UNIST: Unpaired Neural Implicit Shape Translation Network Supplementary Material

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

6. Supplementary

We provide detailed network architectures for both 2D and 3D experiments in Section 6.1, training details in Section 6.2 and dataset details in Section 6.3. Section 6.4 outlines the approach to convert point clouds to images and Section 6.5 the method to convert LOGAN point clouds to meshes. Lastly, Section 6.6 shows user study details and additional results are provided in Section 6.7.

6.1. Implementations

2D Network architecture. Figure 9 shows the detailed network architecture for our 2D experiments. In the autoencoder, given a binary 2D image of size 256×256 as input, several 2D convolutions are used to downsample the input image into a latent grid of size $2 \times 2 \times 64$. We then use bilinear interpolation to extract position-aware encoding given the position of the query point p. This position-aware encoding is passed to several fully connected layers to predict the inside/outside value. The generator takes in the latent grid extracted by the encoder and produces a translated latent grid with the same size as input. The discriminator expects the translated latent grid and outputs a latent grid of size 2×2.

3D Network architecture. Figure 10 shows the detailed network architecture for our 3D experiments. Analogously to the 2D case, given a 3D voxel of size $64 \times 64 \times 64$ as input, the autoencoder uses several 3D convolutions to downsample the input voxel into a latent grid of size $2 \times 2 \times 2 \times 32$. Trilinear interpolation then extracts the position-aware encoding given the position of the query point p. Again, this position-aware encoding is passed to several fully connected layers to predict the inside/outside value. The generator and discriminator are equivalent to the 2D case, adapted to the latent grid size of $2 \times 2 \times 2$.

6.2. Training details

In the 2D experiments, we use $n = 256$ pixels, $k = 2$, and $m = 64$. For autoencoding, we train all the 2D experiments for 800 epochs with batch size 24, and use Adam optimizer with an initial learning rate of 0.00005. We decay the learning rate by half after 400 epochs. For the translation, we train the generators and discriminators for 1,200 epochs with batch size 128, we again use Adam optimizer with an initial learning rate of 0.002 and we halve the learning rate every 100 epochs until it reaches 0.0005. We empirically set $\alpha = 10$, $\beta = 20$ and $\gamma = 20$ for Equation (2) and (3).

In the 3D experiments, we use $n = 64$ voxel resolution, $k = 2$ and $m = 32$. When training the autoencoding for 3D shapes, we employ progressive training [3] on points sampled from voxel grids with different resolutions $(16^3, 32^3, 64^3)$. We train the position-aware encoding model on each resolution with 300, 300 and 600 epochs, respectively. Adam optimizer and an initial learning rate of 0.00005 are used throughout the training of the autoencoding. Here, we train the generators and discriminators for 4,800 epochs with batch size 128 and Adam optimizer with an initial learning rate of 0.002 for the translation. The learning rate is decayed by half every 100 epochs until it reaches 0.0005 and we empirically chose $\alpha = 10$, $\beta = 100$ and $\gamma = 20$ for Equation (2) and (3).

6.3. Datasets

We provide details about the 2D and 3D shape datasets used in the Section 4. For the 2D datasets, to sample ground truth inside/outside values, we first find the boundary of the shapes and then sample $16,384 \times 2/3$ points near the boundary (including points on the boundary) with a radius of 10 pixels. Finally, we randomly sample $16,384 \times 1/3$ points across the rest of the image, so the total number of sample points is 16,384 and the ratio of points near the boundary and the rest of the image is $2 : 1$. Further, we assign the weights (w_p in Equation (2)) of all sampled points to 1. For the $Solid \leftrightarrow Dotted$ dataset, we assign the weights of the points sampled near the boundary to 2 and the weights of the rest of the points to 1. For the 3D dataset, we make use of shapes from ShapeNet [1] and adopt the sampling strategy from [3].

RegularAH $\leftrightarrow$ ItalicAH. Each domain has 4,000 training

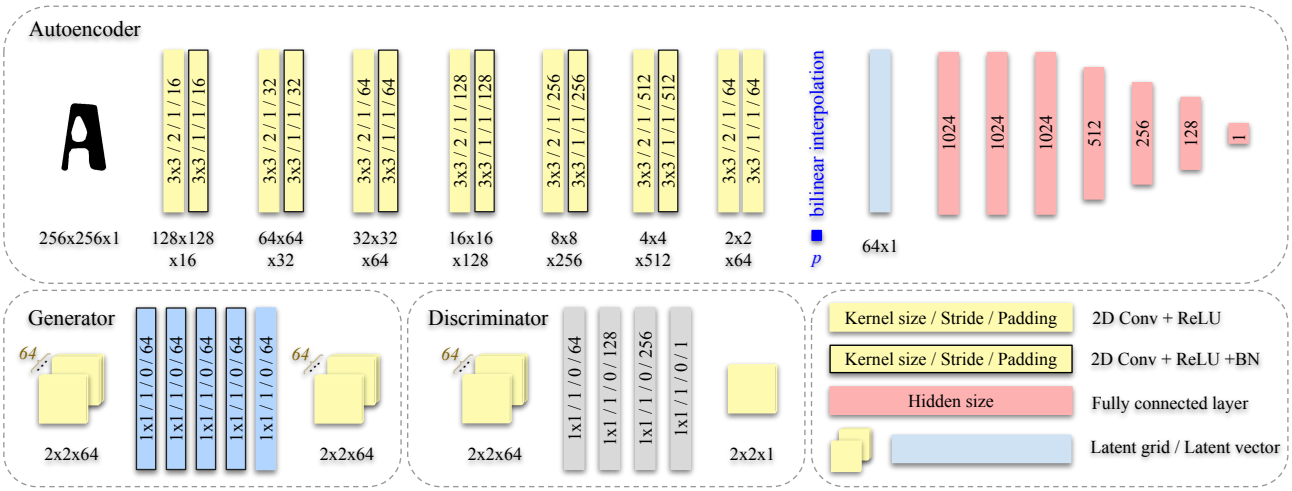

Figure 9. Detailed network architecture for 2D experiments

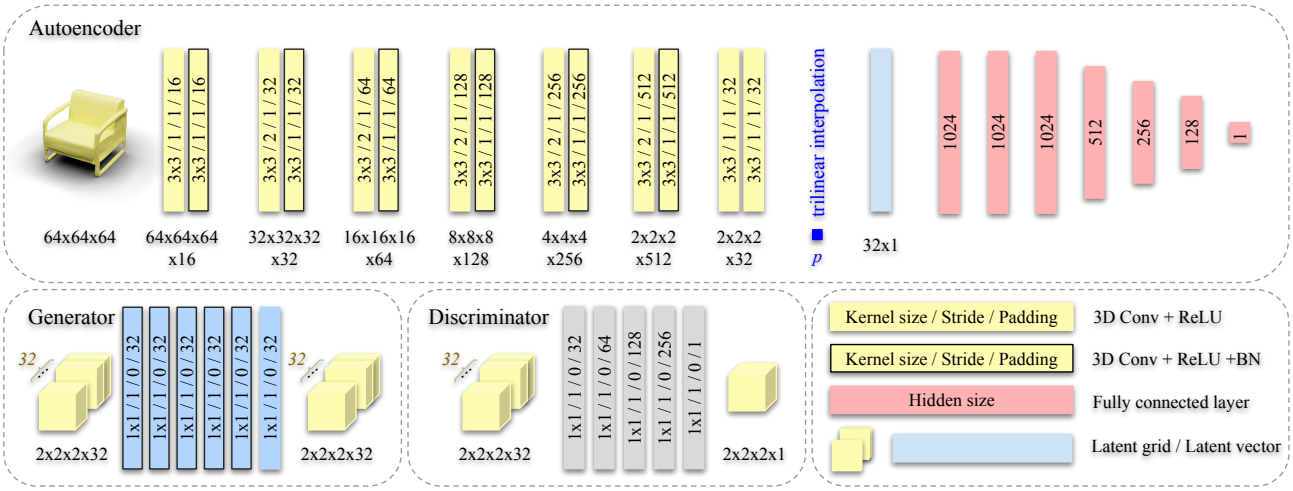

Figure 10. Detailed network architecture for 3D experiments. Note that the input of the autoencoder shown in the figure above is the mesh extracted via Marching Cubes [4] from the original input voxel for better visualization purpose.

images and 1,000 testing images with letters A and H.

SharpAH ↔ SmoothAH. Domain $SharpAH$ has 2,812 training images and 380 testing images with letters A and H. Domain $SmoothAH$ has 3,062 training images and 380 testing images with letters A and H.

RegularGR ↔ BoldGR. Each domain has 4,000 training images and 1,000 testing images with letters G and R.

A ↔ H, G ↔ R, M ↔ N. Each domain has 6,466 training images and 1,000 testing images with letters A, H, G, R, M and N, respectively.

SolidAH ↔ DottedAH. We first collect letters A and H from the above domains. We then randomly divide images into two splits, one for $SolidAH$ with 8,396 images and the other for $DottedAH$ with 8,084 images. To insert dots, we first find the medial axis of the shapes and then keep drawing circles along the medial axis with the radius of 12 pixels if the circle 1) is inside the boundary and 2) does not overlap with other circles, until the shape has five circles. We randomly divide the dataset, so each domain has 224 testing images.

Chair ↔ Table. Domain $Chair$ has 4,768 training shapes and 2,010 testing shapes. Domain $Table$ has 5,933 training shapes and 2,526 testing shapes.

Chair with armrest ↔ without armrest. Domain chair $with\,armrest$ has 1,710 training shapes and 428 testing shapes. Domain chair $without\,armrest$ has 2,857 training shapes and 715 testing shapes.

Tall table ↔ Short table. Each domain has 2,500 training

shapes and 500 testing shapes.

6.4. Point cloud to image

As described in Section 4.2, we convert LOGAN [5] point clouds to images for fair comparison. For each point in a given point cloud, we first find all its neighbors within a radius of $r = 10$, we then compute the convex hull of all the points to get the corresponding image.

6.5. LOGAN point cloud to mesh

To validate that LOGAN followed by an implicit network is inadequate to produce compact translated shapes, we first transform LOGAN point clouds to the IMNET [3] coordinate system, we then voxelize [2] the point clouds and fill the inner volume. Finally, we feed the voxelized point clouds to a pre-trained IMNET [3] to obtain an implicit shape and use Marching Cubes [4] to extract the mesh surface. Figure 18-23 (c) show the translations of LOGAN in mesh representation.

6.6. User study

Section 4.3 describes the user study we performed. We conducted the user study to measure the quality of the translation in the case of $Chair \leftrightarrow Table$. We provide details about task description and randomly selected test shapes for the user study. As LOGAN only produces point clouds at 2,048 resolution, we employ the sampling strategy from [3] to obtain 2,048 points from the surfaces of the meshes to fairly compare with it. In total 72 users participated in the study, half of which were presented with $Chair \leftrightarrow Table$ translations, while the rest was asked to rank $Table \leftrightarrow Chair$ translations.

Task description. The users were shown the following task description before they started the study: "We will show you a chair and then three computer-generated shapes that are supposed to be *table version* of that chair. Specifically, the generated shape should be a table, and further, it should be a table that is as similar to the original chair as possible in terms of its features or characteristics. Using your best judgement, please rank each of the computer-generated *tables* in terms of how well it fulfills the criteria. 1 is the best, 2 is the second best and 3 is the third best." Figure 18-19 (d-g) show the test shapes used in the user study.

6.7. Additional results

In this section we provide additional *randomly* selected results and compare them with other unpaired cross-domain translation networks in the following figures:

- Figure 11 - $A \leftrightarrow H$
- Figure 12 - $G \leftrightarrow R$
- Figure 13 - $M \leftrightarrow N$
- Figure 14 - $SolidAH \leftrightarrow DottedAH$
- Figure 15 - $RegularAH \leftrightarrow ItalicAH$
- Figure 16 - $RegularGR \leftrightarrow BoldGR$
- Figure 17 - $SharpAH \leftrightarrow SmoothAH$
- Figure 18, 19 - $Chair \leftrightarrow Table$
- Figure 20, 21 - $Armrest \leftrightarrow w/o\, Armrest$
- Figure 22, 23 - $Tall\, table \leftrightarrow Short\, table$

References

[1] 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. 1

[2] Zhiqin Chen, Vladimir G Kim, Matthew Fisher, Noam Aigerman, Hao Zhang, and Siddhartha Chaudhuri. Decor-gan: 3d shape detailization by conditional refinement. In *CVPR*, 2021. 3

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

[4] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH*, 1987. 2, 3

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

Figure 11. *Randomly* selected qualitative comparison of translation results by different translation networks on (left) $A \rightarrow H$ and (right) $H \rightarrow A$. (a) Test input, (b) UNIST with position-aware encoding, (c) UNIST with regular encoding, (d) LOGAN in point cloud representation, (e) LOGAN in image representation, (f) CycleGAN and (g) GANHopper.

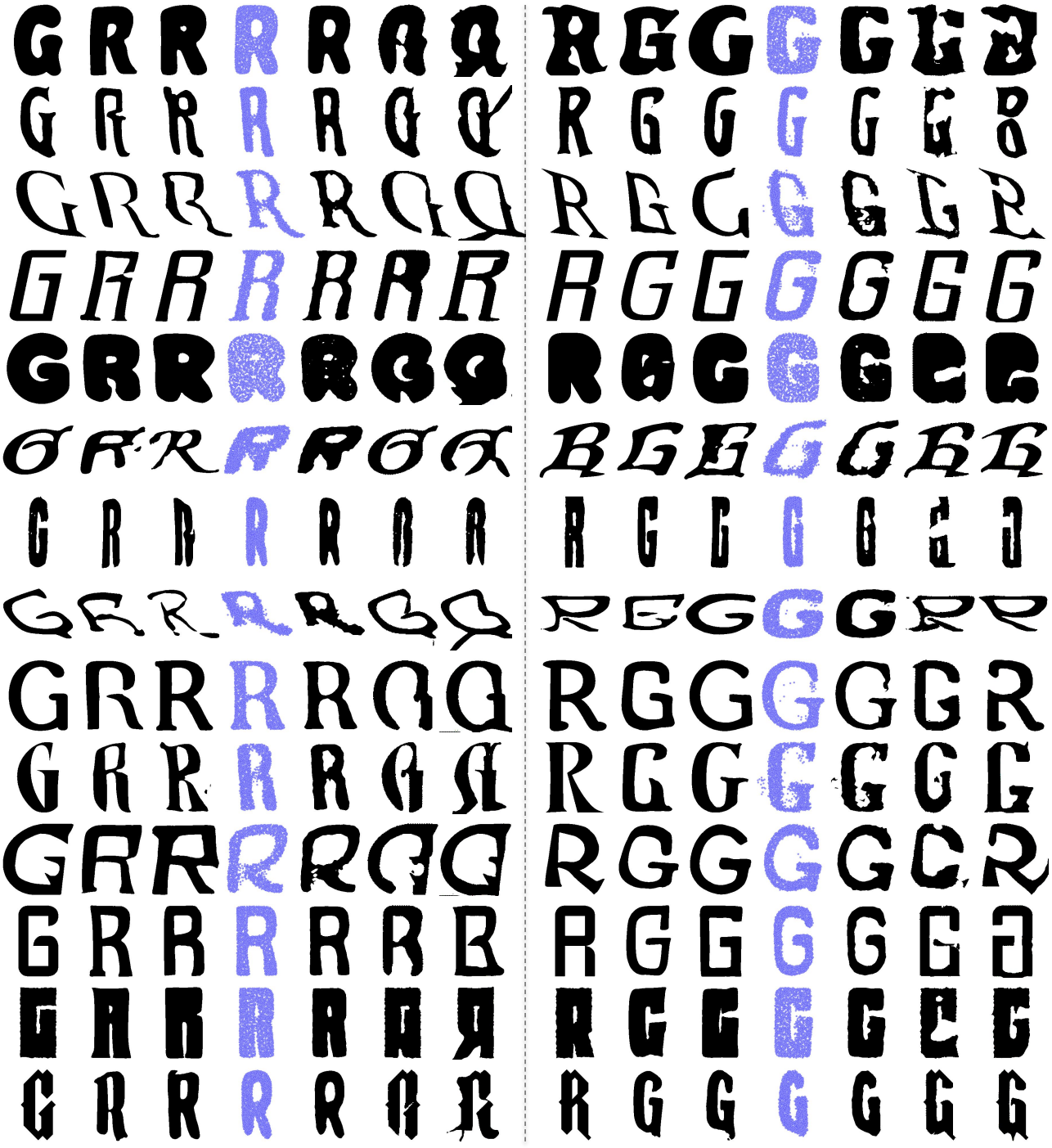

Figure 12. *Randomly* selected qualitative comparison of translation results by different translation networks on (left) $G \rightarrow R$ and (right) $R \rightarrow G$. (a) Test input, (b) UNIST with position-aware encoding, (c) UNIST with regular encoding, (d) LOGAN in point cloud representation, (e) LOGAN in image representation, (f) CycleGAN and (g) GANHopper.

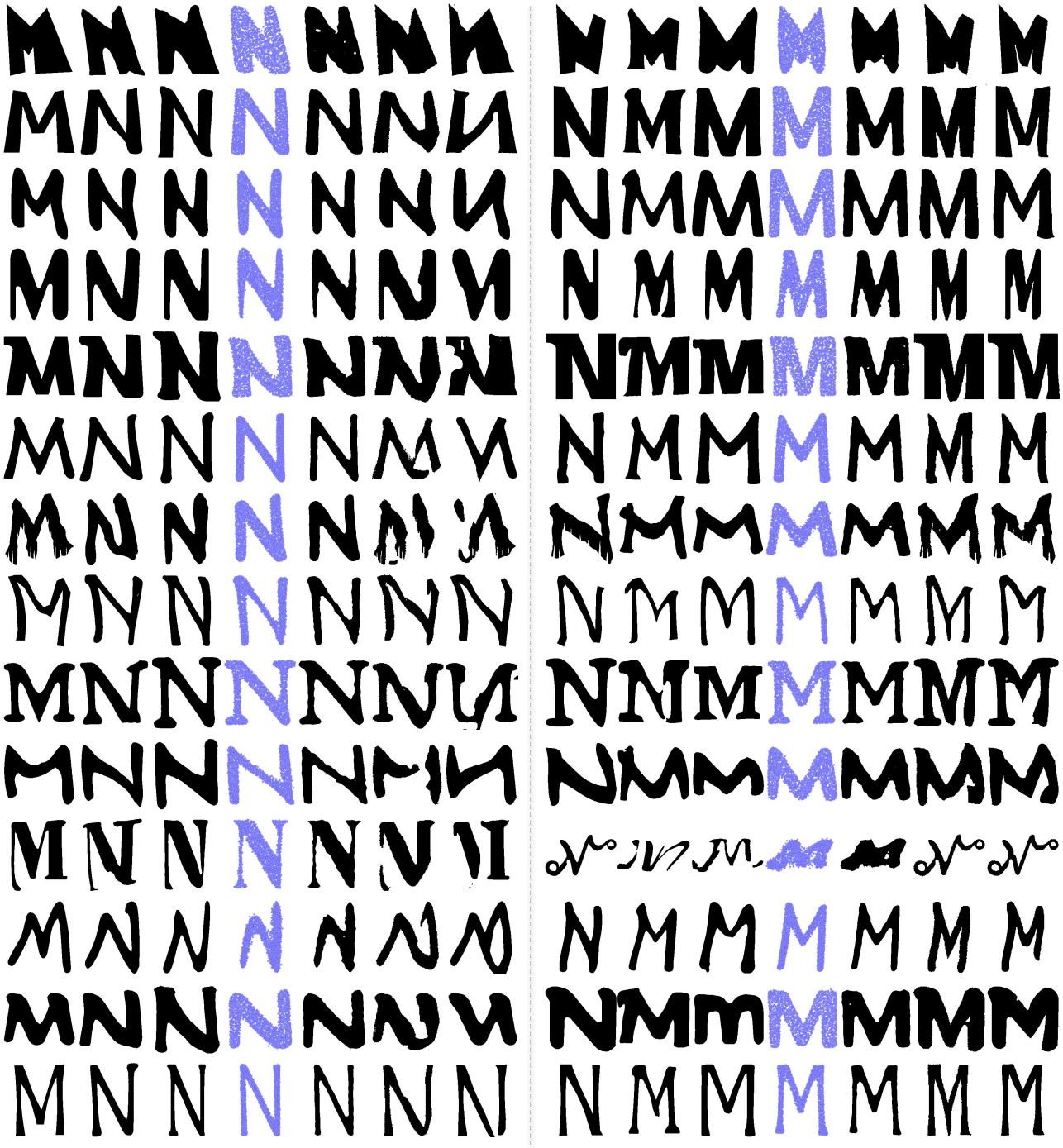

Figure 13. *Randomly* selected qualitative comparison of translation results by different translation networks on (left) $M \rightarrow N$ and (right) $N \rightarrow M$. (a) Test input, (b) UNIST with position-aware encoding, (c) UNIST with regular encoding, (d) LOGAN in point cloud representation, (e) LOGAN in image representation, (f) CycleGAN and (g) GANHopper.

Figure 14. *Randomly* selected qualitative comparison of translation results by different translation networks on (left) $SolidAH \leftrightarrow DottedAH$ and (right) $DottedAH \leftrightarrow SolidAH$. (a) Test input, (b) UNIST with position-aware encoding, (c) UNIST with regular encoding, (d) LOGAN in point cloud representation, (e) LOGAN in image representation, (f) CycleGAN and (g) GANHopper.

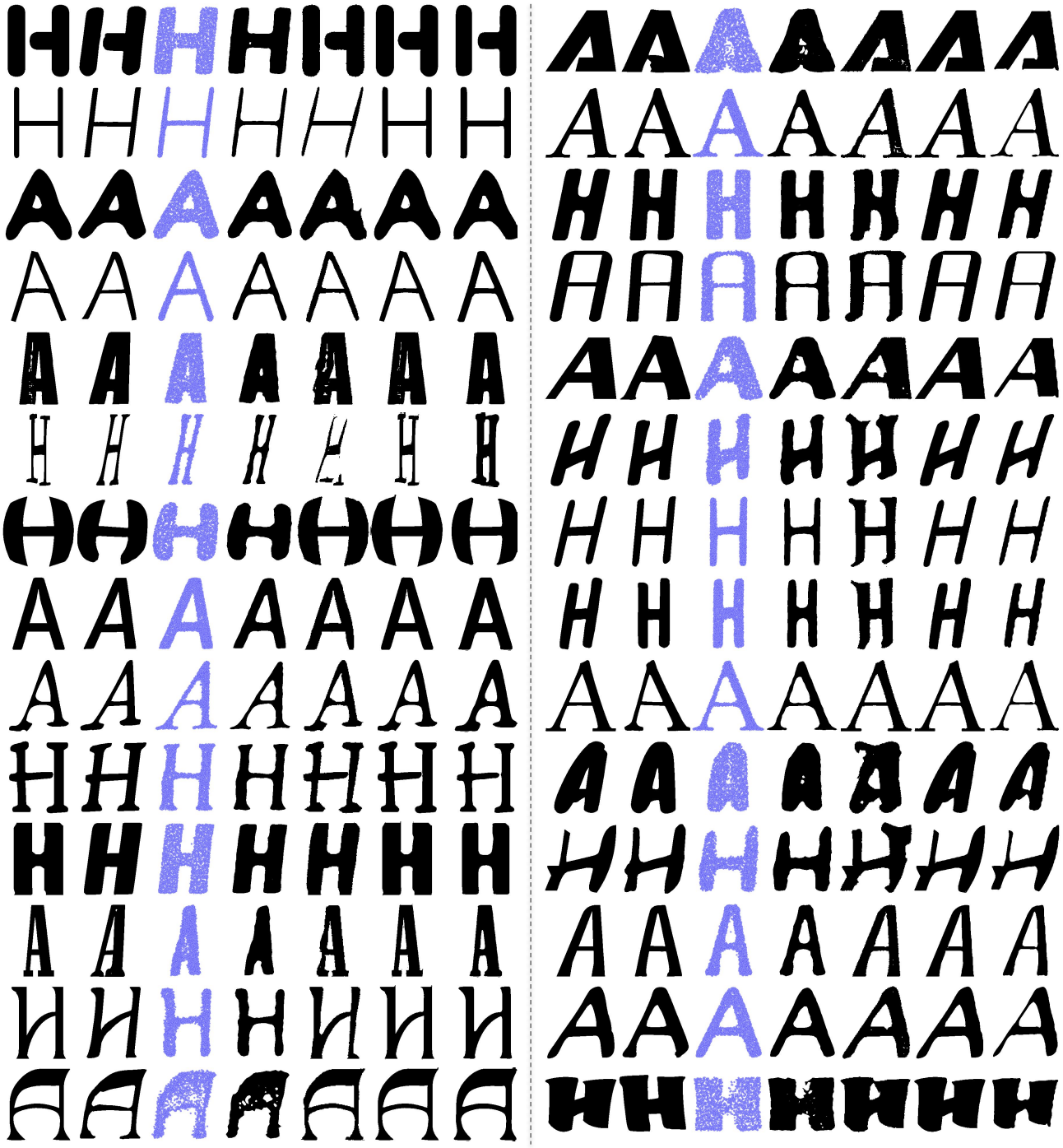

Figure 15. *Randomly* selected qualitative comparison of translation results by different translation networks on (left) $RegularAH \leftrightarrow ItalicAH$ and (right) $ItalicAH \leftrightarrow RegularAH$. (a) Test input, (b) UNIST with position-aware encoding, (c) LOGAN in point cloud representation, (d) LOGAN in image representation, (e) CycleGAN, (f) UNIST and (g) MUNIT.

Figure 16. *Randomly* selected qualitative comparison of translation results by different translation networks on (left) $RegularGR \leftrightarrow BoldGR$ and (right) $BoldGR \leftrightarrow RegularGR$. (a) Test input, (b) UNIST with position-aware encoding, (c) LOGAN in point cloud representation, (d) LOGAN in image representation, (e) CycleGAN, (f) UNIST and (g) MUNIT.

Figure 17. *Randomly* selected qualitative comparison of translation results by different translation networks on (left) $SharpAH \leftrightarrow SmoothAH$ and (right) $SmoothAH \leftrightarrow SharpAH$. (a) Test input, (b) UNIST with position-aware encoding, (c) UNIST with regular encoding, (d) LOGAN in point cloud representation and (e) LOGAN in image representation.

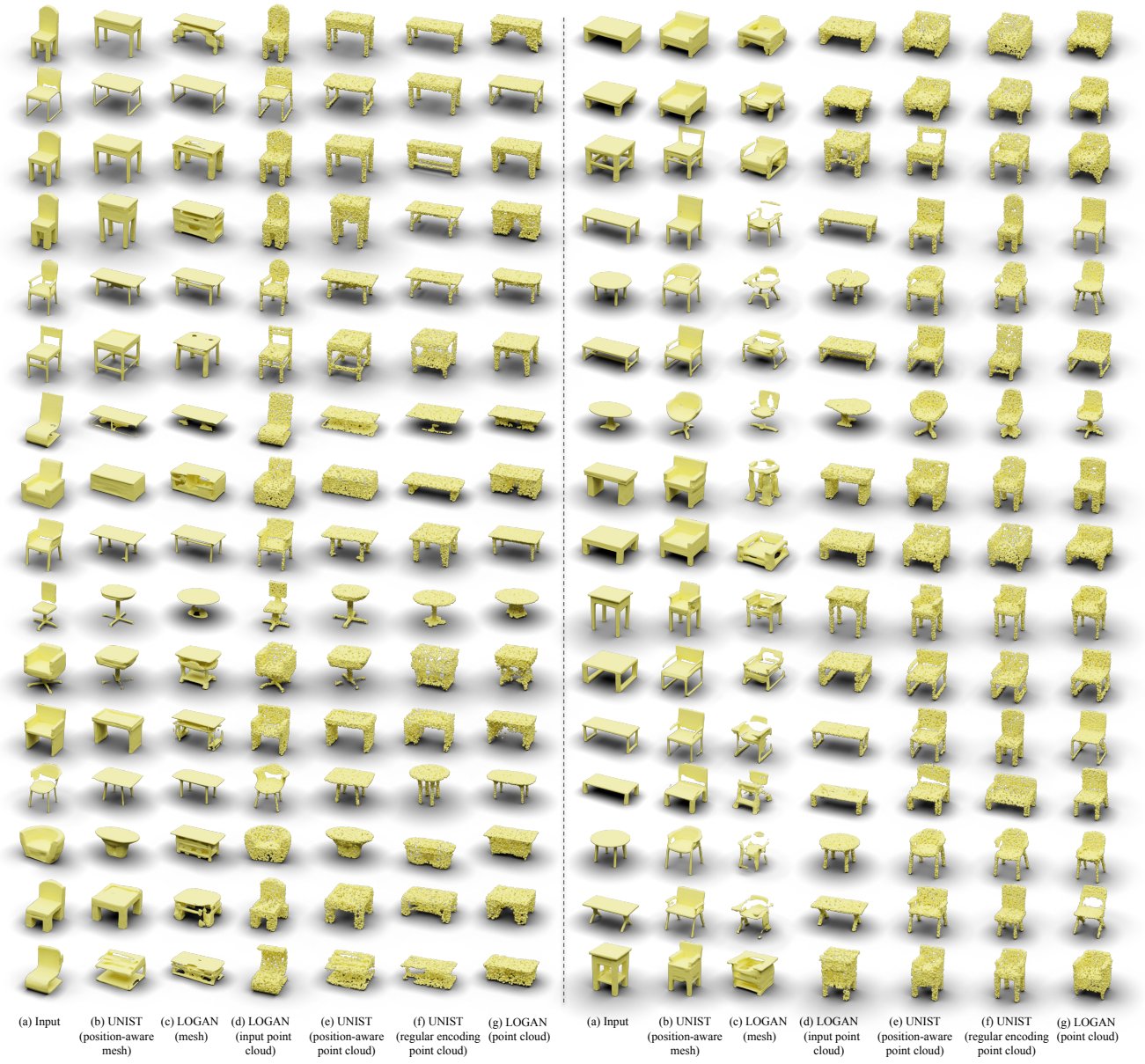

Figure 18. *Randomly* selected qualitative comparison of translation results by different translation networks on (left) $Chair \leftrightarrow Table$ and (right) $Table \leftrightarrow Chair$. (a) Test input, (b) translation of UNIST with position-aware encoding in mesh representation, (c) LOGAN translation in mesh representation, (d) LOGAN input point cloud representation, (e) translation of UNIST with position-aware encoding in point cloud representation, (f) translation of UNIST with regular encoding in point cloud representation and (g) LOGAN translation in point cloud representation.

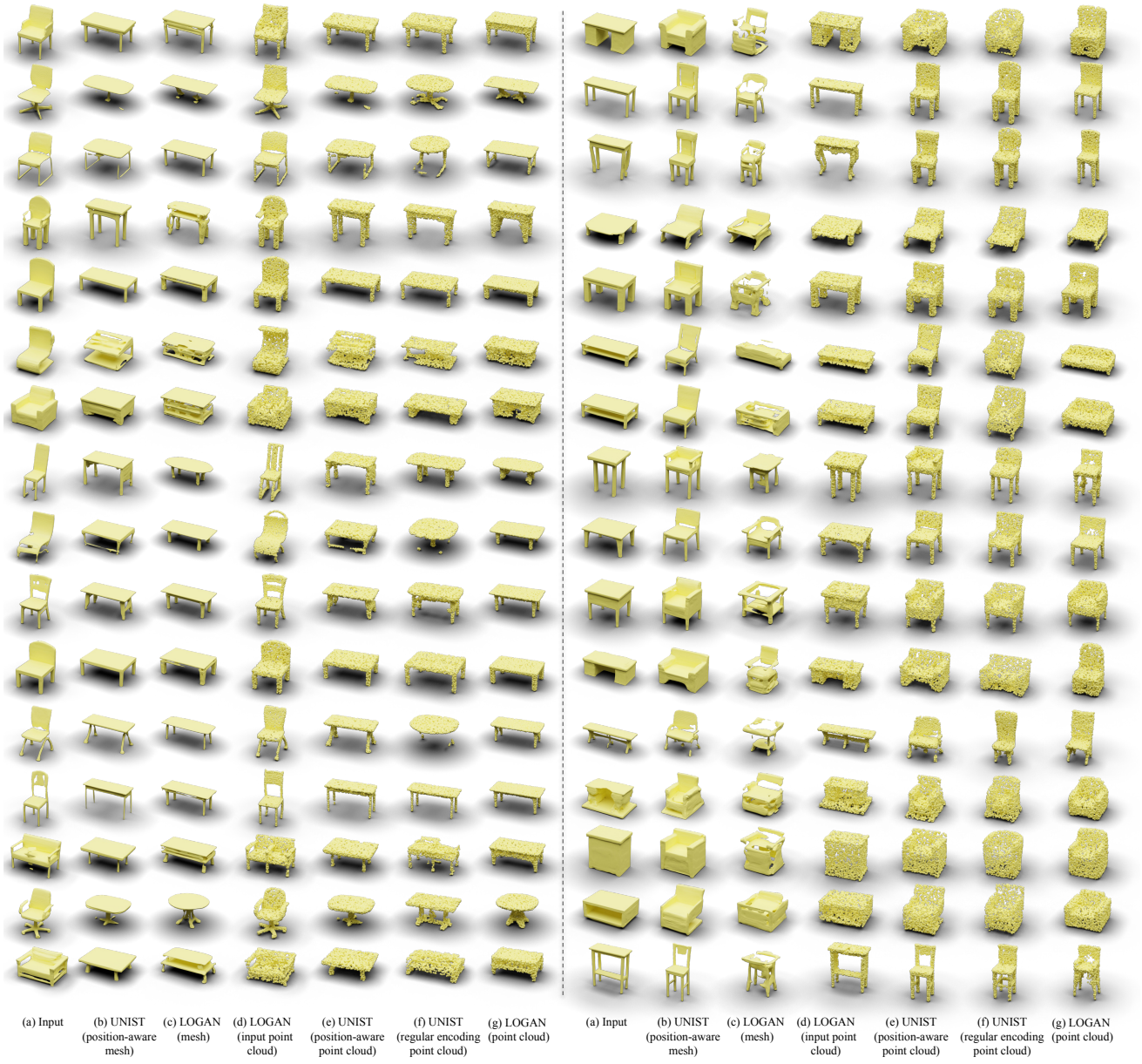

Figure 19. *Randomly* selected qualitative comparison of translation results by different translation networks on (left) $Chair \leftrightarrow Table$ and (right) $Table \leftrightarrow Chair$. (a) Test input, (b) translation of UNIST with position-aware encoding in mesh representation, (c) LOGAN translation in mesh representation, (d) LOGAN input point cloud representation, (e) translation of UNIST with position-aware encoding in point cloud representation, (f) translation of UNIST with regular encoding in point cloud representation and (g) LOGAN translation in point cloud representation.

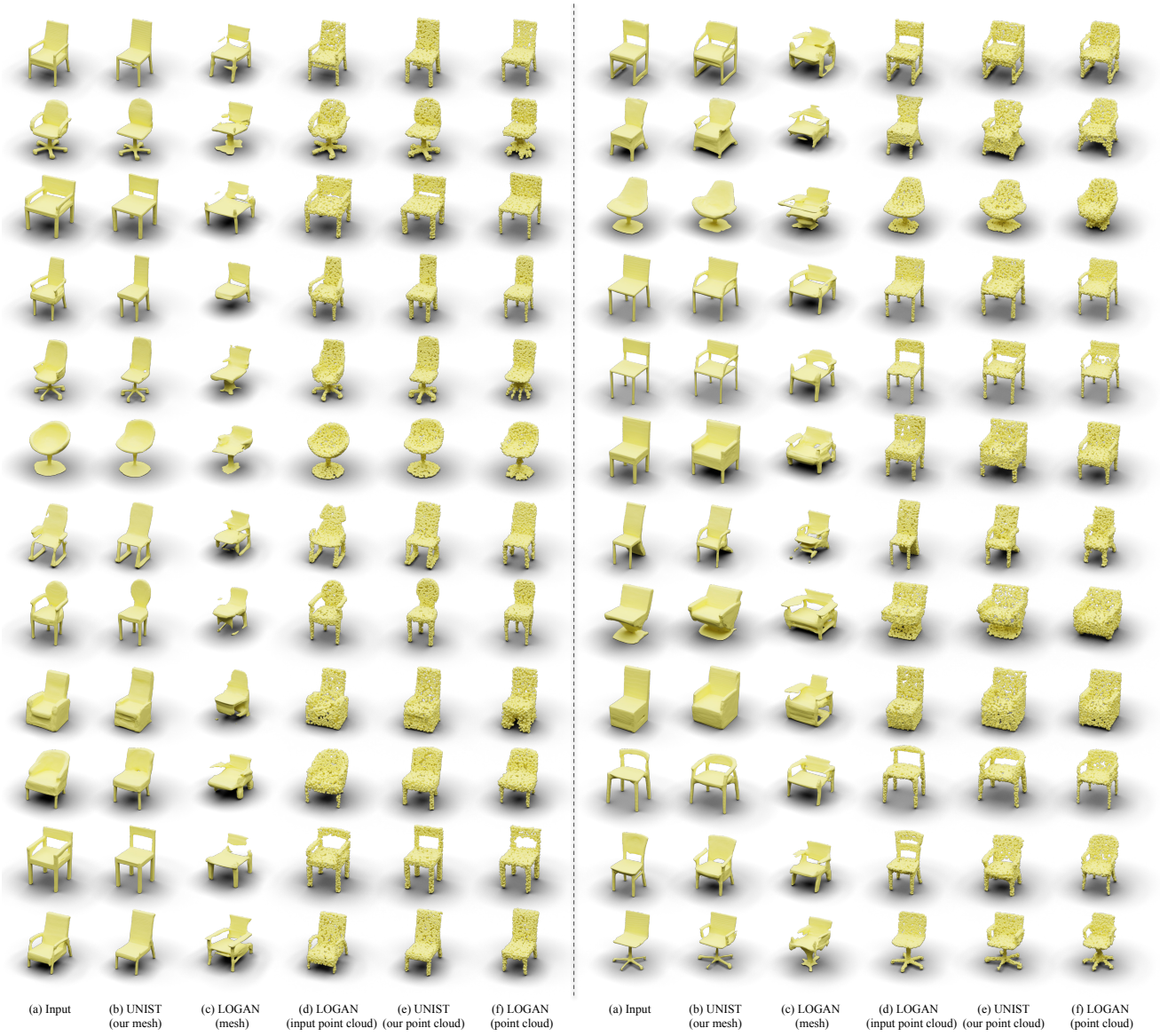

Figure 20. *Randomly* selected qualitative comparison of translation results by different translation networks on (left) chair $w\ Armrest \leftrightarrow w/o\ Armrest$ and (right) chair $w/o\ Armrest \leftrightarrow w\ Armrest$. (a) Test input, (b) UNIST translation in mesh representation, (c) LOGAN translation in mesh representation, (d) LOGAN input point cloud representation, (e) UNIST translation in point cloud representation and (f) LOGAN translation in point cloud representation.

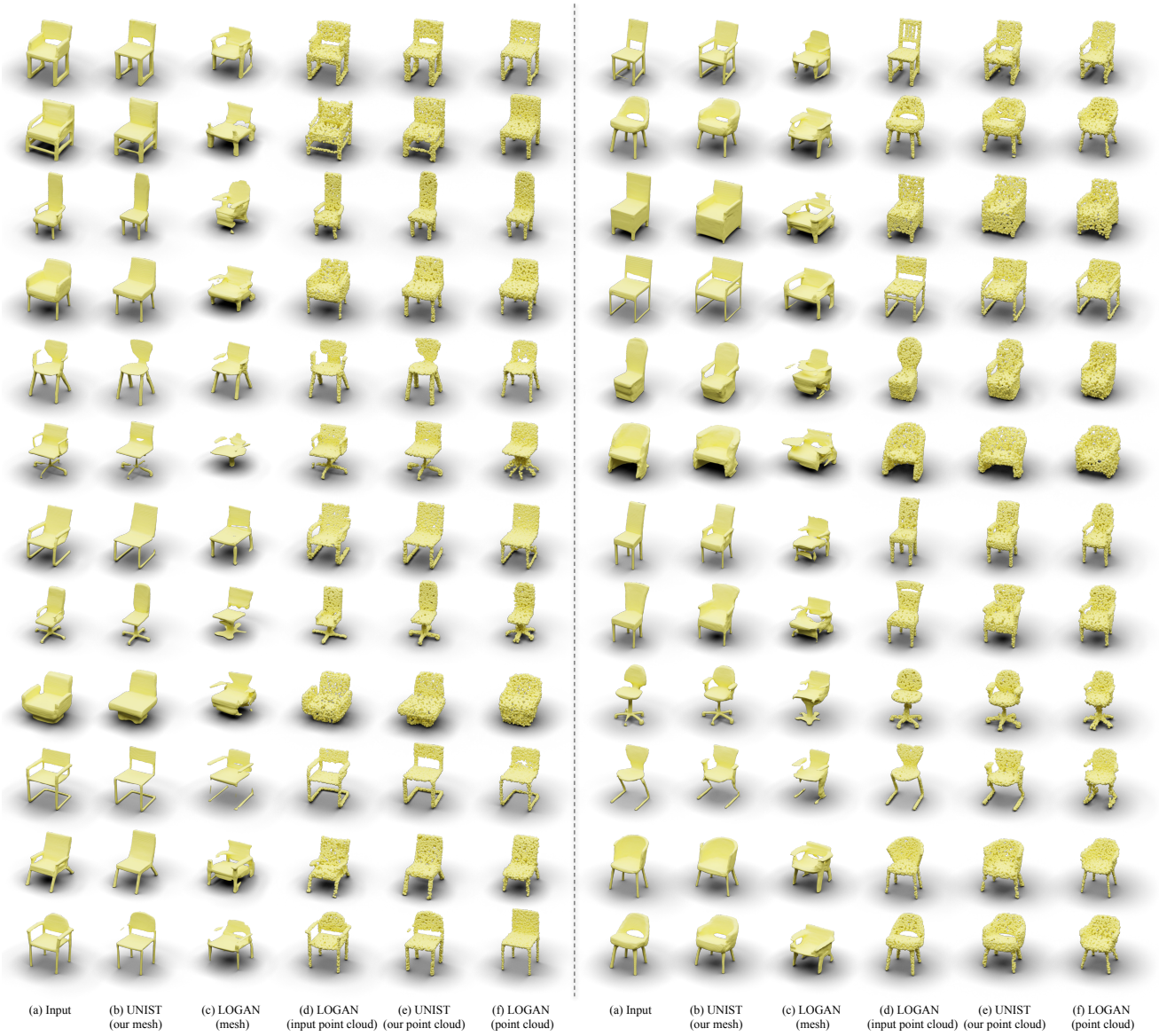

Figure 21. *Randomly* selected qualitative comparison of translation results by different translation networks on (left) chair $w\ Armrest \leftrightarrow w/o\ Armrest$ and (right) chair $w/o\ Armrest \leftrightarrow w\ Armrest$. (a) Test input, (b) UNIST translation in mesh representation, (c) LOGAN translation in mesh representation, (d) LOGAN input point cloud representation, (e) UNIST translation in point cloud representation and (f) LOGAN translation in point cloud representation.

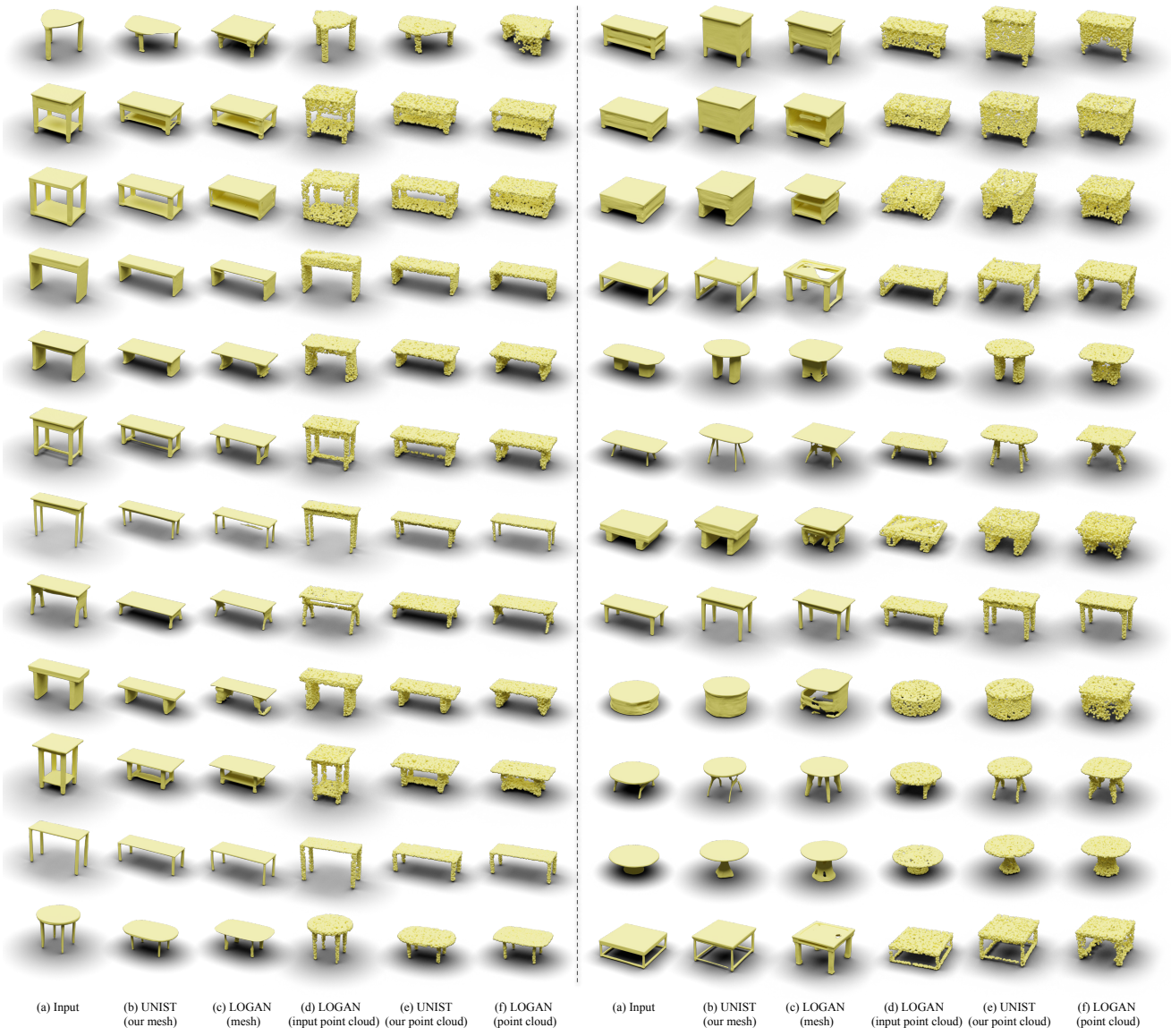

Figure 22. *Randomly* selected qualitative comparison of translation results by different translation networks on (left) $Tall\,table \leftrightarrow Short\,table$ and (right) $Short\,table \leftrightarrow Tall\,table$. (a) Test input, (b) UNIST translation in mesh representation, (c) LOGAN translation in mesh representation, (d) LOGAN input point cloud representation, (e) UNIST translation in point cloud representation and (f) LOGAN translation in point cloud representation.

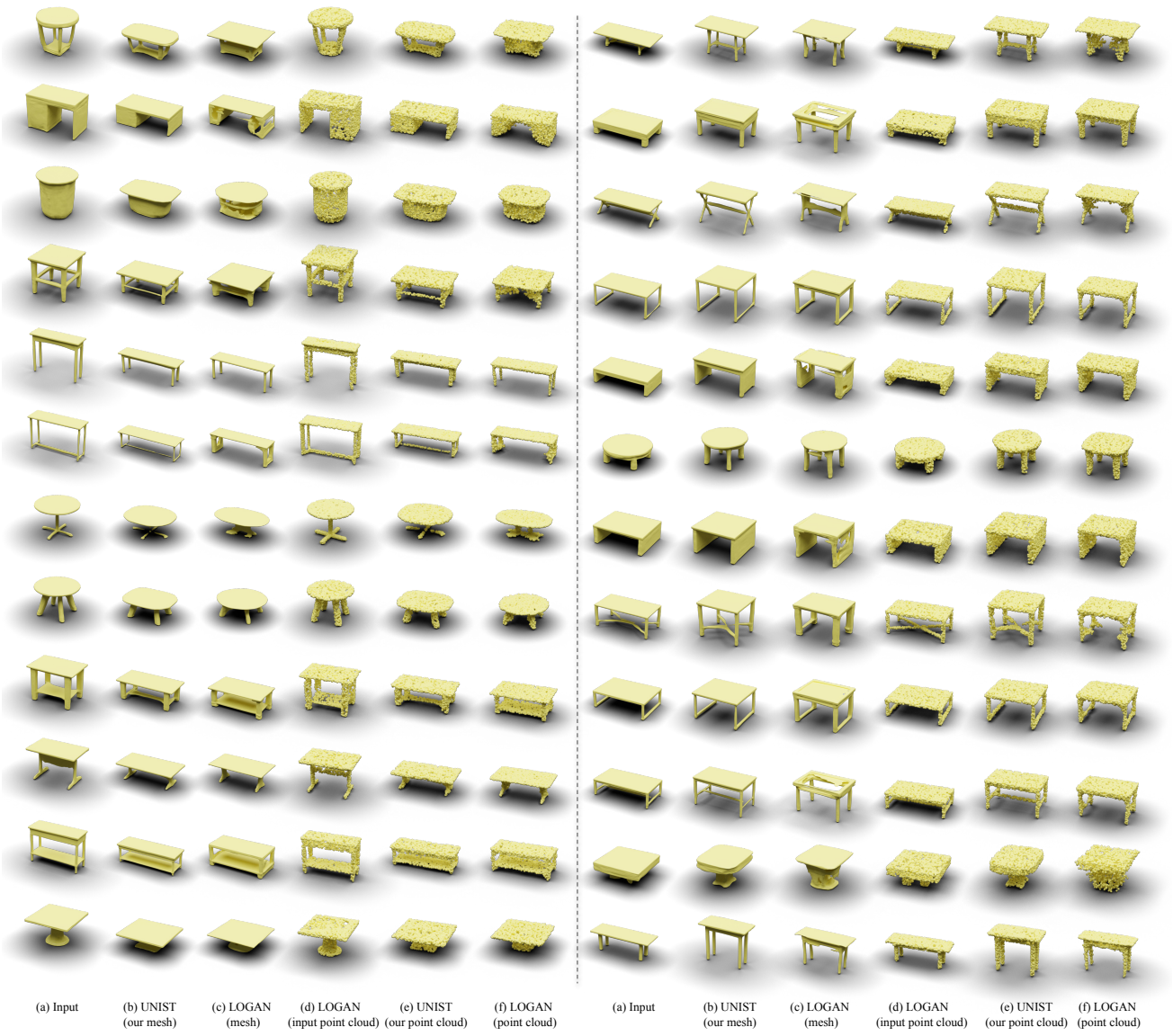

Figure 23. *Randomly* selected qualitative comparison of translation results by different translation networks on (left) $Tall\,table \leftrightarrow Short\,table$ and (right) $Short\,table \leftrightarrow Tall\,table$. (a) Test input, (b) UNIST translation in mesh representation, (c) LOGAN translation in mesh representation, (d) LOGAN input point cloud representation, (e) UNIST translation in point cloud representation and (f) LOGAN translation in point cloud representation.