

EscherNet: A Generative Model for Scalable View Synthesis (Appendix)

Acknowledgement

This research is funded by EPSRC Prosperity Partnerships (EP/S036636/1) and Dyson Technology Ltd. Xin Kong holds a China Scholarship Council-Imperial Scholarship. We would like to thank Sayak Paul and HuggingFace for contributing the training compute that facilitated early project exploration. We would also like to acknowledge Yifei Ren for his valuable discussions on formulating the 6DoF CaPE.

A. Python Implementation of CaPE

```
def compute_4dof_cape(v, P, s):
    """
    :param v: input feature vector with its dimension must be divisible by 8
    :param P: list = [alpha, beta, gamma, r]
    :param s: a small scalar for radius
    :return: rotated v with its corresponding camera pose P
    """
    v = v.reshape([-1, 8])
    psi = np.zeros([8, 8])
    for i in range(4):
        if i < 3:
            psi[2 * i:2 * (i + 1), 2 * i:2 * (i + 1)] = \
                np.array([[np.cos(P[i]), -np.sin(P[i])], [np.sin(P[i]), np.cos(P[i])]])
        else:
            psi[2 * i:2 * (i + 1), 2 * i:2 * (i + 1)] = \
                np.array([[np.cos(s * np.log(P[i])), -np.sin(s * np.log(P[i]))],
                        [np.sin(s * np.log(P[i])), np.cos(s * np.log(P[i]))]])

    return v.dot(psi).reshape(-1)
```

Listing 1. Python implementation for 4 DoF CaPE.

```
def compute_6dof_cape(v, P, s=0.001, key=True):
    """
    :param v: input feature vector with its dimension must be divisible by 4
    :param P: 4 x 4 SE3 matrix
    :param s: a small scalar for translation
    :return: rotated v with its corresponding camera pose P
    """
    v = v.reshape([-1, 4])
    P[:3, 3] *= s
    psi = P if key else np.linalg.inv(P).T
    return v.dot(psi).reshape(-1)
```

Listing 2. Python implementation for 6 DoF CaPE.

B. Additional Training Details and Experimental Settings

Optimisation and Implementation EscherNet is trained using the AdamW optimiser [24] with a learning rate of $1 \cdot 10^{-4}$ and weight decay of 0.01 for $[256 \times 256]$ resolution images. We incorporate cosine annealing, reducing the learning rate to $1 \cdot 10^{-5}$ over a total of 100,000 training steps, while linearly warming up for the initial 1000 steps. To speed up training, we implement automatic mixed precision with a precision of `bfloat16` and employ gradient checkpointing. Our training batches consist of 3 reference views and 3 target views randomly sampled with replacement from 12 views for each object, with a total batch size of 672 (112 batches per GPU). The entire model training process takes 1 week on 6 NVIDIA A100 GPUs.

Metrics For 2D metrics used in view synthesis, we employ PSNR, SSIM [53], LPIPS [61]. For 3D metrics used in 3D generation, we employ Chamfer Distance and Volume IoU. To ensure a fair and efficient evaluation process, each baseline method and our approach are executed only once per scene per viewpoint. This practice has proven to provide stable averaged results across multiple scenes and viewpoints.

B.1. Evaluation Details

In NeRF Synthetic Dataset [29], we consider and evaluate all 8 scenes provided in the original dataset. To assess performance with varying numbers of reference views, we train all baseline methods and our approach using the same set of views randomly sampled from the training set. The evaluation is conducted on all target views defined in the test sets across all 8 scenes (with 200 views per scene). For InstantNGP [31], we run 10k steps (≈ 1 min) for each scene. For 3D Gaussian Splatting [18], we run 5k steps (≈ 2 min) for each scene.

In Google Scanned Dataset (GSO) [10], we evaluate the same 30 objects chosen by SyncDreamer [22]. For each object, we render 25 views with randomly generated camera poses and a randomly generated environment lighting condition to construct our test set. For each object, we choose the first 10 images as our reference views and the subsequent 15 images as our target views for evaluation. It’s crucial to note that all reference and target views are rendered with random camera poses, establishing a more realistic and challenging evaluation setting compared to the evaluation setups employed in other baselines: *e.g.* SyncDreamer uses an evenly distributed environment lighting to render all GSO data, and the reference view for each object is manually selected based on human preference.¹ Additionally, the evaluated target view is also manually selected based on human preference chosen among four independent generations.²

In evaluating 3D generation, we randomly sample 4096 points evenly distributed from the generated 3D mesh or point cloud across all methods. Each method’s generated mesh is aligned to the ground-truth mesh using the camera pose of the reference views. Specifically in Point-E [32] and Shape-E [17], we rotate 90/180 degrees along each x/y/z axis to determine the optimal alignment for the final mesh pose. Our evaluation approach again differs from SyncDreamer, which initially projects the 3D mesh into their fixed 16 generated views to obtain depth maps. Then, points are sampled from these depth maps for the final evaluation.³

In RTMV Dataset [50], we follow the evaluation setting used in Zero-1-to-3 [21], which consists of 10 complex scenes featuring a pile of multiple objects from the GSO dataset. Similar to the construction of our GSO test set, we then randomly select a fixed subset of the first 10 images as our reference views and the subsequent 10 views as our target views for evaluation.

¹<https://github.com/liuyuan-pal/SyncDreamer/issues/21>

²<https://github.com/liuyuan-pal/SyncDreamer/issues/21#issuecomment-1770345260>

³<https://github.com/liuyuan-pal/SyncDreamer/issues/44>

C. Additional Results on 6 DoF CaPE

To validate the effectiveness of the 6 DoF CaPE design, we demonstrate its performance in novel view synthesis on GSO and RTMV datasets in Tab. 4a and on the NeRF Synthetic dataset in Tab. 4c. We also provide 3D reconstruction results on GSO dataset in Tab. 4b. It is evident that EscherNet with 6 DoF CaPE achieves comparable, and often, slightly improved results when compared to our 4 DoF CaPE design.

	Training Data	# Ref. Views	GSO-30			RTMV		
			PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
RealFusion	-	1	12.76	0.758	0.382	-	-	-
Zero123	800K	1	18.51	0.856	0.127	10.16	0.505	0.418
Zero123-XL	10M	1	18.93	0.856	0.124	10.59	0.520	0.401
EscherNet - 4 DoF	800k	1	20.24	0.884	0.095	10.56	0.518	0.410
EscherNet - 4 DoF	800k	2	22.91	0.908	0.064	12.66	0.585	0.301
EscherNet - 4 DoF	800k	3	24.09	0.918	0.052	13.59	0.611	0.258
EscherNet - 4 DoF	800k	5	25.09	0.927	0.043	14.52	0.633	0.222
EscherNet - 4 DoF	800k	10	25.90	0.935	0.036	15.55	0.657	0.185
EscherNet - 6 DoF	800k	1	20.89	0.886	0.093	12.30	0.569	0.332
EscherNet - 6 DoF	800k	2	23.92	0.917	0.057	14.18	0.618	0.252
EscherNet - 6 DoF	800k	3	25.21	0.927	0.045	15.06	0.643	0.217
EscherNet - 6 DoF	800k	5	26.59	0.937	0.036	15.71	0.663	0.190
EscherNet - 6 DoF	800k	10	27.75	0.947	0.030	16.58	0.688	0.160

	# Ref. Views	Chamfer Dist. \downarrow	Volume IoU \uparrow
Point-E	1	0.0447	0.2503
Shape-E	1	0.0448	0.3762
One2345	1	0.0632	0.4209
One2345-XL	1	0.0667	0.4016
DreamGaussian	1	0.0605	0.3757
DreamGaussian-XL	1	0.0459	0.4531
SyncDreamer	1	0.0400	0.5220
NeuS	3	0.0366	0.5352
NeuS	5	0.0245	0.6742
NeuS	10	0.0195	0.7264
EscherNet - 4 DoF	1	0.0314	0.5974
EscherNet - 4 DoF	2	0.0215	0.6868
EscherNet - 4 DoF	3	0.0190	0.7189
EscherNet - 4 DoF	5	0.0175	0.7423
EscherNet - 4 DoF	10	0.0167	0.7478
EscherNet - 6 DoF	1	0.0274	0.6382
EscherNet - 6 DoF	2	0.0196	0.7100
EscherNet - 6 DoF	3	0.0180	0.7348
EscherNet - 6 DoF	5	0.0176	0.7392
EscherNet - 6 DoF	10	0.0160	0.7628

(a) Novel view synthesis performance on GSO and RTMV datasets.

(b) 3D reconstruction performance on GSO.

	# Reference Views (Less \rightarrow More)							
	1	2	3	5	10	20	50	100
InstantNGP (Scene Specific Training)								
PSNR \uparrow	10.92	12.42	14.27	18.17	22.96	24.99	26.86	27.30
SSIM \uparrow	0.449	0.521	0.618	0.761	0.881	0.917	0.946	0.953
LPIPS \downarrow	0.627	0.499	0.391	0.228	0.091	0.058	0.034	0.031
GaussianSplatting (Scene Specific Training)								
PSNR \uparrow	9.44	10.78	12.87	17.09	23.04	25.34	26.98	27.11
SSIM \uparrow	0.391	0.432	0.546	0.732	0.876	0.919	0.942	0.944
LPIPS \downarrow	0.610	0.541	0.441	0.243	0.085	0.054	0.041	0.041
EscherNet - 4 DoF (Zero Shot Inference)								
PSNR \uparrow	13.36	14.95	16.19	17.16	17.74	17.91	18.05	18.15
SSIM \uparrow	0.659	0.700	0.729	0.748	0.761	0.765	0.769	0.771
LPIPS \downarrow	0.291	0.208	0.161	0.127	0.114	0.106	0.099	0.097
EscherNet - 6 DoF (Zero Shot Inference)								
PSNR \uparrow	13.73	15.66	16.91	17.72	18.47	18.77	19.24	19.28
SSIM \uparrow	0.664	0.712	0.745	0.762	0.779	0.786	0.795	0.796
LPIPS \downarrow	0.294	0.197	0.149	0.120	0.103	0.095	0.085	0.084

(c) Novel view synthesis performance on NeRF Synthetic dataset.

Table 4. EscherNet 6 DoF presents a similar and sometimes improved performance than EscherNet 4 DoF.

D. Additional Results on NeRF Synthetic Dataset

We present additional visualisation on the NeRF Synthetic Dataset using EscherNet trained with 4 DoF CaPE.





1	2	3	# Reference Views (Less → More)		20	50	100
InstantNGP (Scene Specific Training)							
							
PSNR 9.45	PSNR 11.41	PSNR 13.64	PSNR 19.30	PSNR 23.14	PSNR 26.18	PSNR 28.54	PSNR 28.87
							
PSNR 10.37	PSNR 11.72	PSNR 12.82	PSNR 15.58	PSNR 19.71	PSNR 21.28	PSNR 23.09	PSNR 23.78
3D Gaussian Splatting (Scene Specific Training)							
							
PSNR 8.07	PSNR 9.16	PSNR 11.72	PSNR 17.32	PSNR 24.19	PSNR 25.34	PSNR 26.98	PSNR 29.01
							
PSNR 9.14	PSNR 10.63	PSNR 11.43	PSNR 14.81	PSNR 20.15	PSNR 22.88	PSNR 23.49	PSNR 23.51
EscherNet (Zero Shot Inference)							
							
PSNR 10.86	PSNR 10.80	PSNR 15.51	PSNR 17.07	PSNR 17.40	PSNR 17.38	PSNR 17.77	PSNR 17.85
							
PSNR 10.10	PSNR 13.25	PSNR 13.43	PSNR 14.33	PSNR 14.97	PSNR 15.65	PSNR 15.70	PSNR 15.90

Table 5. Novel View Synthesis on NeRF Synthetic Dataset. We report the average PSNR per scene, conditioned on the respective number of reference views.

E. Additional Results on Text-to-3D

We present additional visualisation on text-to-image-to-3D using EscherNet trained with 4 DoF CaPE.




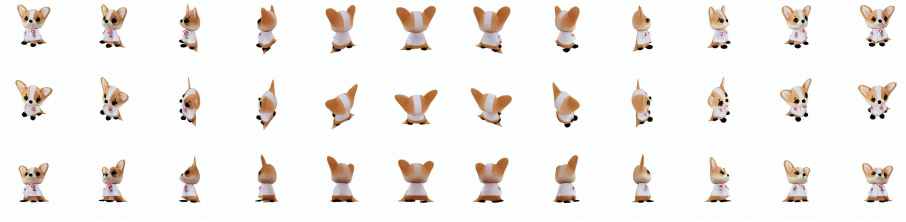

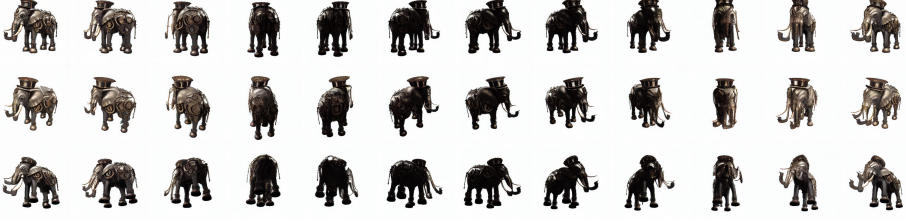






<p>A robot made of vegetables.</p> 	
<p>A nurse corgi.</p> 	
<p>A cute steampunk elephant.</p> 	
<p>A bull dog wearing a black pirate hat.</p> 	
<p>An astronaut riding a horse.</p> 	
<p>Medieval House, grass, medieval, medieval-decor, 3d asset.</p> 	

Table 6. Text-to-3D generation with SDXL (top 3) and MVDream (bottom 3).

F. Additional Discussions, Limitations and Future Work

Direct v.s. Autoregressive Generation EscherNet’s flexibility in handling arbitrary numbers of reference and target views offers multiple choices for view synthesis. In our experiments, we employ the straightforward direct generation to jointly generate all target views. Additionally, an alternative approach is autoregressive generation, where target views are generated sequentially, similar to text generation with autoregressive language models.

For generating a large number of target views, autoregressive generation can be significantly faster than direct generation (e.g. more than $20\times$ faster for generating 200 views). This efficiency gain arises from converting a quadratic inference cost into a linear inference cost in each self-attention block. However, it’s important to note that autoregressive generation may encounter a *content drifting problem* in our current design, where the generated quality gradually decreases as each newly generated view depends on previously non-perfect generated views. Autoregressive generation boasts many advantages in terms of inference efficiency and is well-suited for specific scenarios like SLAM (Simultaneous Localization and Mapping). As such, enhancing rendering quality in such a setting represents an essential avenue for future research.

Stochasticity and Consistency in Multi-View Generation We also observe that to enhance the target view synthesis quality, especially when conditioning on a limited number of reference views, introducing additional target views can be highly beneficial. These supplementary target views can either be randomly defined or duplicates with the identical target camera poses. Simultaneously generating multiple target views serves to implicitly reduce the inherent stochasticity in the diffusion process, resulting in improved generation quality and consistency. Through empirical investigations, we determine that the optimal configuration ensures a minimum of 15 target views, as highlighted in orange in Fig. 8. Beyond this threshold, any additional views yield marginal performance improvements.

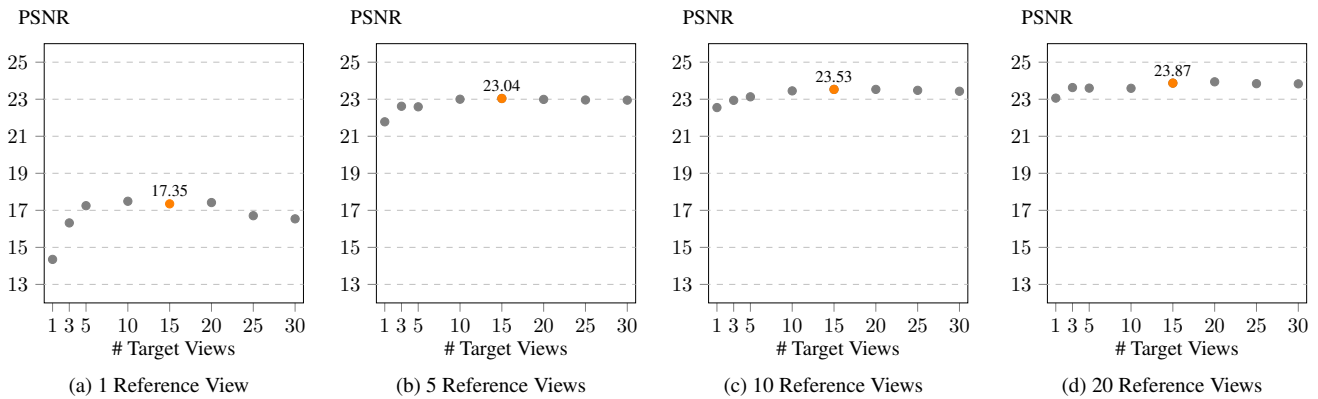


Figure 8. **Novel view synthesis with a different number of reference and target views.** We present the averaged performance of EscherNet on *one* pre-selected target view across objects in the GSO dataset. We observe a clear improvement in view synthesis quality as the number of both reference and target views increases. In this scenario, the multiple target views are essentially multiple duplicates of the initially chosen single pre-selected view, a strategy we find effective in enhancing view synthesis quality.

Training Data Sampling Strategy We have explored various combinations of $N \in \{1, 2, 3, 4, 5\}$ reference views and $M \in \{1, 2, 3, 4, 5\}$ target views during EscherNet training. Empirically, a larger number of views demand more GPU memory and slow down training speed, while a smaller number of views may restrict the model’s ability to learn multi-view correspondences. To balance training efficiency and performance, we set our training views to $N = 3$ reference views and $M = 3$ target views for each object, a configuration that has proven effective in practice. Additionally, we adopt a random sampling approach with replacement for these 6 views, introducing the possibility of repeated images in the training views. This sampling strategy has demonstrated a slight improvement in performance compared to sampling without replacement.

Scaling with Multi-view Video EscherNet’s flexibility sets it apart from other multi-view diffusion models [22, 23] that require a set of fixed-view rendered images from 3D datasets for training. EscherNet can efficiently construct training samples using just a pair of posed images. While it can benefit from large-scale 3D datasets like [6, 7], EscherNet’s adaptability extends to a broader range of posed image sources, including those directly derived from videos. Scaling EscherNet to accommodate multiple data sources is an important direction for future research.