Supplementary Material: RIVQ-VAE: Rotation-invariant Discrete 3D Representation Learning

Mariem Mezghanni* Malika Boulkenafed Maks Ovsjanikov*

*LIX, Ecole Polytechnique, IP Paris
mezghanni,maks@lix.polytechnique.frr

Figure 1. Similar patch embedding. Each row displays colored patches that are mapped to the same codebook vector.

1. Rotation-Invariant embedding

Figure 1 displays example patches represented by *the same codebook vector*. Observe that similarly-colored patches are geometrically close when discarding rotations/reflections. These examples help to illustrate geometric and rotation patterns that are compactly represented using our canonicalized codebook.

To further demonstrate the rotation-invariance property, we propose to randomly select N different local surface patches from test shapes. For each patch, we generate M rotated point cloud patches by (1) randomly sampling 128 surface points, and (2) applying a random rotation. Then we feed each obtained point cloud patch to both our TFN [3] backbone and the PointNet [2] backbone, in order to compute patch embedding. Figure 2 provides a t-sne [1] visualization of the obtained embeddings using each of the backbones. Observe that TFN [3] backbone manages to recognize clusters of patches associated with the same intrinsic geometry, while the PointNet [2] backbone ignores the up-to-rotation similarity, leading to unstructured embeddings.

2. Codebook generalization power across shapes and categories

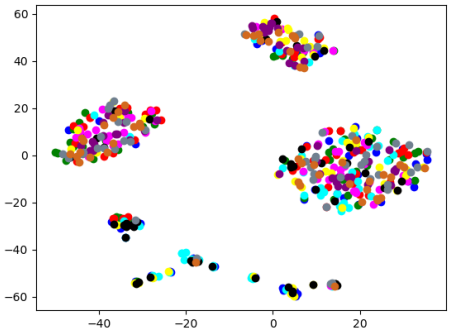
We analyze the contribution of each codebook vector across different shapes and categories. To this end, Figure 3 shows, for each codebook index, the number of test shapes (3a) and the number of categories (3b) to which it contributes.

Figure 3a shows that all codebook vectors are leveraged to represent test shapes. The index associated with the highest occurrence inherently corresponds to empty region that occurs in all shapes. The remaining vectors occur in at least 136 shapes among the 7791 test shapes.

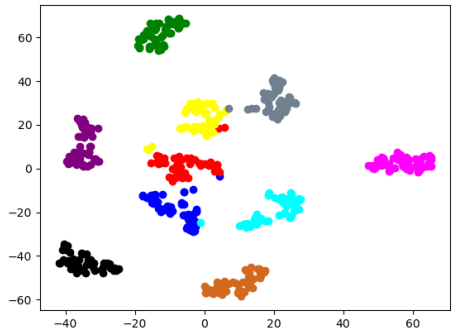
Furthermore, Figure 3b shows that our model not only grasps local similarities across different shapes but also across different categories. Observe that all codebook vectors contribute to at least 10 categories among the learning 13 categories. This particularly proves that our region-based local encoder is category agnostic, allowing to efficiently embed different categories to the same latent space.

3. Impact of codebook size

We re-train ShapeFormer-8 [4] and our method using codebook size $K = 256$. We recall that we set $K = 512$ and

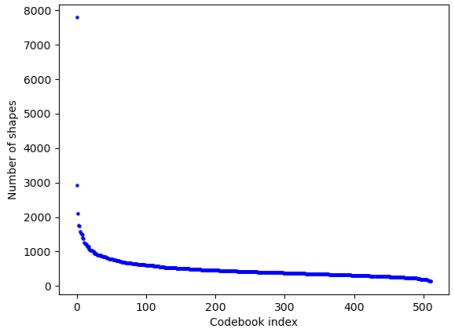


(a) PointNet [2] backbone.

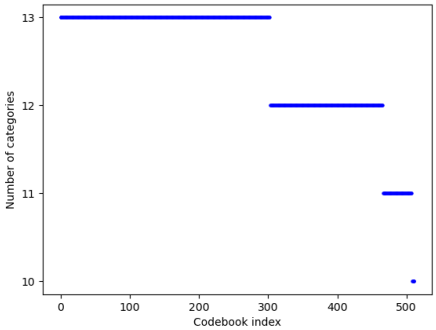


(b) TFN [3] backbone.

Figure 2. t-sne [1] visualization of rotated patches embeddings. We consider N=10 patches, colored each differently, and M=50 random rotations for each patch.



(a) For each codebook vector index, we provide the number of shapes in which it occurs.



(b) For each codebook vector index, we provide the number of categories in which it occurs.

Figure 3. Number of shapes and categories to which contribute each codebook vector index. For clarity, we sort codebook indices according to the coordinate values for each plot.

Method	K=512	K=256
ShapeFormer-8[4]	0.195	0.210
Ours	**0.120**	**0.122**

Table 1. Quantitative results for shape auto-encoding across different codebook sizes K. Our approach achieves better results thanks to the compact embedding. CD is multiplied by 10^2.

$K = 4096$ in Table 1 from the main paper based on baselines settings, for a fair comparison.

Quantitative evaluations in Table 1 show that our approach maintains its superiority in terms of reconstruction quality and accuracy. Besides, in terms of CD for instance, our method entails smaller performance drop of $\downarrow$1.6% than baseline $\downarrow$7.7% when reducing codebook size. This observation particularly supports the compact representation enabled by our model.

4. Additional comments on shape auto-encoding experiments in Section 4.1 from the main paper

Ablation study interpretation. Even when removing the rotation invariance, ours-PN outperforms ShapeFormer-8 [4]. We recall that both methods encode local regions using PointNet [2]-like backbones, and predict deep implicit functions. We hence attribute the superiority of our approach to the design of our decoder architecture that, in contrast to SF-8 decoder, is tailored to promote the sharing of knowledge across the different local representations. While this decoder design is important to recover the local orientations and produce smooth and high-quality surfaces, it however inherits limitations of the global-based design, in particular, the limited generalization power when producing shapes that belong to unseen categories or that undergo a global rotation (unaligned).

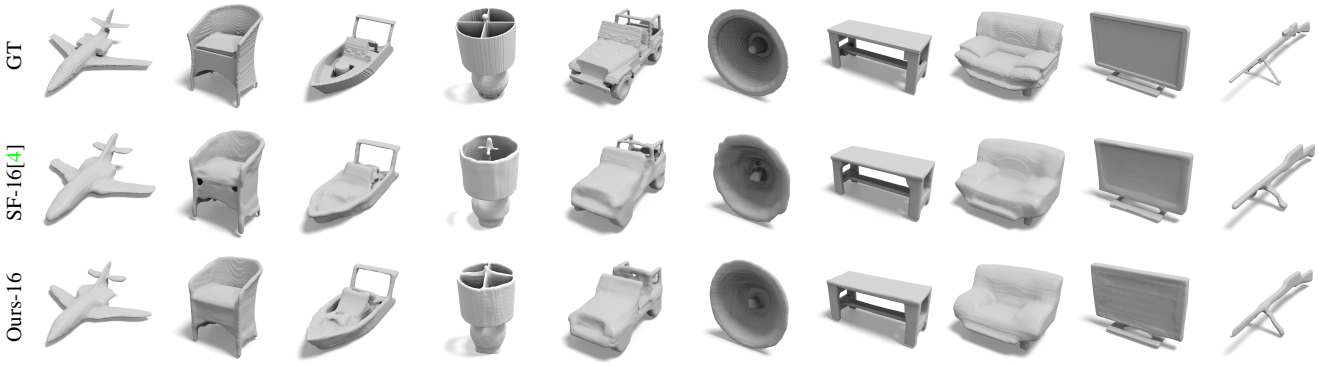


Figure 4. Qualitative results for shape auto-encoding. Our method allows higher detailed reconstructions.

Additional qualitative results. In this section, we collect additional visual results that were not included in the main manuscript.

Figure 4 compares the shape reconstruction quality of baseline ShapeFormer-16 [4] with RIVQ-VAE-16 while emphasizing the superiority of our method.

Training settings. Our model consists of 30M parameters and is trained for about 5 days using 3 Quadro RTX 8000.

References

[1] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9: 2579–2605, 2008. 1, 2

[2] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017. 1, 2

[3] Nathaniel Thomas, Tess E. Smidt, Steven Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. Tensor field networks: Rotation- and translation-equivariant neural networks for 3d point clouds. *CoRR*, abs/1802.08219, 2018. 1, 2

[4] Xingguang Yan, Liqiang Lin, Niloy J. Mitra, Dani Lischinski, Danny Cohen-Or, and Hui Huang. Shapeformer: Transformer-based shape completion via sparse representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 1, 2, 3