UNIST: Unpaired Neural Implicit Shape Translation Network

Qimin Chen[1] Johannes Merz[1] Aditya Sanghi[2] Hooman Shayani[2]
Ali Mahdavi-Amiri[1] Hao Zhang[1]
[1]Simon Fraser University [2]Autodesk AI Lab

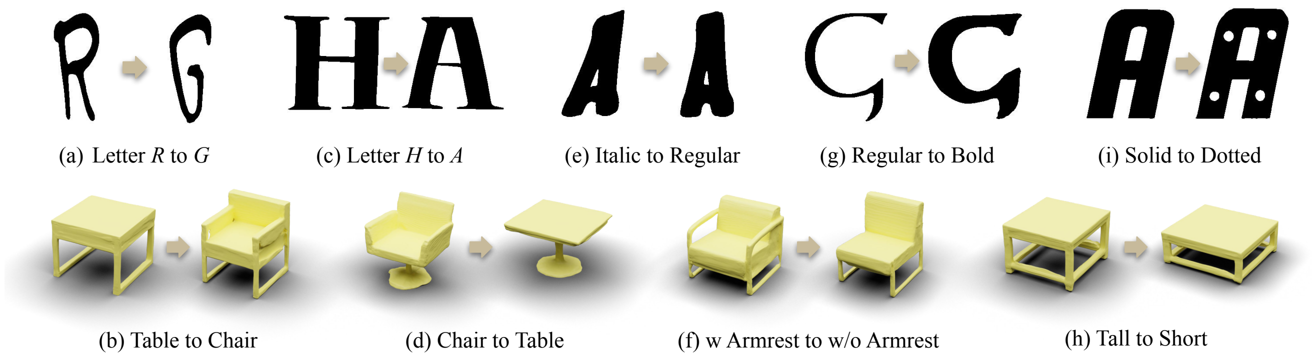


Figure 1. We present UNIST, a model built on *neural implicit* representation that is able to learn both *style-preserving content alteration* (a-d) and *content-preserving style transfer* (e-i) between two *unpaired* domains of shapes, using the *same* network architecture.

Abstract

We introduce UNIST, the first deep neural implicit model for general-purpose, unpaired shape-to-shape translation, in both 2D and 3D domains. Our model is built on autoencoding implicit fields, rather than point clouds which represents the state of the art. Furthermore, our translation network is trained to perform the task over a latent grid *representation which combines the merits of both latent-space processing and* position awareness, *to not only enable drastic shape transforms but also well preserve spatial features and fine local details for natural shape translations. With the same network architecture and only dictated by the input domain pairs, our model can learn both style-preserving content alteration and content-preserving style transfer. We demonstrate the generality and quality of the translation results, and compare them to well-known baselines. Code is available at* https://qiminchen.github.io/unist/.

1. Introduction

Unpaired image-to-image translation has become one of the most extensively studied problems in computer vision since the advent of CycleGAN [29], DualGAN [26], and UNIT [16] in 2017. Somewhat surprisingly, there have been much fewer works on the same problem for shapes, i.e., *unpaired shape-to-shape translation*. To date, most image translation networks have been designed for style transfers that are localized, without large structural alterations. For shape translations, however, one may naturally expect more of the latter, e.g., to change the shape of a letter 'R' to that of a 'G', or a table to a chair; see Figures 1(a-d).

Recently, Yin et al. [27] proposed LOGAN, an unpaired shape translation network that can be trained to execute both style and content, i.e., shape- and structure-level, transforms. However, their network was designed to operate on low-resolution point clouds (up to 2,048 points), which can severely limit the quality of the reconstructed and translated shapes, especially in the 3D case. In addition, the translation network is trained to operate on "holistic" latent codes which encode global information that is multi-scale but *not position-aware*. A consequence of such a lack of positional information in the encoding is losing control of spatial features, as well as local details, during shape translation. For example, when the translation is supposed to only italicize a letter shape, which is a pose change, the local details of a source shape (e.g., thickness/sharpness of certain tips of the shape) may be unexpectedly altered as well, as shown in the second row of Figure 4 on the letter A translation.

In this paper, we present a method for unpaired shape-to-shape translation that is built on autoencoding *neural implicit fields* [3, 18, 20], rather than point clouds [27]. In recent years, advantages of learning continuous implicit functions over discrete representations such as voxels, mesh

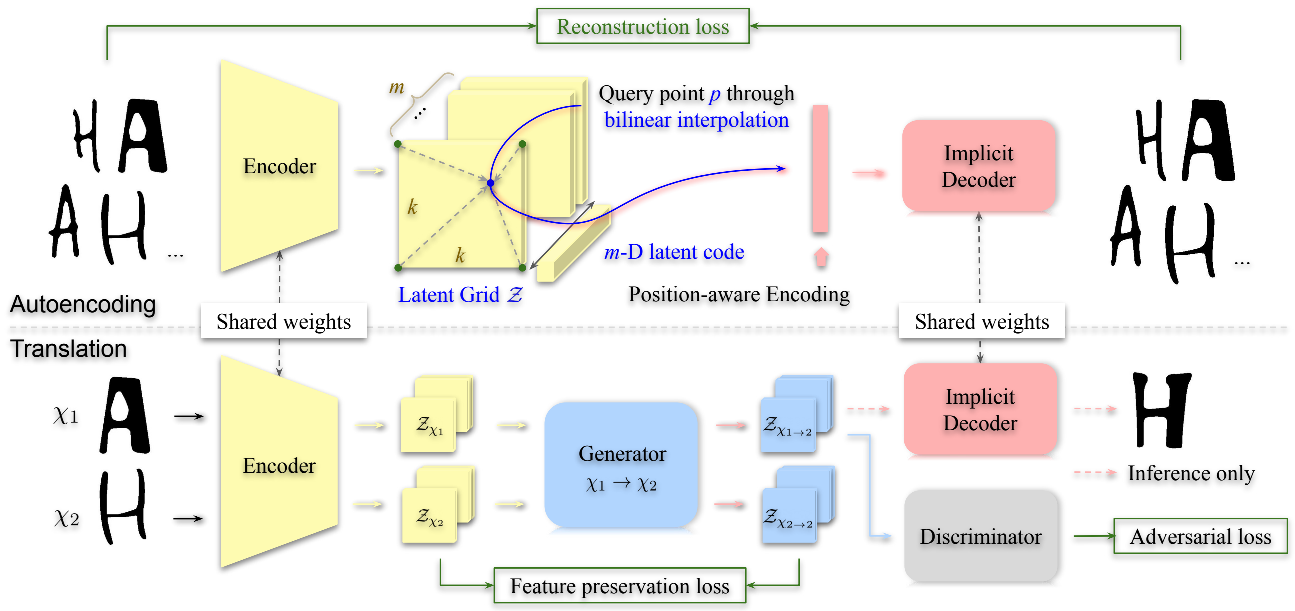


Figure 2. Overview of our framework for unpaired neural implicit shape-to-shape translation, which consists of two separately trained networks. The autoencoding network (top) learns to encode and decode binary voxel occupancies for shapes from both the source and target domains, where the encoder maps an input shape to a *latent grid* representation $\mathcal{Z}$. In the 2D case, $\mathcal{Z} \in \mathbb{R}^{k\times k\times m}$, where the $k \times k$ grid is obtained via spatial convolution over the $n \times n$ input image, and m is the length of the latent code. The latent feature at any query point p is obtained via *bilinear interpolation* over the latent codes stored in $\mathcal{Z}$. In the 3D case, the grid is three-dimensional and obtained via volumetric convolution and trilinear interpolation is performed to extract latent features for the decoder. The translation network (bottom) employs the pre-trained autoencoder network above to transform the translation problem into a latent space. In that space, a generator learns two tasks: 1) translating source-domain codes ($\mathcal{Z}_{\chi_1}$) into target-domain codes ($\mathcal{Z}_{\chi_{1\to 2}}$); 2) preserving target-domain codes, from $\mathcal{Z}_{\chi_2}$ to $\mathcal{Z}_{\chi_{2\to 2}}$. $\mathcal{Z}_{\chi_{1\to 2}}$ is passed to the pre-trained implicit decoder to obtain the final target shape resulting from the generator network.

patches, and point clouds have been demonstrated predominantly for reconstructive tasks including neural rendering, shape completion, and single-view 3D reconstruction. Our work shall show that the same advantages of neural implicit models can be carried over to domain translation.

Furthermore, our translation network is trained to perform the task over a *latent grid* representation whose grid structure is *spatially correlated* with that of the input shapes, via convolution, while its remaining dimension encodes the latent features. Hence, our approach combines the merits of both latent-space processing and *position awareness*, with the former facilitating more drastic shape translations [16,27] and the latter resulting in better preservation of spatial features and details [4, 5, 22] during the translation.

Our model, coined UNIST for unpaired neural implicit shape translation, consists of two separately trained networks, as illustrated in Figure 2. Given two *unpaired* domains of shapes, e.g., chairs and tables or various letters in different fonts (see Figure 1 for several examples of domain pairs), the autoencoding network learns to encode and decode shapes (in the form of binary voxel occupancies) from *both* domains, using latent grids. The network training is self-supervised with the typical reconstruction loss.

The translation network is based on the LOGAN [27] architecture which consists of a latent generator that is trained to perform two tasks: one is to translate the code of a source shape to that of a target shape under the adversarial setting, and the other is to turn the code of a target shape to itself based on a feature preservation loss. Differently from the original LOGAN, inputs to the UNIST generator are no longer the holistic and overcomplete latent codes; they are replaced by the latent grid features produced by the encoder of the pre-trained autoencoder network (top in Figure 2). The translation network is trained with the same set of losses as LOGAN, while the outputs from the generator would go through the pre-trained decoder (top in Figure 2) to produce the final shapes in the target domain.

Our work represents the first deep implicit model for general-purpose, unpaired shape-to-shape translation. With the same network architecture and only dictated by the input domain pairs, our model can learn both style-preserving content alteration and content-preserving style transfer, as shown in Figure 1. We demonstrate the generality and quality of the translation results, and compare them to LOGAN [27] and other baselines. We show that clear quality improvements on both shape reconstruction and translation are obtained merely by autoencoding implicit fields rather than point clouds. Further, adding position awareness through latent grids leads to more natural shape translation, with better preservation of spatial features and fine details.

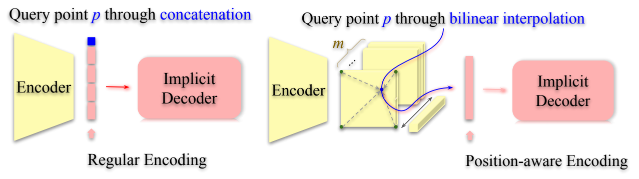
2. Related work

Image translation. Image-to-image translation can be done under a paired or unpaired setting. An example of a paired translation is pix2pix [10] that employs a conditional GAN with a reconstruction loss. Unpaired translation is more complex and usually requires additional loss functions such as cycle consistency [26, 29]. While most methods [26, 29] operate in image space, works such as [16] have utilized a shared latent space to transfer more structural changes across two domains. However, such methods tend to produce less spatially aware local changes. Our work performs unpaired translation in latent space by using a latent grid with an implicit decoder, instead of a single holistic vector, to allow both structural and spatially aware local changes.

Shape translation. Paired shape-to-shape translation has been studied by P2P-Net [28], while Gao et al. [7] have explored unpaired deformation transfer. UNIST is inspired by LOGAN [27], a general-purpose network for unpaired shape translation. However, our work differs from LOGAN in several significant ways. First, instead of point clouds, we employ implicit representations that can generate topology-varying shape translations. Second, we use position-aware latent grids rather than holistic latent codes that only encode global features. Finally, through extensive experiments, UNIST is shown to produce higher-quality reconstruction and more natural shape translation.

Deep implicit functions. Implicit representations have gained immense popularity in the machine learning community and have recently been applied to 3D vision [3, 18, 20], images [19, 23, 24] and dynamic scenes [13, 21, 25]. Deep implicit representations can be broadly divided into three categories: global, local and hierarchical methods. Global methods use a single latent code for all location-based query points to decode objects and have been explored in works such as [3, 6, 12, 18, 20]. Whereas local implicit methods obtain a different latent code for each location-based query point by either tri-linearly interpolating on the encoder feature grid at a given point [5, 22] or just assigning a different latent code to each local part [1, 8, 11]. Recently, hierarchy-based methods have also been proposed [4, 15] that incorporate level of detail to decode 3D shapes.

Instead of global latent codes, we use latent grids as they have been successful in preserving details in autoencoding [5, 22]. Since our main goal is to perform *translation* and translation at each query point is computationally expensive, we instead query latent grids for the decoder and perform the translation directly on latent grids.



(a) Regular encoding. (b) Position-aware encoding.

Figure 3. Regular vs. position-aware encoding. (a) The regular latent encoding is directly concatenated with point coordinate p to predict inside/outside value. (b) The position-aware latent encoding is bilinear-interpolated from the latent grid centered at the point coordinate p to predict inside/outside value.

3. Methods

In this section, we detail our design of UNIST, an implicit model for general-purpose, unpaired shape-to-shape translation in both 2D and 3D domains. As shown in Figure 2, the network encodes input shapes from source and target domains with position-aware encoding and performs translation in a latent grid space, as guided by a set of losses to ensure domain translation as well as feature preservation.

3.1. Neural implicit shapes

Our network employs a neural implicit representation of shapes, by learning inside/outside or occupancy information [3, 18]. To reconstruct such shapes, the volumetric space around shapes is sampled. Denoting the input shape as s and a query point as $p \in \mathbb{R}^3$, the implicit field f is defined as: $f(p) = D(E(s), p) \rightarrow [0, 1]$, where query point p along with the latent code of the input shape that is obtained from an encoder E are passed to a decoder D to predict 0/1 indicating inside/outside values. Then a mesh surface can be extracted via Marching Cubes [17].

3.2. Position-aware encoding

We adopt the idea of position-aware encoding from prior works, e.g., [4, 22], where the input shape is encoded into a latent grid $\mathcal{Z}$ that offers a latent code for each sample point in the space according to its position via linear interpolations. Instead of using a regular latent vector that is agnostic to spatial information, latent grids capture structural information about the input shape; see Figure 3.

More specifically, as illustrated in Figure 2 (top), given an input shape as a 2D binary image with size of $n \times n$, we first map it into a latent grid $\mathcal{Z}$ with size of $k \times k \times m$ using an encoder E that consists of several 2D strided convolutions, and m is the feature dimension. This latent grid $\mathcal{Z}$ encodes a compressed representation of the input shape with high-level spatial information. We then use bilinear interpolation to extract a latent code with size of $m \times 1$ from the latent grid $\mathcal{Z}$ for query point p according to its spatial position. This differs from [3, 18, 20] where all the query points share the same latent vector, which results in networks paying more attention to global information and overlooking local details. A consequence of the absence of structural information in such an encoding is that when translating the latent space, a minor change in the latent vector could result in an unreasonable transform since all the information about the input shape is encoded in a single latent code.

We choose IM-NET [3] as our implicit decoder D to decode the latent code interpolated from latent grid $\mathcal{Z}$ and predict inside/outside value for each point p. We optimize the position-aware encoding model for the reconstruction task using a weighted mean squared error between the ground truth SDF $\bar{s_p}$ and predicted SDF for query point p. Let $\mathcal{S}$ be the training shapes and $\mathcal{P}$ be a set of points sampled on each shape, we then define the reconstruction loss as:

$$\mathcal{L}_{recon} = \frac{1}{|\mathcal{S}|}\frac{1}{|\mathcal{P}|}\sum_{s\in\mathcal{S}}\sum_{p\in\mathcal{P}} |(D(\delta(E(s),p)) - \bar{s_p}) \cdot w_p|^2 \tag{1}$$

where $\delta(\cdot)$ denotes trilinear interpolation in the 3D case and bilinear interpolation in the 2D case, and w_p denotes the weight assigned to point p during sampling. More specifically, we set weights of points sampled near the boundary to 2 whereas weights of other points are set to 1. After training the autoencoder, we use its encoder and decoder as pre-trained networks for the translation task.

3.3. Position-aware translation

Unlike LOGAN, our generator takes the latent grid feature $\mathcal{Z}$ and translates it to match the target domain distribution while the discriminator learns the real distribution of the target domain over the *entire* grid space; see Figure 2. That is, given latent grids $\mathcal{Z}_{\chi_1}$ and $\mathcal{Z}_{\chi_2}$ with size $k \times k \times m$ from domains χ_1 and χ_2, the generator G learns to translate $\mathcal{Z}_{\chi_1}$ to $\mathcal{Z}_{\chi_{1\to 2}}$ in the adversarial setting and $\mathcal{Z}_{\chi_2}$ to $\mathcal{Z}_{\chi_{2\to 2}}$ with feature preservation loss. Unlike LOGAN [27] whose discriminator works with a global latent vector and outputs a single scalar value, our discriminator D takes in $\mathcal{Z}_{\chi_{1\to 2}}$ and outputs a scalar value for each latent vector in our $k \times k$ grid indicating whether features of each grid position are real or not. This results in more regulated translated feature vectors capable of carrying local and spatial information.

We optimize the translation network via losses over the latent grid space:

$$\mathcal{L}_{trans} = \mathcal{L}_{\chi_1\to\chi_2} + \mathcal{L}_{\chi_2\to\chi_1} + \gamma\mathcal{L}_{cycle} \tag{2}$$

The loss function $\mathcal{L}_{\chi_1\to\chi_2}$ for translating domain $\chi_1 \to \chi_2$ is defined as follows, $\mathcal{L}_{\chi_2\to\chi_1}$ can be easily defined by switching the domains:

$$\mathcal{L}_{\chi_1\to\chi_2} = \mathcal{L}^{WGAN}_{\chi_1\to\chi_2} + \alpha\mathcal{L}^{GP}_{\chi_1\to\chi_2} + \beta\mathcal{L}^{FP}_{\chi_1\to\chi_2} \tag{3}$$

where $\mathcal{L}^{WGAN}_{\chi_1\to\chi_2}$ along with $\mathcal{L}^{GP}_{\chi_1\to\chi_2}$ are included in the usual WGAN loss with gradient penalty [9] and $\mathcal{L}^{FP}_{\chi_1\to\chi_2}$ is the feature preservation loss:

$$\mathcal{L}^{WGAN}_{\chi_1\to\chi_2} = \mathop{\mathbb{E}}_{z_1\sim\mathbb{P}(\mathcal{Z}_{\chi_1})}[D(G(z_1))] - \mathop{\mathbb{E}}_{z_2\sim\mathbb{P}(\mathcal{Z}_{\chi_2})}[D(z_2)] \tag{4}$$

$$\mathcal{L}^{FP}_{\chi_1\to\chi_2} = \mathop{\mathbb{E}}_{z_2\sim\mathbb{P}(\mathcal{Z}_{\chi_2})}[||G(z_2) - z_2||_1] \tag{5}$$

We enforce the network to naturally translate between two domains via cycle consistency loss defined as:

$$\begin{aligned}\mathcal{L}_{cycle} = &\mathop{\mathbb{E}}_{z_1\sim\mathbb{P}(\mathcal{Z}_{\chi_1})}[||G_{2\to 1}(G_{1\to 2}(z_1)) - z_1||_1] \\ &+ \mathop{\mathbb{E}}_{z_2\sim\mathbb{P}(\mathcal{Z}_{\chi_2})}[||G_{1\to 2}(G_{2\to 1}(z_2)) - z_2||_1]\end{aligned} \tag{6}$$

where $G_{1\to 2}$ and $G_{2\to 1}$ are used for $\chi_1 \to \chi_2$ and $\chi_2 \to \chi_1$ translations, respectively.

3.4. Implementation and training details

In 2D experiments, we use $n = 256$ pixels, $k = 2$, and $m = 64$. For training our autoencoding network, we use a simple 2D conv-encoder where each layer downsamples images by half, and doubles the number of feature channels. For training the translation network, we use a generator with five 2D convolutional layers and a discriminator with four 2D convolutional layers. For autoencoding, we train all the 2D experiments for 800 epochs with batch size 24, and use Adam optimizer and initial learning rate 0.00005. We decay the learning rate by half after 400 epochs. For translation, we train the generators and discriminators for 1,200 epochs with batch size 128, we again use Adam optimizer with initial learning rate 0.002 and we halve the learning rate every 100 epochs until it reaches to 0.0005. We empirically set $\alpha = 10$, $\beta = 20$ and $\gamma = 20$ for Equation (2) and (3).

We run all tests on a machine with two Nvidia GeForce GTX 1080 Ti GPUs. Training the autoencoder and translator networks takes 10 hours and 30 minutes, respectively, on 2D data. For 3D data, the times are 24 hours and 56 minutes, averaged over the object categories. Inference is fast: 0.2s per 2D shape; 2.7s and 4.2s for 3D outputs at 64^3 and 256^3 resolutions. More details on the network architecture and training can be found in the supplementary material.

4. Experiments and results

We first validate our network design by investigating the behavior of the regular encoding and position-aware encoding in Section 4.1. We then demonstrate the superior performance of UNIST on 2D shape translation in Section 4.2 and 3D shape translation in Section 4.3.

4.1. Ablation study

To verify the efficacy of position-aware encoding in translating shapes and preserving local details, we compare with our baseline, the regular encoding model, that is built on implicit autoencoding without position awareness.

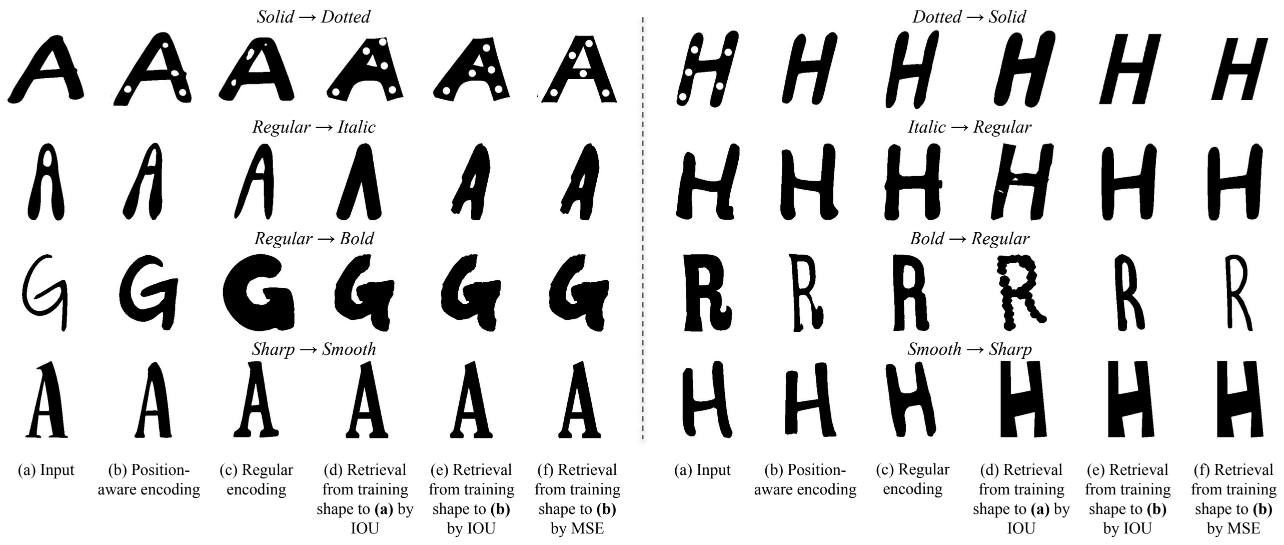

Solid → Dotted
Dotted → Solid
Regular → Italic
Italic → Regular
Regular → Bold
Bold → Regular
Sharp → Smooth
Smooth → Sharp
(a) Input
(b) Position-aware encoding
(c) Regular encoding
(d) Retrieval from training shape to (a) by IOU
(e) Retrieval from training shape to (b) by IOU
(f) Retrieval from training shape to (b) by MSE

Figure 4. Qualitative comparison of position-aware encoding, regular encoding and retrieval results on different font shape datasets. First row (left): $Solid \rightarrow Dotted$, (right): $Dotted \rightarrow Solid$. Second row (left): $Regular \rightarrow Italic$, (right): $Italic \rightarrow Regular$. Third row (left): $Regular \rightarrow Bold$, (right): $Bold \rightarrow Regular$. Fourth row (left): $Sharp \rightarrow Smooth$, (right): $Smooth \rightarrow Sharp$. (a) Test input image. (b) Translation results by position-aware encoding. (c) Translation results by regular encoding. (d) Retrieved training shapes from the *target* domain that are closest to test inputs (a) based on IOU measurement. Retrieved training shapes from the *target* domain that are closest to position-aware encoding translation (b) based on (e) IOU measurement and (f) MSE measurement.

Regular encoding. Our baseline model utilizes regular encoding (Figure 3a) as opposed to position-aware encoding (Figure 3b). The input shape is encoded into four sub-vectors with size of $m/4 \times 1$ from different convolutional layers in the encoder. These sub-vectors are concatenated to form an overcomplete latent code with size of $m \times 1$ that is passed to the implicit decoder along with the coordinates of query point p to predict its inside/outside value.

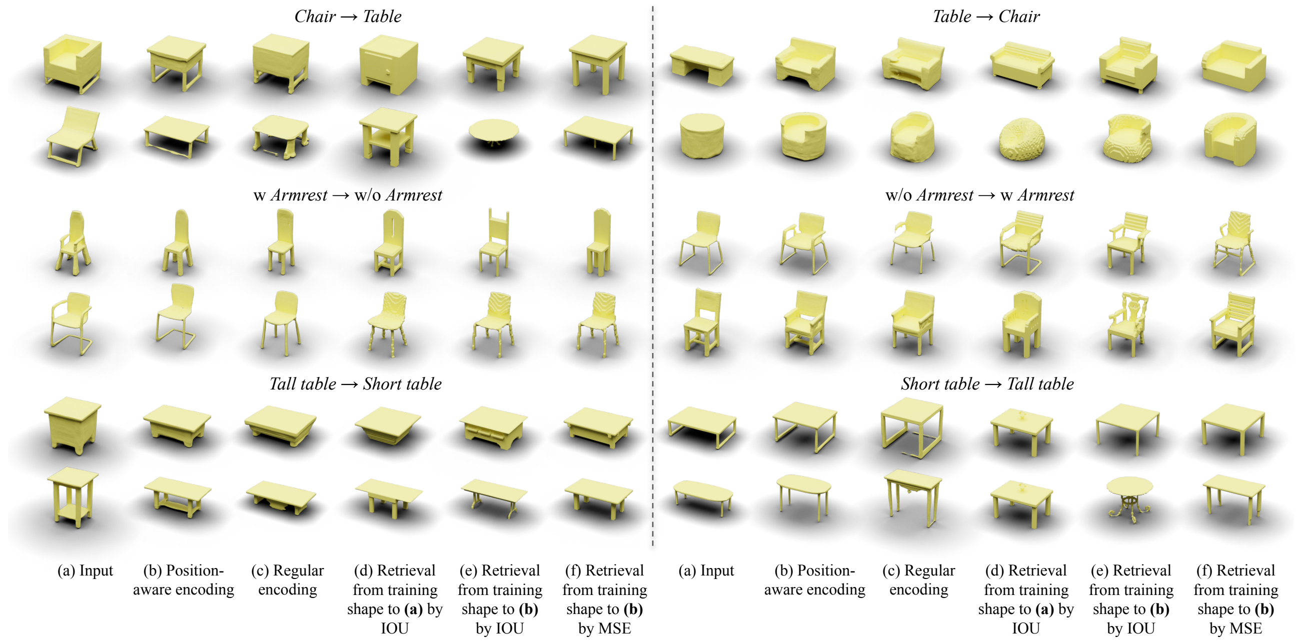
Regular vs. position-aware encoding. We compare translation results of the regular and position-aware encoding on four 2D cross-domain datasets: $Solid \leftrightarrow Dotted$, $Regular \leftrightarrow Italic$, $Regular \leftrightarrow Bold$ and $Sharp \leftrightarrow Smooth$. In these datasets, local geometric features, e.g., sharp/round corner and local curvature, are more prominent, resulting in content-preserving style transfer, which is suitable for validating the importance of position awareness in translation. We follow the same train and test split as in [27]. As shown in Figure 4 (b) and (c), it is evident that regular encoding manages to translate the input shapes into the target domain with fairly clean boundaries and is able to generate compact shapes. However, regular encoding is less capable of preserving the local geometric characteristics of each font (e.g., missing local curvature and irregular dots). We make use of three 3D cross-domain datasets: $Chair \leftrightarrow Table$, chair with $Armrest$ $\leftrightarrow$ without $Armrest$ and $Tall\,table \leftrightarrow Short\,table$ with same train and test split as [27] to further validate the position awareness. Similar observations are valid for 3D translation results illustrated in Figure 5 (b) and (c) where our network benefiting from position-aware encoding can produce results whose geometric and structural features have been better preserved.

Retrieval. We show that UNIST is indeed aware of structural information embedded in the latent grid. That is, it generates a shape that is as similar to the input shape as possible and alters the shape to match the most distinctive features of the target domain. It is evident from Figures 4 (e-f), that translations with position-aware encoding are quite different from retrieved training shapes in the target domain. We also show retrieval results from the target domain that best match the test inputs and the translations with position-aware encoding on the 3D datasets in Figures 5 (d-f). One may notice that results in row 3, column (b) are similar to those in column (f), and the same in row 5. However, we can still observe differences in local details.

4.2. Translation on 2D shapes

For 2D shape translation, we compare UNIST to several unpaired cross-domain image-to-image translation networks: LOGAN [27], CycleGAN [29] and GANHopper [14] on four datasets: $Solid \leftrightarrow Dotted$, $A \leftrightarrow H$, $G \leftrightarrow R$, and $M \leftrightarrow N$. Note that LOGAN uses point cloud representation for translation, hence, we fill the convex hull of the point clouds and convert to images for fair comparison.

Figure 6 shows the qualitative comparison of translation results. We observe that CycleGAN and GANHopper are less capable of learning both content-preserving style trans-



Chair → Table
Table → Chair
w *Armrest* → w/o *Armrest*
w/o *Armrest* → w *Armrest*
Tall table → Short table
Short table → Tall table
(a) Input
(b) Position-aware encoding
(c) Regular encoding
(d) Retrieval from training shape to **(a)** by IOU
(e) Retrieval from training shape to **(b)** by IOU
(f) Retrieval from training shape to **(b)** by MSE

Figure 5. Qualitative comparison of position-aware encoding, regular encoding and retrieval results on different 3D shape datasets. Row 1-2 (left): $Chair \rightarrow Table$, (right): $Table \rightarrow Chair$. Row 3-4 (left): with $Armrest \rightarrow$ without $Armrest$, (right): without $Armrest \rightarrow$ with $Armrest$. Row 5-6 (left): $Tall\,table \rightarrow Short\,table$, (right): $Short\,table \rightarrow Tall\,table$. (a) Test input. (b) Translation results by position-aware encoding. (c) Translation results by regular encoding. (d) Retrieved training shapes from the *target* domain that are closest to test inputs (a) based on IOU measurement. Retrieved training shapes from the *target* domain that are closest to position-aware encoding translation (b) based on (e) IOU measurement and (f) MSE measurement.

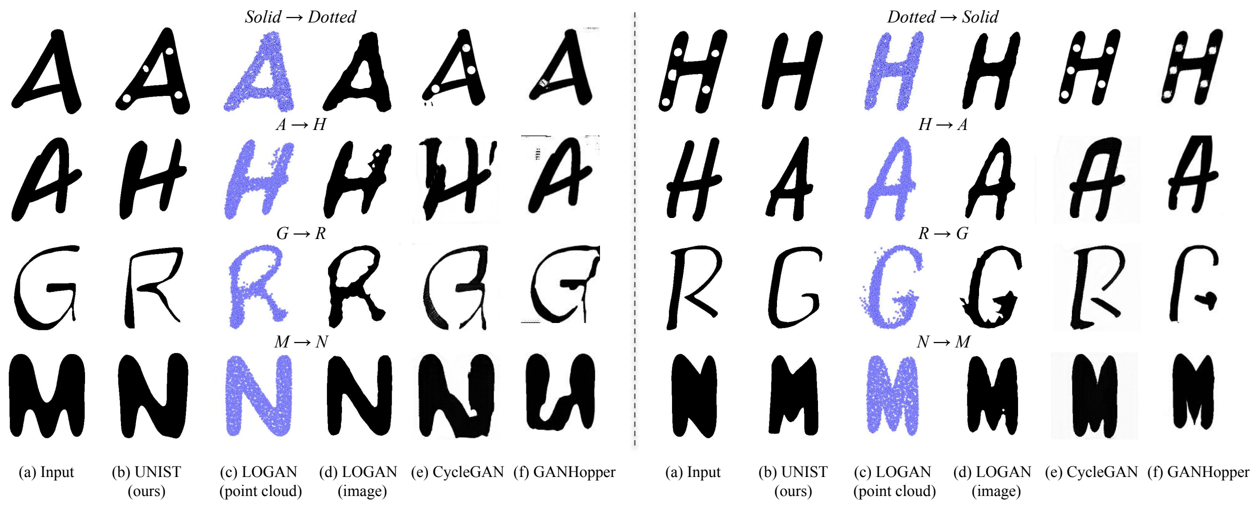


Figure 6. Comparison of translation results by UNIST (ours), LOGAN, CycleGAN and GANHopper on different 2D shapes. First row (left): $Solid{\rightarrow}Dotted$, (right): $Dotted{\rightarrow}Solid$. Second row (left): $A{\rightarrow}H$, (right): $H{\rightarrow}A$. Third row (left): $G{\rightarrow}R$, (right): $R{\rightarrow}G$. Fourth row (left): $M{\rightarrow}N$, (right): $N{\rightarrow}M$. (a) Test input. Translations by (b) UNIST, (c) LOGAN in point cloud representation, (d) LOGAN in image, (e) CycleGAN and (f) GANHopper.

fer ($Solid \leftrightarrow Dotted$) and style-preserving content transfer ($A \leftrightarrow H$, $G \leftrightarrow R$ and $M \leftrightarrow N$) while LOGAN manages to preserve and transfer features, it sometimes produces scattered point clouds, resulting in less compact shapes. Our proposed UNIST, on the other hand, is capable of producing shapes with significantly better visual quality, as it can reproduce small scale stylistic features as well as preserve topological features such as the dots in the dotted fonts.

	$A \leftrightarrow H$		$G \leftrightarrow R$		$M \leftrightarrow N$	
	MSE ↓	IoU ↑	MSE ↓	IoU ↑	MSE ↓	IoU ↑
CycleGAN	0.246	0.385	0.229	0.412	0.266	0.383
UNIT	0.253	0.376	0.264	0.377	0.295	0.348
MUNIT	0.280	0.286	0.358	0.171	0.363	0.292
LOGAN	0.195	0.490	0.213	0.472	0.207	0.506
GANHopper	0.258	0.384	0.268	0.380	0.296	0.356
Regular encoding (ours)	**0.184**	**0.507**	**0.197**	**0.493**	**0.201**	**0.508**
Position-aware encoding (ours)	0.215	0.441	**0.211**	0.452	0.233	0.433

Table 1. Quantitative comparisons between unpaired translation networks on A-H, G-R and M-N where *one possible* ground-truth (GT) target is available. For each domain pair, Mean Squared Error (MSE) and Intersection over Union (IoU) are measured against *that* GT target letter and averaged over the translation in both directions (e.g. average over $A \rightarrow H$ and $H \rightarrow A$). ↓ means the lower the better and ↑ means the higher the better.

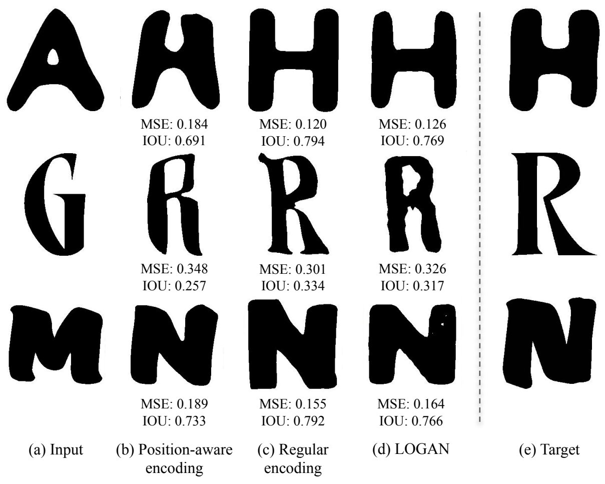


Figure 7. Visual comparisons between LOGAN and UNIST (regular vs. position-aware encodings). The target letters (e) are of the same font as the respective inputs (a). While position-aware encoding appears to produce more natural translations, with better spatial feature and style preservation, it is outperformed by the other two on MSE and IOU, measured against the targets.

Quantitative comparison. We need ground-truth (GT) for a quantitative study. For $A \leftrightarrow H$, $G \leftrightarrow R$, and $M \leftrightarrow N$, a natural GT target would be letters from the same font family. Measured against that GT, using Mean Squared Error (MSE) and Intersection over Union (IoU), we show quantitative comparisons between various unpaired translation networks in Table 1. As we can see, UNIST with regular encoding beats all competitors, including LOGAN, owing to the use of neural implicit shapes. On the other hand, the use of position-aware encoding underperforms against LOGAN, but still improves over other baselines.

By a qualitative comparison shown in Figure 7, we observe that position-aware encoding tends to preserve spatial and stylistic characteristics of the input as much as possible, only altering it in ways that are deemed most "critical" to reach the target domain. For the $A \leftrightarrow H$ example, only a small breaking was introduced at the top. Note that in Figure 7, the regular encoding results may look closer to the targets. However, approaching the target is *not* what the translators are trained to do as UNIST is fully *unsupervised* and the translation results should be qualitatively judged by how well they preserve input features. Overall, we find position-aware encoding to produce more natural translation than regular encoding and LOGAN, e.g, see the $G \leftrightarrow R$ and $M \leftrightarrow N$ examples in Figure 7. Its underperformance in Table 1 may be attributed to the strong feature preservation or the inadequacies of MSE and IoU as viable perceptual metrics as they only measure spatial distortions.

4.3. Translation on 3D shapes

We conduct 3D experiments on $Chair \leftrightarrow Table$, chair with $Armrest \leftrightarrow$ without $Armrest$ and $Tall\,table \leftrightarrow Short\,table$ from ShapeNet [2], and compare our method to LOGAN [27], as it is the state-of-the-art unpaired shape-to-shape translation network. We use Marching Cubes [17] to obtain a mesh from the output, sampled at 256^3 resolution. Note that we employ the sampling strategy from [3] to obtain 2,048 points from the surfaces of the meshes to fairly compare with LOGAN, as it only produces point clouds at 2,048 resolution. Figure 8 shows the qualitative results.

Quantitative evaluations present challenges again, since shape translation is inherently a domain-specific task. What a *correct* translation is can be highly varied, depending on the shape semantics from the two chosen domains and the nature of the translation itself. As a result, we provide different ways to evaluate 3D translations as follows.

When translating between chairs with and without armrests (third and fourth rows of Figure 8), our natural expectation is that the network should only add/remove the armrests while preserving the input. We treat this as the GT scenario and measure the quality of a translation using the *one-sided Chamfer Distance* (CD) from the armrest-less chair to its corresponding chair with armrests, regardless of the direction of the translation. The numbers given in Table 3 show that UNIST outperforms both our baseline and LOGAN on this metric, demonstrating that it better learns the essential difference between both domains.

In the case of $Table \leftrightarrow Chair$ translations, the network is tasked to not only modify the geometry of the input shape, but also change its semantics. Owing to this, it is hard to quantify a good result, motivating a user study to measure the quality of the translation. In our user study, we asked 72 participants via Amazon Mechanical Turk to rank the quality of translations performed by LOGAN [27] and UNIST, using regular or position-aware encoding. We report the study results in Table 2, which show that the par-

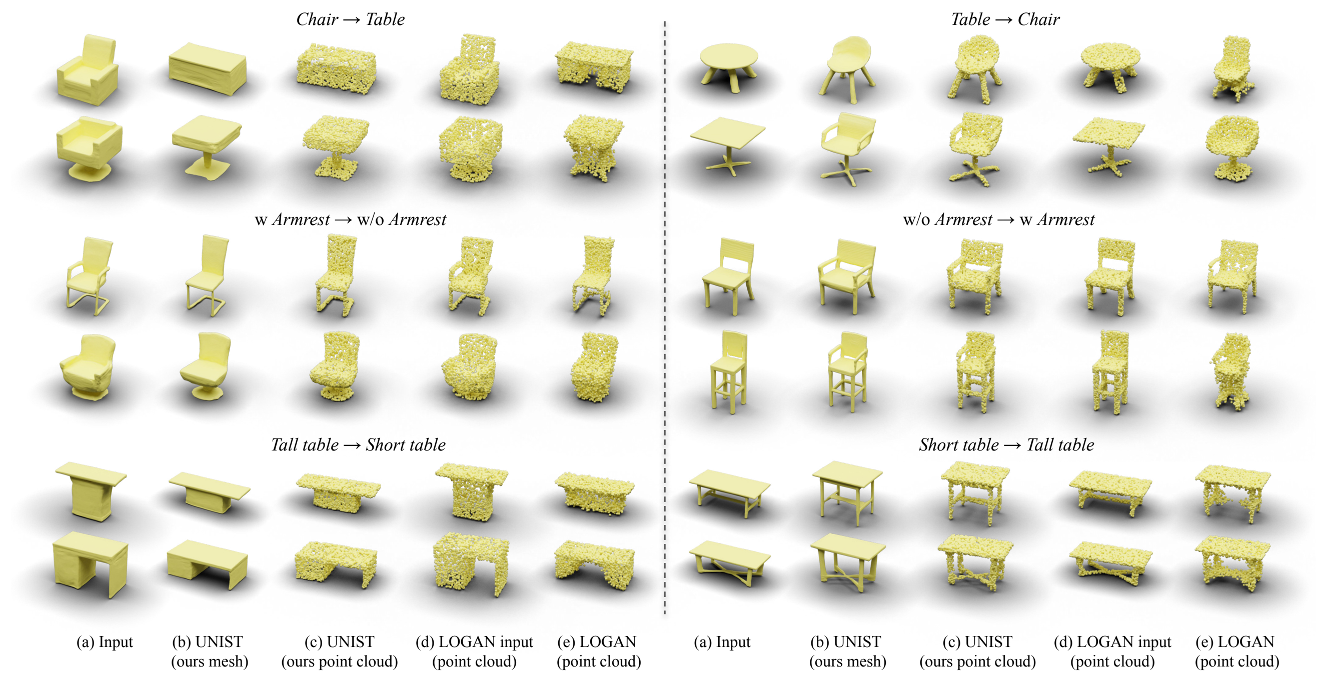


Figure 8. Comparison of translation results by UNIST (ours) and LOGAN on different 3D shapes. Row 1-2 (left): $Chair \rightarrow Table$ (right): $Table \rightarrow Chair$. Row 3-4 (left): w $Armrest \rightarrow$ w/o $Armrest$ (right): w/o $Armrest \rightarrow$ w $Armrest$. Row 5-6 (left): $Tall\,table \rightarrow Short\,table$, (right): $Short\,table \rightarrow Tall\,table$. (a) Test input mesh from voxel. (b) Translation by UNIST (ours) in mesh representation. (c) Translation by UNIST (ours) in point cloud representation. (d) LOGAN test input point cloud. (e) Translation by LOGAN.

	Chair → Table			Table → Chair		
	1st	2nd	3rd	1st	2nd	3rd
LOGAN	30.55%	35.68%	33.25%	35.94%	30.47%	33.51%
Regular	31.34%	30.38%	37.67%	27.78%	36.37%	36.02%
Position-aware	**38.11%**	33.94%	29.08%	**36.28%**	33.16%	30.47%

Table 2. User study on $Chair \leftrightarrow Table$ via Amazon Mechanical Turk. Turkers were asked to rank translation results generated by different methods. % are relative to the total votes given per rank.

ticipants were most likely to choose UNIST with position-aware encoding as the best translation method over both directions. At the same time, position-aware UNIST was also least likely to be ranked as the worst of the three compared methods. Still, the gains are somewhat marginal, which is not entirely surprising given the ambiguity and subjectivity over how to judge what a good translation is.

5. Conclusion, limitation, and future work

We show that the popular neural implicit representations are well suited to the task of unpaired shape-to-shape translation, under the general framework of latent overcomplete GANs (LOGAN) [27]. Improvements of UNIST over its point cloud counterpart are evident, especially when the translation or reconstruction involves finer details and topological changes. In addition, incorporating position-aware encoding into the design further strengthens the translation network in terms of feature preservation.

	w Arm → w/o Arm	w/o Arm → w Arm
LOGAN	0.0249	0.0273
Regular	0.0255	0.0267
Position-aware	**0.0234**	**0.0235**

Table 3. One-sided CD for translations between shapes with and without armrests to measure how well the common parts between both shapes are preserved. In the first result column, the one-sided distance was calculated as *Output* → *Input* and the second column represents *Input* → *Output*. In all calculations, we sample 2,048 points from the meshes to ensure a fair comparison to LOGAN.

On the other hand, implicit functions are not as apt at representing geometric structures such as skeletons or profile curves, as point cloud LOGAN would. A more critical limitation however is related to *controllability*, or lack thereof. Hence the main path for future work is to explore few-shot learning and conditional generative modeling with UNIST to guide or constrain the translation network.

Acknowledgements We thank the annoymous reviewers for their comments, and Fenggen Yu, Wallace Lira, and Kangxue Yin for their discussions. This work was supported in part by NSERC (611370) and an Autodesk gift.

References

[1] 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 *ECCV*, 2020. 3

[2] Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An information-rich 3D model repository. *CoRR*, abs/1512.03012, 2015. 7

[3] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *CVPR*, 2019. 1, 3, 4, 7

[4] Zhang Chen, Yinda Zhang, Kyle Genova, Sean Fanello, Sofien Bouaziz, Christian Hane, Ruofei Du, Cem Keskin, Thomas Funkhouser, and Danhang Tang. Multiresolution deep implicit functions for 3d shape representation. In *ICCV*, 2021. 2, 3

[5] Julian Chibane, Thiemo Alldieck, and Gerard Pons-Moll. Implicit functions in feature space for 3d shape reconstruction and completion. In *CVPR*, 2020. 2, 3

[6] Yueqi Duan, Haidong Zhu, He Wang, Li Yi, Ram Nevatia, and Leonidas J Guibas. Curriculum deepsdf. In *ECCV*, 2020. 3

[7] Lin Gao, Jie Yang, Yi-Ling Qiao, Yu-Kun Lai, Paul L Rosin, Weiwei Xu, and Shihong Xia. Automatic unpaired shape deformation transfer. *ACM TOG*, 37(6):1–15, 2018. 3

[8] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas Funkhouser. Local deep implicit functions for 3d shape. In *CVPR*, 2020. 3

[9] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein GANs. In *NeurIPS*, 2017. 4

[10] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 3

[11] Chiyu Jiang, Avneesh Sud, Ameesh Makadia, Jingwei Huang, Matthias Nießner, Thomas Funkhouser, et al. Local implicit grid representations for 3d scenes. In *CVPR*, 2020. 3

[12] Marian Kleineberg, Matthias Fey, and Frank Weichert. Adversarial generation of continuous implicit shape representations. *arXiv preprint arXiv:2002.00349*, 2020. 3

[13] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *CVPR*, 2021. 3

[14] Wallace Lira, Johannes Merz, Daniel Ritchie, Daniel Cohen-Or, and Hao Zhang. Ganhopper: Multi-hop gan for unsupervised image-to-image translation. In *ECCV*, 2020. 5

[15] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In *NeurIPS*, 2020. 3

[16] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *NeurIPS*, 2017. 1, 2, 3

[17] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH*, 1987. 3, 7

[18] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *CVPR*, 2019. 1, 3

[19] 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 *ECCV*. Springer, 2020. 3

[20] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019. 1, 3

[21] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo-Martin Brualla. Deformable neural radiance fields. In *ICCV*, 2021. 3

[22] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *ECCV*, 2020. 2, 3

[23] Vincent Sitzmann, Julien NP Martel, Alexander W Bergman, David B Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *NeurIPS*, 2020. 3

[24] 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. In *NeurIPS*, 2020. 3

[25] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. In *CVPR*, 2021. 3

[26] Zili Yi, Hao Zhang, Ping Tan, and Minglun Gong. Dualgan: Unsupervised dual learning for image-to-image translation. In *ICCV*, 2017. 1, 3

[27] Kangxue Yin, Zhiqin Chen, Hui Huang, Daniel Cohen-Or, and Hao Zhang. LOGAN: Unpaired shape transform in latent overcomplete space. *ACM Trans. on Graphics (TOG)*, 2019. 1, 2, 3, 4, 5, 7, 8

[28] Kangxue Yin, Hui Huang, Daniel Cohen-Or, and Hao Zhang. P2P-NET: Bidirectional point displacement net for shape transform. *ACM Trans. on Graphics (TOG)*, 2018. 3

[29] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 1, 3, 5