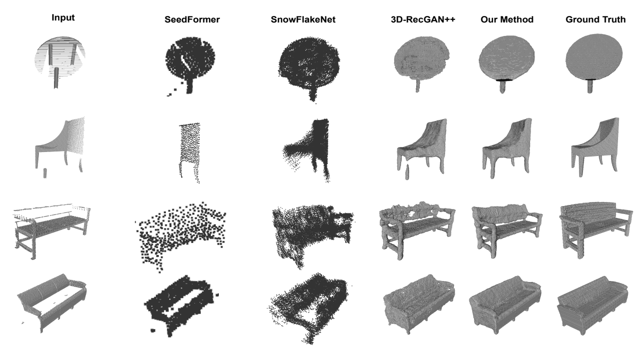


Fig. 7. Visual comparison of completed Multi categories on same view samples.

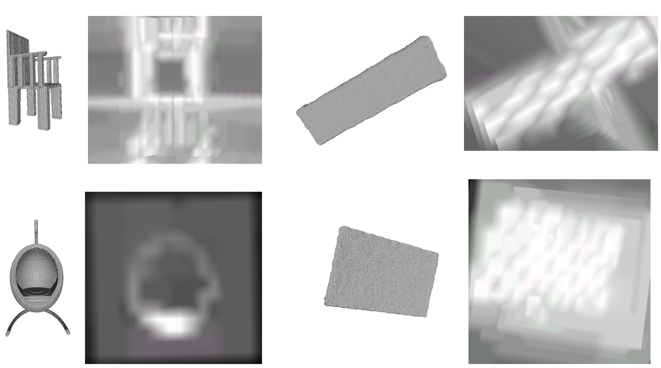


Fig. 8. Visualization of self-attention maps where the layer attends to features relating to shapes.

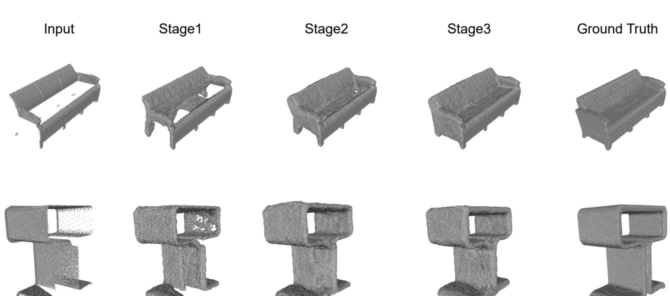


Fig. 9. Visualization of cascade stages.

Reconstruction Loss. As in [12], modified binary cross entropy (BCE) [55] is used rather than mean square error (MSE), to avoid a non-convex problem:

$$L_{BCE} = -\frac{1}{N}\sum_{i=1}^{N}[-\bar{y}_i \log(y_i) - \alpha(1-\bar{y}_i)\log(1-y_i)]. \quad (3)$$

When using the standard BCE equation the empty space will dominate the generated volume, which encourages the model to classify occupied grid cells as empty voxels, resulting in estimation errors. Thus, α is introduced in Eq. 3 to represent the cost weight of the terms. $\bar{y}_i$ represents the ith voxel in the ground truth and y_i represents the ith voxel in the reconstructed shape where N is the number of voxels in the space.

GAN Loss. L_G (Eq. 4) is the loss for generating fake shapes, while L_D (Eq. 5) is the discriminator loss used by WGAN-GP [56]. y represents the generated shape from input x (2.5D) and $\bar{y}$ is the ground truth for the complete shape. In order to tackle the vanishing gradient problem, WGAN-GP adds a penalty term (with weight λ) to encourage the gradient norm of the discriminator to be close to 1; $\hat{y}$ is a perturbed version of y.

$$L_G = -E[D(y|x)]. \quad (4)$$

$$L_D = E[D(y|x)] - E[D(\bar{y}|x)] + \lambda E[(\|\nabla_{\hat{y}} D(\hat{y}|x)\|_2 - 1)^2]. \quad (5)$$

Classifier Loss. We use log loss. M represents the number of classes. y is a binary indicator for whether class label c is the correct classification for observation o. p is the predicted probability that observation o is of class c.

$$L_{Classifier} = -\sum_{c=1}^{M}[y_{o,c}\log(p_{o,c})]. \quad (6)$$

Combined generator loss. As the generator has two objectives, a weight is applied to balance both losses during optimization as follows:

$$L_{weighted} = \gamma L_{BCE} + (1-\gamma)L_G + \zeta L_{Classifier}. \quad (7)$$

$L_{weighted}$ is minimized when training the generator, and L_D is minimized when training the discriminator.

4. Experiments

4.1. Training Details

The model was trained for 20 epochs with a batch size of 3. We set the learning rate for both the generator and discriminator to 0.0001. For the optimizer, Adam [57] was used with $\beta_1 = 0.9, \beta_2 = 0.999$, and $\epsilon = 10^{-8}$. We set the WGAN-GP gradient penalty to $\lambda = 10$ and $\alpha = 0.35$ for modified binary cross entropy. Finally, we set the weighted loss parameter $\gamma = 0.8$ and $\zeta = 0.01$. The networks were trained on Nvidia GTX 1080ti, and it took on average 4.5 days to train a model.

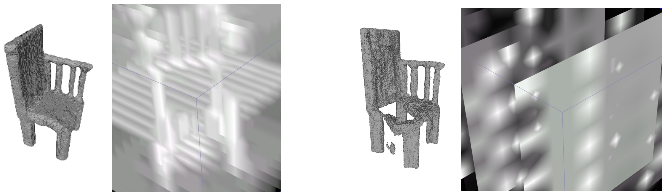


Fig. 10. Comparison of applying self-attention to the discriminator (left) and generator (right). A more meaningful self-attention map and shape are obtained when incorporating self-attention in the discriminator.

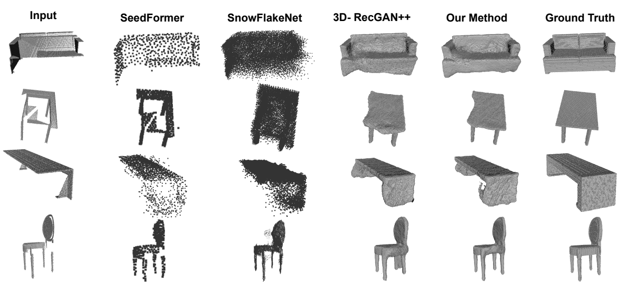


Fig. 11. Qualitative results of single category reconstruction on testing datasets with cross viewing angles.

4.2. Dataset

In our experiments, we used datasets provided by [12], for which the authors had generated depth views from ShapeNet datasets. In total, 272 CAD models were used. The breakdown was: training used 220 models, testing 40 models, and validation 12 models. All models in the dataset were voxelized to a 256^3 grid. Datasets were split into two sets: same view (all input depth images captured in one direction, 125 different views) and cross view (depth images from multiple views, 216 different views). For training, only the same view depth images were generated, while for testing and validation both same view and cross view sets generated. In total, there are 26000 training samples. The same view test consists of 4500 samples and 8000 cross view test samples. The validation set contains 1500 samples for same view and 2500 for cross view. Four categories have training sets (chair, table, bench, couch) while the rest are used for testing as unseen objects (plane, car, monitor, faucet, guitar, firearm).

4.3. Evaluation

To compare our work with other state-of-the-art methods, we evaluated our model using intersection over union (IoU). IoU was applied on a per voxel basis to the ground truth and recovered shape. The second evaluation metric was mean value cross-entropy (CE).

As discussed in [12], Chamfer distance and earth mover distance are infeasible for high-resolution voxel sets due to the high computational cost.

Comparison to prior work. To evaluate the performance of the model in reconstructing a 3D shape from a single-depth view, we compared it to three recent works on reconstructing a 3D shape from a single-depth image. (1) The 3D-EPN model presented by [58] completed the shape by leveraging semantic features; the resolution of the reconstructed shape was 32^3. The

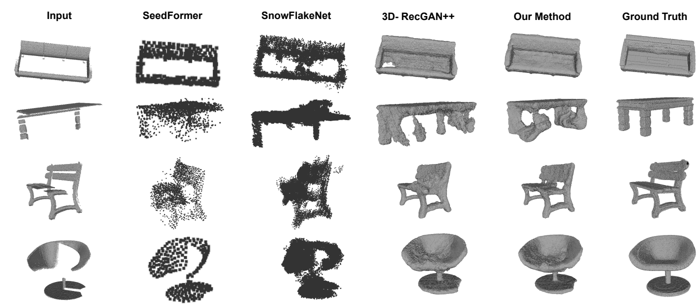


Fig. 12. Qualitative results of Multi-categories reconstruction on testing datasets with cross viewing angles.

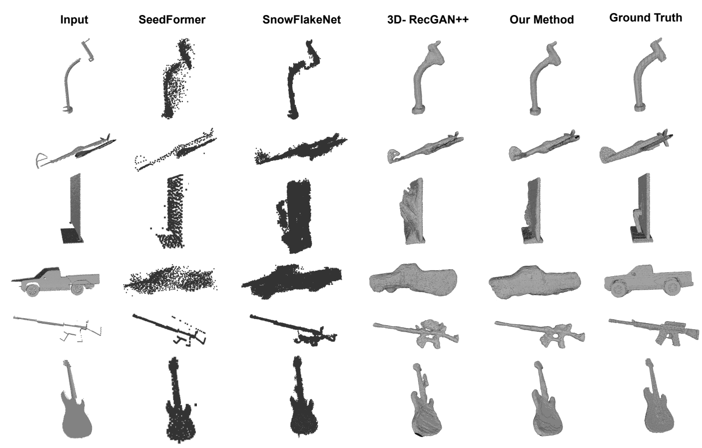


Fig. 13. Qualitative results of Multi-categories reconstruction on testing datasets with same viewing angles for unseen objects.

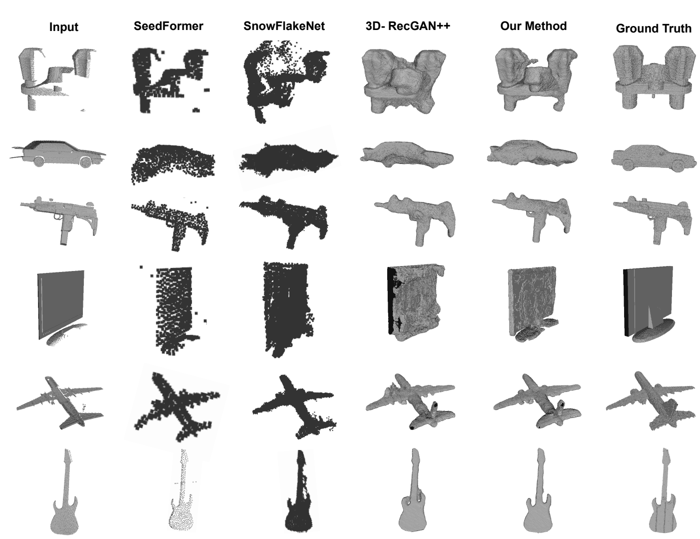


Fig. 14. Qualitative results of Multi-categories reconstruction on testing datasets with cross viewing angles for unseen objects.

Table 1. IoU and Cross entropy evaluation metric for Single categories, same view, comparing 3D-EPN [58], [59], SnowFlakeNet [60], SeedFormer [61], 3D-RecGAN++ [12] (denoted as Yang in the table) and our 3DCascade-GAN.

IoU	Bench	Chair	Couch	Table
3D-EPN	0.423	0.488	0.631	0.508
Varley	0.227	0.317	0.544	0.233
SnowFlakeNet	0.562	0.631	0.745	0.659
SeedFormer	0.553	0.618	0.740	0.656
Yang	0.580	0.647	0.753	0.679
Ours	**0.641**	**0.701**	**0.809**	**0.698**
CE	Bench	Chair	Couch	Table
3D-EPN	0.087	0.105	0.144	0.101
Varley	0.111	0.157	0.195	0.191
SnowFlakeNet	0.037	0.063	0.068	0.043
SeedFormer	0.038	0.065	0.069	0.044
Yang	0.034	0.060	0.066	0.040
Ours	**0.030**	**0.053**	**0.063**	**0.038**

Table 2. IoU and Cross entropy evaluation metric for Multi categories, same view

IoU	Bench	Chair	Couch	Table
3D-EPN	0.428	0.484	0.634	0.506
Varley [59]	0.234	0.317	0.543	0.236
SnowFlakeNet	0.548	0.624	0.736	0.633
SeedFormer	0.542	0.613	0.727	0.628
3D-RecGAN++	0.581	0.640	0.745	0.667
3DCascade-GAN	**0.624**	**0.669**	**0.773**	**0.682**
CE	Bench	Chair	Couch	Table
3D-EPN	0.087	0.107	0.138	0.102
Varley [59]	0.103	0.132	0.197	0.170
SnowFlakeNet	0.035	0.053	0.064	0.043
SeedFormer	0.036	0.054	0.066	0.045
3D-RecGAN++	0.030	0.051	0.063	0.039
3DCascade-GAN	**0.028**	**0.049**	**0.060**	**0.037**

Table 3. IoU and Cross entropy evaluation metric for Single categories, cross view

IoU	Bench	Chair	Couch	Table
3D-EPN	0.408	0.446	0.572	0.482
Varley [59]	0.185	0.278	0.475	0.187
SnowFlakeNet	0.508	0.578	0.628	0.603
SeedFormer	0.503	0.563	0.627	0.601
3D-RecGAN++	0.531	0.594	0.646	0.618
3DCascade-GAN	**0.585**	**0.628**	**0.680**	**0.647**
CE	Bench	Chair	Couch	Table
3D-EPN	0.086	0.112	0.163	0.103
Varley [59]	0.108	0.171	0.210	0.186
SnowFlakeNet	0.045	0.079	0.118	0.055
SeedFormer	0.046	0.080	0.120	0.056
3D-RecGAN++	0.041	0.074	0.111	0.053
3DCascade-GAN	**0.038**	**0.070**	**0.109**	**0.051**

model then used a retrieval approach to collect similar shapes for shape reconstruction. (2) Varley et al. [59] addressed the issue of robot grasp planning; the model reconstructed a 3D shape from 2.5D images that were captured using a depth camera. The model resolution was 40^3 voxels. (3) SnowflakesNet [60] processes a point cloud representation, and the model predicts a complete shape from an incomplete point cloud. We process the output by voxelizing the output points to 256^3 resolution for quantitative comparison. (4) SeedFormer [61] also uses a point cloud representation where the input is an incomplete point cloud and the prediction is a complete shape. We process the output by voxelizing the output points to 256^3 resolution for quantitative comparison. (5) 3D RecGAN++ [12] reconstructed a 3D shape from a 2.5D image with a resolution of 64^3 and up sampled to 256^3. For methods based on implicit representations, neither [43] or [62] provided the code for 3D completion, so we trained the model of [63] on our datasets, but it failed to learn the representation.

For the qualitative comparison, we show results of 3D RecGAN++ [12], SnowFlakeNet [60] and SeedFormer [61], as these models are state-of-the-art and have the same recovered shape resolution as our model. Note, in the qualitative results for [60] and [61] we show point cloud representations to avoid the potential distortions caused by discretization.

4.4. Results

Seen shape category experimental results. The model was trained on 4 different datasets (chair, table, bench, and couch). A single category means each one was trained separately with the same settings as mentioned. On the other hand, Multi-categories means the model was trained on all the 4 datasets (chair, table, bench, and couch). The IoU and CE results for single categories, same view are displayed in Table 1. Table 2 shows IoU and CE results for Multi categories same view. Table 3 presents single categories cross view using IoU and CE respectively and Table 4 shows cross view for Multi categories. After training, we find the best threshold between [0.1, 0.9] with a step of 0.05 on a validation dataset using only the IoU criterion. After finding the best threshold to represent the model, we applied it on the test dataset as suggested by [12]. In the quantitative results, both IoU and CE demonstrated that our model outperformed the state-of-the-art model, and qualitatively it can be seen that our method recovered 3D shapes at high resolution with accurate details. For the qualitative results for single categories in same view testing datasets, see Figure 6, where artifacts appear in the results of 3D RecGAN++ such as incorrect structure/geometry and Multi categories also in same view datasets in Figure 7. Figure 12 shows Multi categorises in cross view datasets. Figure 8 visualizes self-attention maps when completing some shapes, which clearly capture global structures. The intermediate results after each of the three stages are shown in Figure 9.

Unseen shape category experimental results. Lastly, we conduct experiments on six more categories where the model is trained on chair, bench, couch, table and then tested on car, faucet, firearm, guitar, monitor, plane for both same view and cross view datasets. The IoU and CE results for cross-view re-

Table 4. IoU evaluation metric for Multi categories, cross view

IoU	Bench	Chair	Couch	Table
3D-EPN	0.415	0.452	0.531	0.477
Varley [59]	0.201	0.283	0.480	0.199
SnowFlakeNet	0.534	0.586	0.631	0.612
SeedFormer	0.532	0.583	0.629	0.609
3D-RecGAN++	0.540	0.594	0.643	0.621
3DCascade-GAN	**0.574**	**0.620**	**0.673**	**0.633**
CE	Bench	Chair	Couch	Table
3D-EPN	0.091	0.115	0.147	0.111
Varley [59]	0.105	0.143	0.207	0.174
SnowFlakeNet	0.039	0.068	0.095	0.050
SeedFormer	0.040	0.069	0.097	0.052
3D-RecGAN++	0.038	0.061	0.091	0.048
3DCascade-GAN	**0.036**	**0.058**	**0.089**	**0.047**

sults are shown in Table 5 and same view results in Table 6. Figure 13 shows visualization for the same view dataset and figure 14 shows cross view visualization. Our method performs consistently better than state-of-the-art methods in all categories, and both same-view and cross-view cases.

4.5. Ablation Studies

In this section, we describe three ablation studies: dynamic latent code, second self-attention layer and classifier. For comparison, we choose the chair datasets for our ablation experiments as these samples show more complex structure compared to bench, table and couch.

Dynamic latent code. We conducted an experiment where the dynamic layer was disabled and a fixed 2000 code size was used; the result was worse compared to the dynamic layer, as shown in Table 9. Also, three different experiments with three different K values: 50, 100 and 150 on a single encoder-decoder conducted. We found that the result was worse when $K = 50$; however, performance with both $K = 100$ and 150 had the same result. We also observe the model behavior when k approaches n ($K = 600$, $K = 900$), and the results show the performance drops gradually. Using the dynamic latent code encoder tends to optimize the latent codes where most values are set to zero, and these codes vary based on input shape. Furthermore, to show effectiveness of dynamic latent code, we trained the model with/without each components, the results shown in Table 7.

Self-attention. We tried using self-attention in both the networks (i.e. the encoder-decoder and discriminator), as shown in Figure 10, and tried using it on different layers to achieve the optimum results. The trials revealed that adding self-attention to the encoder-decoder did not improve the results; in fact, the self-attention maps obtained when adding the self-attention layer to the generator network did not capture global structures well, and lead to poor reconstruction results. On the other hand, adding our self-attention layer to the discriminator effectively increased its capability to differentiate between real and fake 3D shapes, and eventually helped improve the capability of the generator to produce improved reconstruction.

Table 5. IoU and cross entropy evaluation metric for multi-category training and applied to unseen object categories, cross view, comparing 3D-EPN, [59], SnowFlakeNet [60] (denoted Snow), SeedFormer [61] (denoted Seed), 3D-RecGAN++ and our 3DCascade-GAN.

IoU	car	faucet	firearm	guitar	monitor	plane
3D-EPN	0.446	0.439	0.324	0.359	0.448	0.309
Varley	0.489	0.260	0.274	0.255	0.334	0.283
Snow	0.534	0.510	0.409	0.437	0.549	0.384
Seed	0.527	0.507	0.407	0.435	0.546	0.383
Yang	0.553	0.529	0.416	0.449	0.555	0.390
Ours	**0.564**	**0.537**	**0.425**	**0.455**	**0.560**	**0.394**
CE	car	faucet	firearm	guitar	monitor	plane
3D-EPN	0.160	0.086	0.033	0.036	0.127	0.065
Varley	0.171	0.123	0.028	0.030	0.136	0.043
Snow	0.103	0.060	0.018	0.016	0.078	0.033
Seed	0.105	0.061	0.018	0.017	0.079	0.034
Yang	0.100	0.055	0.014	0.015	**0.074**	**0.031**
Ours	**0.098**	**0.054**	**0.013**	**0.013**	**0.074**	**0.031**

Classifier. For the classifier, we compared the full version of the model (including cascade, dynamic latent code, self-attention and classifier) against a model without a classifier. As shown in Table 8, there are slight differences in that the classifier enhances the shapes, and this improvement is consistent.

5. Conclusion

In this paper, we proposed an end-to-end model for 3D reconstruction from a single depth image. We introduced a 3D self-attention layer to attend to the non-local features, helping to connect the recovered views with the known view of the 3D shape. We also demonstrate introducing a dynamic latent code as an aid to optimizing the encoder, reducing the effective size of the latent space which enhanced the results. These additions helped stabilize adversarial learning which leads to better estimation as demonstrated on different shape categories, both qualitatively and quantitatively. We further added multi-stage networks to sequentially refine 3D shapes. Furthermore, incorporating the classifier network showed improvement to the reconstructed shapes. Our method produces shapes with improved structure/geometry, outperforming state-of-the-art methods.

Acknowledgement

The study benefited from Advanced Research Computing at Cardiff (ARCCA) computing facilities.

References

[1] Janoch, A, Karayev, S, Jia, Y, Barron, JT, Fritz, M, Saenko, K, et al. A category-level 3-D object dataset: Putting the Kinect to work. In: ICCV Workshop. 2011,.

[2] Li, X, Yi, W, Chi, HL, Wang, X, Chan, AP. A critical review of virtual and augmented reality (VR/AR) applications in construction safety. Automation in Construction 2018;86:150–162.

Table 6. IoU and cross entropy evaluation metric for multi-category training and applied to unseen object categories, same view, comparing 3D-EPN, [59], SnowFlakeNet [60] (denoted Snow), SeedFormer [61] (denoted Seed), 3D-RecGAN++ and our 3DCascade-GAN.

IoU	car	faucet	firearm	guitar	monitor	plane
3D-EPN	0.450	0.442	0.339	0.351	0.444	0.314
Varley	0.484	0.260	0.280	0.255	0.341	0.295
Snow	0.548	0.526	0.412	0.438	0.554	0.371
Seed	0.545	0.524	0.409	0.435	0.553	0.367
Yang	0.555	0.536	0.426	0.442	0.562	0.394
Ours	**0.559**	**0.541**	**0.430**	**0.455**	**0.569**	**0.395**
CE	car	faucet	firearm	guitar	monitor	plane
3D-EPN	0.170	0.088	0.036	0.036	0.123	0.066
Varley	0.173	0.122	0.029	0.030	0.130	0.042
Snow	0.104	0.056	0.018	0.017	0.069	0.033
Seed	0.105	0.058	0.019	0.018	0.068	0.034
Yang	0.102	**0.053**	**0.016**	0.014	0.067	**0.031**
Ours	**0.101**	**0.053**	**0.016**	**0.013**	**0.065**	**0.031**

Table 7. Ablation study on Dynamic latent code and self-attention

	Chair-IoU	Chair-CE
3D-Cascade-GAN	0.701	0.053
without Dynamic layer	0.663	0.054
without self-attention	0.692	0.053
without self-attention & dynamic	0.654	0.054

Table 8. Ablation study on Classifier

	Bench	Chair	Couch	Table
with classifier	0.624	0.669	0.773	0.682
without classifier	0.622	0.667	0.771	0.681

Table 9. Ablation study on Dynamic latent code, we compare fixed latent code with different variation of dynamic code.

	Chair-IoU	Chair-CE
Fixed latent code: 2000	0.649	0.059
$n = 1000, K = 50$	0.645	0.061
$n = 1000, K = 100$	0.701	0.053
$n = 1000, K = 150$	0.700	0.053
$n = 1000, K = 600$	0.698	0.057
$n = 1000, K = 900$	0.656	0.059

[3] Theodoropoulos, A, Antoniou, A. VR games in cultural heritage: A systematic review of the emerging fields of virtual reality and culture games. Applied Sciences 2022;12(17):8476.

[4] Li, Q, Cao, R, Zhu, J, Fu, H, Zhou, B, Fang, X, et al. Learn then match: A fast coarse-to-fine depth image-based indoor localization framework for dark environments via deep learning and keypoint-based geometry alignment. ISPRS Journal of Photogrammetry and Remote Sensing 2023;195:169–177.

[5] Macario Barros, A, Michel, M, Moline, Y, Corre, G, Carrel, F. A comprehensive survey of visual SLAM algorithms. Robotics 2022;11(1):24.

[6] Liu, Y, Xu, H, Liu, D, Wang, L. A digital twin-based sim-to-real transfer for deep reinforcement learning-enabled industrial robot grasping. Robotics and Computer-Integrated Manufacturing 2022;78:102365.

[7] Cheng, J, Zhang, L, Chen, Q, Hu, X, Cai, J. A review of visual SLAM methods for autonomous driving vehicles. Engineering Applications of Artificial Intelligence 2022;114:104992.

[8] Wan, L, Jiang, J, Zhang, H. Incomplete 3D shape retrieval via sparse dictionary learning. In: Pacific Graphics Short Papers. 2015, p. 25–30.

[9] Cheung, G, Kanade, T, Bouguet, J, Holler, M. A real time system for robust 3D voxel reconstruction of human motions. In: IEEE CVPR; vol. 2. 2000, p. 714–720.

[10] Nießner, M, Zollhöfer, M, Izadi, S, Stamminger, M. Real-time 3D reconstruction at scale using voxel hashing. ACM Trans Graph 2013;32(6):169.

[11] Hane, C, Tulsiani, S, Malik, J. Hierarchical surface prediction for 3D object reconstruction. In: Intl. Conf. 3D Vision (3DV). 2017,.

[12] Yang, B, Rosa, S, Markham, A, Trigoni, N, Wen, H. Dense 3D object reconstruction from a single depth view. IEEE Trans Patt Anal Mach Intell 2019;41(12):2820–2834.

[13] Wu, J, Wang, Y, Xue, T, Sun, X, Freeman, B, Tenenbaum, J. MarrNet: 3D shape reconstruction via 2.5D sketches. In: Proceedings of the neural information processing systems (NIPS). 2017, p. 540–550.

[14] Zhang, X, Zhang, Z, Zhang, C, Tenenbaum, J, Freeman, B, Wu., J. Learning to reconstruct shapes from unseen classes. In: NIPS. 2018, p. 2257–2268.

[15] Wu, J, Zhang, C, Zhang, X, Zhang, Z, Freeman, WT, Tenenbaum, JB. Learning shape priors for single-view 3D completion and reconstruction. In: ECCV. 2018, p. 673–691.

[16] Yan, X, Yang, J, Yumer, E, Guo, Y, Lee, H. Perspective transformer nets: Learning single-view 3D object reconstruction without 3D supervision. In: NIPS. 2016, p. 1696–1704.

[17] Yu, J, Yin, W, Hu, Z, Liu, Y. 3D reconstruction for multi-view objects. Computers and Electrical Engineering 2023;106:108567.

[18] Xie, H, Yao, H, Zhang, S, Zhou, S, Sun, W. Pix2Vox++: multi-scale context-aware 3D object reconstruction from single and multiple images. International Journal of Computer Vision 2020;128(12):2919–2935.

[19] Huang, Z, Jampani, V, Thai, A, Li, Y, Stojanov, S, Rehg, JM. ShapeClipper: Scalable 3D shape learning from single-view images via geometric and CLIP-based consistency. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023, p. 12912–12922.

[20] Xian, Y, Chibane, J, Bhatnagar, BL, Schiele, B, Akata, Z, Pons-Moll, G. Any-shot GIN: Generalizing implicit networks for reconstructing novel classes. In: International Conference on 3D Vision (3DV). IEEE; 2022, p. 526–535.

[21] Hui, KH, Li, R, Hu, J, Fu, CW. Neural template: Topology-aware reconstruction and disentangled generation of 3D meshes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022, p. 18572–18582.

[22] Hafiz, AM, Bhat, RUA, Parah, SA, Hassaballah, M. SE-MD: a single-encoder multiple-decoder deep network for point cloud reconstruction from 2D images. Pattern Analysis and Applications 2023;:1–12.

[23] Kurenkov, A, Ji, J, Garg, A, Mehta, V, Gwak, J, Choy, C, et al. DeformNet: Free-form deformation network for 3D shape reconstruction from a single image. In: IEEE Winter Conference on Applications of Computer Vision (WACV). 2018,.

[24] Kanazawa, A, Tulsiani, S, Efros, A, Malik, J. Learning category-specific mesh reconstruction from image collections. In: ECCV. 2018, p. 371–386.

[25] Wang, N, Zhang, Y, Li, Z, Fu, Y, Liu, W, Jiang, Y. Pixel2mesh: Generating 3D mesh models from single RGB images. In: ECCV. 2018, p. 52–67.

[26] Wen, C, Zhang, Y, Cao, C, Li, Z, Xue, X, Fu, Y. Pixel2Mesh++: 3D mesh generation and refinement from multi-view images. IEEE Transactions on Pattern Analysis and Machine Intelligence 2022;45(2):2166–2180.

[27] Richter, S, Roth, S. Matryoshka networks: Predicting 3D geometry via nested shape layers. In: IEEE CVPR. 2018, p. 1936–1944.

[28] Peng, K, Islam, R, Quarles, J, Desai, K. TMVNet: Using transformers for multi-view voxel-based 3D reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022, p. 222–230.

[29] Lin, C, Kong, C, Lucey, S. Learning efficient point cloud generation for dense 3D object reconstruction. In: AAAI. 2018,.

[30] Gao, J, Kong, D, Wang, S, Li, J, Yin, B. DASI: Learning domain adaptive shape impression for 3D object reconstruction. IEEE Transactions on Multimedia 2022;.

[31] Yang, G, Cui, Y, Belongie, S, Hariharan, B. Learning single-view 3D reconstruction with limited pose supervision. In: ECCV. 2018, p. 86–101.

[32] Robert, T, Thome, N, Cord, M. HybridNet: Classification and reconstruction cooperation for semi-supervised learning. In: ECCV. 2018, p. 153–169.

[33] Jiang, L, Shi, S, Qi, X, Jia, J. GAL: Geometric adversarial loss for single-view 3D-object reconstruction. In: ECCV. 2018, p. 802–816.

[34] Gwak, J, Choy, C, Chandraker, M, Garg, A, Savarese, S. Weakly supervised 3D reconstruction with adversarial constraint. In: IEEE Intl. Conf. 3D Vision (3DV). 2017, p. 263–272.

[35] Tatarchenko, M, Dosovitskiy, A, Brox, T. Octree generating networks: Efficient convolutional architectures for high-resolution 3D outputs. In: ICCV. 2017,.

[36] Wang, PS, Liu, Y, Tong, X. Deep octree-based CNNs with output-guided skip connections for 3D shape and scene completion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. 2020, p. 266–267.

[37] Hu, T, Han, Z, Zwicker, M. 3D shape completion with multi-view consistent inference. In: Proceedings of the AAAI Conference on Artificial Intelligence; vol. 34. 2020, p. 10997–11004.

[38] Wang, X, Ang Jr, MH, Lee, GH. Cascaded refinement network for point cloud completion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020, p. 790–799.

[39] Huang, Z, Yu, Y, Xu, J, Ni, F, Le, X. PF-Net: Point fractal network for 3D point cloud completion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020, p. 7662–7670.

[40] Wen, X, Li, T, Han, Z, Liu, YS. Point cloud completion by skip-attention network with hierarchical folding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020, p. 1939–1948.

[41] Pan, L, Chen, X, Cai, Z, Zhang, J, Zhao, H, Yi, S, et al. Variational relational point completion network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021, p. 8524–8533.

[42] Wang, Y, Tan, DJ, Navab, N, Tombari, F. ForkNet: Multi-branch volumetric semantic completion from a single depth image. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. 2019, p. 8608–8617.

[43] Park, JJ, Florence, P, Straub, J, Newcombe, R, Lovegrove, S. DeepSDF: Learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019, p. 165–174.

[44] Wu, T, Pan, L, Zhang, J, Wang, T, Liu, Z, Lin, D. Density-aware Chamfer distance as a comprehensive metric for point cloud completion. In: Advances in Neural Information Processing Systems (NeurIPS). 2021,.

[45] Alliegro, A, Valsesia, D, Fracastoro, G, Magli, E, Tommasi, T. Denoise and contrast for category agnostic shape completion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021, p. 4629–4638.

[46] Li, S, Gao, P, Tan, X, Wei, M. ProxyFormer: Proxy alignment assisted point cloud completion with missing part sensitive transformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023, p. 9466–9475.

[47] Chen, Z, Long, F, Qiu, Z, Yao, T, Zhou, W, Luo, J, et al. AnchorFormer: Point cloud completion from discriminative nodes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023, p. 13581–13590.

[48] Wang, Y, Tan, DJ, Navab, N, Tombari, F. SoftPool++: An encoder–decoder network for point cloud completion. International Journal of Computer Vision 2022;130(5):1145–1164.

[49] Zhang, Z, Yu, Y, Da, F. Partial-to-partial point generation network for point cloud completion. IEEE Robotics and Automation Letters 2022;7(4):11990–11997.

[50] Yang, Y, Feng, C, Shen, Y, Tian, D. FoldingNet: Point cloud auto-encoder via deep grid deformation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2018, p. 206–215.

[51] Wei, M, Zhu, M, Zhang, Y, Sun, J, Wang, J. Cyclic global guiding network for point cloud completion. Remote Sensing 2022;14(14):3316.

[52] Xiao, Y, Li, Y, Yu, Q, Liu, S, Gang, J. DF-Net: Dynamic and folding network for 3D point cloud completion. IEEE Access 2022;10:97835–97842.

[53] Zhang, J, Chen, X, Cai, Z, Pan, L, Zhao, H, Yi, S, et al. Unsupervised 3D shape completion through GAN inversion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021, p. 1768–1777.

[54] Zhang, H, I., G, Metaxas, D, Odena, A. Self-attention generative adversarial networks. arXiv:180508318 2018;.

[55] Brock, A, Lim, T, Ritchie, JM, Weston, N. Generative and discriminative voxel modeling with convolutional neural networks. arXiv:160804236 2016;.

[56] Gulrajani, I, Ahmed, F, Arjovsky, M, Dumoulin, V, Courville, A. Improved training of Wasserstein GANs. In: NIPS. 2017, p. 5767–5777.

[57] Kingma, D, Ba, J. Adam: A method for stochastic optimization. arXiv:14126980 2014;.

[58] Dai, A, Ruizhongtai Qi, C, Nießner, M. Shape completion using 3D-encoder-predictor CNNs and shape synthesis. In: CVPR. 2017, p. 5868–5877.

[59] Varley, J, DeChant, C, Richardson, A, Ruales, J, Allen, P. Shape completion enabled robotic grasping. In: IEEE/RSJ IROS. 2017, p. 2442–2447.

[60] Xiang, P, Wen, X, Liu, YS, Cao, YP, Wan, P, Zheng, W, et al. Snowflake point deconvolution for point cloud completion and generation with skip-transformer. IEEE Transactions on Pattern Analysis and Machine Intelligence 2023;45(5):6320–6338. doi:10.1109/TPAMI.2022.3217161.

[61] Zhou, H, Cao, Y, Chu, W, Zhu, J, Lu, T, Tai, Y, et al. SeedFormer: Patch seeds based point cloud completion with upsample transformer. arXiv preprint arXiv:220710315 2022;.

[62] Genova, K, Cole, F, Sud, A, Sarna, A, Funkhouser, T. Local deep implicit functions for 3D shape. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020, p. 4857–4866.

[63] Mescheder, L, Oechsle, M, Niemeyer, M, Nowozin, S, Geiger, A. Occupancy networks: Learning 3D reconstruction in function space. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019, p. 4460–4470.