APPENDIX A: MORE DETAILS ON MODULATED MLP LAYER

The modulated MLP layer used in our paper is similar to the style modulation in Karras et al. (2020). Mathematically, we denote the weights of a MLP layer (1x1 convolution) by $W \in \mathbb{R}^{d_{in} \times d_{out}}$, where d_{in}, d_{out}, and w_{ij} denote the dimension of input and output, the element in i^{th} row and j^{th} column of W, respectively. We obtain the style vector $\mathbf{s} \in \mathbb{R}^{d_{in}}$ by passing the latent variable $\mathbf{z}$ into two MLP layers. The i^{th} element of style vector s_i is thus used to modulate the parameter of W as,

$$w'_{ij} = s_i \cdot w_{ij}, \quad j = 1, \cdots, d_{out}, \tag{1}$$

where w_{ij} and w'_{ij} denote the original and modulated weights, respectively.

The modulated weights are normalized to preserve training stability,

$$w''_{ij} = w'_{ij} \Big/ \sqrt{\sum_i w'^2_{ij} + \epsilon}, \quad j = 1, \cdots, d_{out}, \tag{2}$$

where ϵ is a very small constant to prevent the denominator to be zero. These two equations describe the mechanism of the modulate MLP used in our method.

APPENDIX B: DETAILED EXPERIMENTAL SETTINGS

Here, we provide the detailed hyper parameters used in all the experiments in Table 1. Among them, some hyper parameters denote to, L_s number of self-attention layers in bottleneck encoder, L_c number of cross-attention layers, K number of hierarchical blocks, d the feature dimension through the entire network. The row of Fourier features describes whether we embed the coordinate into high-frequency embeddings as suggested in (Tancik et al., 2020). β is the hyper parameter used to balance the losses of reconstruction and KL divergence. The reconstruction term describes how we calculate the reconstruction loss $\mathbb{E}_{z \sim q_\phi(z|D_T)}[-\log p_\theta(Y_T|z, X_T, D_C)]$.

	1D synthetic functions	2D images	3D scenes
L_s	3	6	6
L_c	1	4	2
L_K	6	6	4
d	128	512	512
d_z	16	64	64
model size	2.3M	56.7M	34.3M
Fourier features	✗	✓	✓
batch size	100	16	32
iteration number	0.1 million	0.5 million	0.1 million
lr	5e-4	1e-4	1e-4
β	1	0.1	0.001
reconstruction term	Gaussian	discrete logistic mixture (Salimans et al.)	MSE
Training resources	1x V100 16GB	4x V100 16GB (CelebA178)	4x V100 32GB

Table 1: Hyper parameters in our experiments.

In addition, when training for 1D synthetic functions, we will randomly sample 5 to 15 context points and 15 to 25 target points on Matern and RBF kernel. For the last four rows listed in the table 1 of our main paper, we use importance weighted sampling to calculate the approximated log likelihood. We use this metric because these four models are all latent variable models, where the results of importance weighted sampling would be more accurate, especially for hierarchical latent variable models such as NVAE (Vahdat & Kautz, 2020).

Training a VNP model for the 1D regression task requires about 5 hours. Training a 2D VNP model on CelebA64 requires about 24 hours. Training a 3D VNP model for novel view synthesis takes

around 40 hours. For the inference speed, VNP requires very short time for 1D and 2D tasks. On 1D regression task, VNP takes 0.285 second for testing a batch with batch size of 2000. On the CelebA64 dataset (2D task), our VNP takes around 0.112 second to infer a single batch of size 8. On the 3D Cars dataset, our VNP takes 5.28 second to render an image with a novel view (reasons explained in Appendix C). All these results are tested with a single V100 GPU.

APPENDIX C: COMPARISONS WITH THE OPTIMIZATION-BASED METHOD

In this section, we provide the comparison of our VNP and the previous optimization-based method. We take a representative optimization-based method, SIREN (Sitzmann et al., 2020) for comparison. We conduct experiments on the 3D Car dataset Chang et al. (2015), and measure both the inference speed and the PSNR of the novel synthesized view. Note that since SIREN requires to optimize the network to fit for a specific signal, the iteration time of SIREN is a part of the inference time. We test the inference speed with a single V100 GPU for both schemes.

The results are shown in Table 2. It is observed that for the task of novel view synthesis conditioned on a single view, our proposed VNP provides much better prediction performance compared with the optimized-based method SIREN, even if SIREN is optimized for many iterations for a given test image. One reason is that VNP can learn the dataset prior, which complements many useful information to predict a specific 3D signal. However, if we optimize the SIREN network to fit the known single view, the network will be initially optimized for the right direction, but will then tend to overfit to this single view after many iterations. As a result, it is observed that when the SIREN network gets better prediction performance as the iteration number increased from 1 to 100. However, the performance decreases if we apply more iterations (such as 300 iterations), compared with that of 100 iterations.

As for the inference speed, we can see that our VNP takes 5.28s to render an image in the novel view with a single forward pass. The reason for such a long inference time is that we use a large number of sampling points in 3D space. If an image is with the resolution of 128×128, to render the RGB value of every pixel, there are $128 \times 128 \times p$ target points, where p is the number of sampled point along each ray (set as $p = 128$ in our experiment). Therefore, processing all these target points with cross-attention requires huge GPU memory. We have to divide the image into several pixel groups, and render these pixel groups sequentially. Currently, we have not implement any optimization of our code. We will improve the implementation of method on tasks involving 3D signals in the future.

Note our VNP only presents slow inference speed on this 3D novel view synthesis task. On the 1D and 2D tasks, since there are not so many target points and the model does not incorporate rendering process, the inference speed of our method is fast.

SIREN						VNP (4 blocks)
Iteration Number	1	10	30	100	300	no finetuning
Time (s)	0.21	1.35	3.68	11.59	33.68	4.79
PSNR	11.92	12.27	12.72	12.73	12.00	24.21

Table 2: Novel view synthesis conditioned on a single view. We evaluate the inference time and the prediction performance (PSNR) on ShapeNet Cars (Chang et al., 2015).

APPENDIX D: LIMITATION

The proposed VNP inherits the limitation of NP family. NPs are interesting techniques for meta-learning implicit neural representations due to their reduction of the high cost of training. NPs learns the common knowledge shared by the dataset, enabling fast inference of an unseen signal without the need of finetuning. NPs still cannot work well for dataset including diverse objects (with less shared knowledge), e.g., ImageNet.

REFERENCES

Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8110–8119, 2020.

Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P. Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. In *5th International Conference on Learning Representations, ICLR 2017,*.

Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. volume 33, pp. 7462–7473, 2020.

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. volume 33, pp. 7537–7547, 2020.

Arash Vahdat and Jan Kautz. Nvae: A deep hierarchical variational autoencoder. volume 33, pp. 19667–19679, 2020.