SEMANTIC FRAMEWORK FOR QUERY SYNTHESISED 3D SCENE RENDERING

Sri Gayathri Devi I, Sowmiya Sree S, Jerrick Gerald, Geetha Palanisamy

College of Engineering, Anna University, Chennai, India

ABSTRACT

View synthesis allows the generation of new views of a scene given one or more images. Current methods rely on multiple input images, which are practically not feasible for such applications. While using a single image to create a 3D scene is difficult since it necessitates a solid understanding of 3D settings. To facilitate this, a complete scene understanding of a single-view image is performed using spatial feature extraction and depth map prediction. This work proposes a novel end-to-end model trained on real images without any ground-truth 3D information. The learned 3D features are exploited to render the 3D view. Further, on querying, the target view is generated using the Query network. The refinement network decodes the projected features to in-paint missing regions and generates a realistic output image. The model was trained on two datasets, namely RealEstate10K and KITTI, containing an indoor and outdoor scene.

KEYWORDS

3D Scene Rendering, Differentiable Renderer, Scene Understanding, Quantized Variational AutoEncoder

1. INTRODUCTION

View Synthesis is the process of synthesizing a target image with an arbitrary target camera pose when the inputs are source images and their camera postures. When it comes to how we interpret a visual scene, there is more to it than meets the eye: our brains draw on prior knowledge to reason and make inferences that transcend the patterns of light that pierce our retinas. These visual and cognitive tasks appear simple to humans but are challenging for artificial systems. The task is challenging, as view synthesis requires cognitive scene understanding. In particular, knowledge of the original image's semantics and 3D structure is necessary for successful view synthesis. To record the relative mobility of items that are visible under a view transform, modeling 3D structures is crucial. For modeling view changes to capture perspective changes accurately, 3D awareness is necessary. In order to give machines the capability of cognitive reasoning, **Semantic Framework For Query Synthesised 3D Scene Rendering (SQ3D-Net)** is proposed for generating a scene from a given query of new viewpoints. It requires a comprehensive understanding of 3D scenes from images. The task of synthesizing immersive scenes is presented. The technique outpaints major view changes in a 3D-consistent approach, enabling scene synthesis. It combines 3D reasoning with auto-regressive modeling.

Multi-view input is used because modeling 3D structure is crucial for capturing the motion of items that are visible under a view transform. In order to overcome such difficulties, we work on view synthesis methods in which 3D information is not utilized during training. Instead, an entire generative model with intermediate representations that are 3D-aware can be trained solely using image supervision.

David C. Wyld et al. (Eds): ICDIPV, CBIoT, ICAIT, WIMO, NC, CRYPIS, ITCSE, NLCA, CAIML -2023
pp. 29-39, 2023. CS & IT - CSCP 2023 DOI: 10.5121/csit.2023.131303

The existing works revolve around the task of precise neural rendering and a radiance field with multi-view inputs. The idea of querying a new view from a single view is put forth. Given an image and a new viewpoint as a query, the model learns the scene through 3D scene understanding and generates the rendered 3D. The missing regions are synthesized automatically based on latent features of the input image. The 3D geometry is utilized for cognition, and context understanding is used for filling the missing regions. This shifts more as viewpoints change.

A model is presented for view synthesis from a single picture in real-world settings in this study. The model has been trained from the beginning to end without any ground-truth 3D supervision. It uses a high-resolution point cloud of learned features to describe a 3D scene structure, which is predicted from the input picture using a pair of convolutional networks. A high-performance differentiable point cloud renderer draws the point cloud from the target view to create new views, a scene semantics model that builds on recent improvements in generative models [2] and trains against learned discriminators in an adversarial manner. The model is trained end-to-end using picture pairs, and their related camera postures since all model components are differentiable; at test time, it gets just a single image and a target perspective. The work is made more difficult by the fact that all datasets contain substantial angle changes and translations. The method produces high-quality pictures and outperforms existing voxel-based 3D representation approaches.

The main contributions of this research work are summarised as follows.

1. 3D scene understanding from a single-view image.
2. Generating 3D unseen views from a given query.
3. Synthesizing the missing regions in the newly generated view

The research's direction is structured as follows. Section 2 discusses relevant research in 3D scene rendering and query view transformation. The details of the dataset used in this work are given in Section 3. The problem definition is presented in Section 4. Section 5 elucidates the component in the proposed architecture. The results of the work and other experiments are illustrated in Section 6. Finally, the authors conclude the work and discuss the future scope of this research in Section 7.

2. RELATED WORK

Recent advancements in view synthesis and picture completeness, have been rather rapid. Although massive view shift has been explored in innovative view synthesis work, it often requires multiple 2D input images. When there is just one input picture, completeness becomes crucial for outpainting. The previous works have been dispersed into whether they employ multiple input images or a single image at the testing phase and whether they require annotated 3D or semantic information.

If multiple images of a scene are available, inferred 3D geometry may be utilized to rebuild the scene and then produce new views. Traditionally, depth maps were used to do this. DNN [5], which improves view synthesis from a set of noisy, partial, or inconsistent depth maps, can be used to learn depths. DNNs can be used to learn depth. [9], [5] a DNN is used to enhance view synthesis from a collection of noisy, partial, or inconsistent depth maps. Given two or more views of a scene inside a restricted baseline, [7] achieves outstanding results in combining views within this narrow baseline. [2] learns an implicit voxel representation of one object given many training views and generates new views of that object at test time. It also uses no implicit 3D representation. In this work, only one image at test time is assumed.

To train the 3D representation, a different area of study assumes a huge collection of pictures with accompanying ground-truth 3D and semantic information [8]. These approaches require large references and the annotation work that goes with that as well. A depth or lidar camera or Sfm (Structure from motion) [5] can be used to acquire the depth; however, this is time-consuming and difficult, especially for outdoor situations, necessitating the usage of synthetic settings. This work aims to generate predictions in realistic scenarios without using 3D information or semantic labeling.

DNNs can be used to learn view synthesis from end to end. Another area of work uses only image-to-image transformations to synthesize new perspectives. Later work interprets latent space as an implicit surface or directly conducts 3D operations on the learned embedding [1]. DNNs are used to create high-quality pictures, building on the latest developments in generative models. Moving between the latent codes of various instances of an object class in [7] appears to interpolate posture, but controlling and evaluating explicitly changing pose is complicated. [1] provides for precise position control, but not from a specific image; they also employ a voxel representation, which is computationally restricting. These studies use synthetic datasets with only one item per picture and train one model for each object type. Larger motions that result in major voids and disocclusions in the target picture are ignored. In KITTI [6] they also investigate a more limited configuration, with synthetic object classes and predominantly forward motion. But a variety of numerous outdoor scenes is used.

Recent work in inpainting takes an adversarial approach [12] and has been used in novel view synthesis refinement. Inpainting, however, is not suited for synthesizing large-angle change, which results in significant missing areas. Methods for outpainting [14] increase extrapolation but are not adaptable to random missing patches in view synthesis.

Recent advances in generative models [11] are utilized to produce high-quality images with DNNs. Moving between the latent codes of various instances of an object class appears to interpolate the postures in [7], but directly changing the pose is difficult to regulate and analyze. [10] supports precise position control but not from a given picture; they also employ a voxel representation, which is computationally limited.

3. PROBLEM DEFINITION

The model takes in a single view 2D image I and a query T as input. Spatial features f and depth map d are extracted. The input image is embedded in a feature space F via a spatial feature predictor (F) and a depth map via a depth regressor (D). F and D are combined to form a point cloud P, which is then rendered into the new view, which is a neural point cloud render. The differentiable renderer generates the new view using Rasterization and alpha composite accumulation. It projects P onto a 2D grid under transformation T. A 3D point pi is projected and splattered to a region with a center pic and radius r. The final scene Ig is generated through the Refinement Network (R) by precisely refining the rendered features. At training time, it is enforced that Ig should match the target image (discriminator).

4. DATASET AND SYSTEM REQUIREMENTS

4.1. RealEstate10K

RealEstate10K is a vast dataset of camera postures, 10 million frames of which were obtained from 80,000 or so video recordings. For each clip, the poses create a trajectory, with each pose specifying the camera location and orientation along the trajectory. Running SLAM and bundle

adjustment algorithms on a huge collection of videos yields these positions. Each video clip's timestamps and postures are specified in a separate.txt file that makes up the data. Frames from the training videos can be sampled for a learning application in order to learn, for example, a view synthesis model. Triplets of frames were taken from each clip during training in Google's study Stereo Magnification: Learning view synthesis with multiplane pictures from SIGGRAPH 2018, two for predicting a model, and a third stands out as ground truth for computing a loss of view synthesis that is used to train the network.

4.2. KITTI

The dataset contains hours of traffic scenarios captured with multiple sensor modalities, including high-resolution RGB and grayscale stereo cameras, as well as a 3D laser scanner. Although widely used, the dataset itself lacks ground truth for semantic segmentation. Many contributed to annotating and labeling tasks for applications such as road detection, object categorization, and visual odometry prediction.

4.3. System Requirement

The system has to handle a large amount of data during the implementation of the model. The renderer requires CUDA kernel-facilitated GPU. The hardware minimum of 8GB RAM and 40 GB SSD Free Space. Models are trained using PyTorch, and Quaternion is used for Query Network.

5. ARCHITECTURE

The dataset consists of videos from which 2D single-view image pairs are generated in the first module, Preprocessing. The single-view images, along with the extracted frame meta-information, are fed into Spatial Network and Depth Map Predictor to embark on the process of Scene Understanding. Spatial features and depth maps are extracted in this block, where the model learns the context of the scene. These features are represented as a 3D Point Cloud and further rendered into a 3D scene using a differentiable neural renderer. This is followed by query transformation, where the rendered scene is further transformed into a new view using input queries. Further refinement and optimization are implemented in the Refinement and Synthesis Network. Figure 1 depicts the overall architecture of the proposed work.

5.1. Data Processing

Diverse videos comprising indoor and outdoor scenes are loaded, and frames are extracted to generate a single view image for each instance. Frames are extracted as 2D images using FFMPEG. Further, frame metadata information is also extracted, containing the camera poses and camera parameters. The views are sampled by adopting a reference video frame and then a second video frame separated by a maximum of 30 frames.

5.2. Scene Understanding

The raw input image is mapped into a higher-dimensional feature map and a depth map. The scene semantics are generated in the spatial feature network. The 3D structure of the input image is predicted at the same resolution. Since the 3D rendering is facilitated using a single input image, a comprehensive understanding of the scene for the single view image is vital to render the 3D view.

5.2.1. Spatial Features Extractor

At the same resolution as the source image, the spatial feature network generates feature maps. The scene semantics, or a higher level of representation than just RGB colors, should be

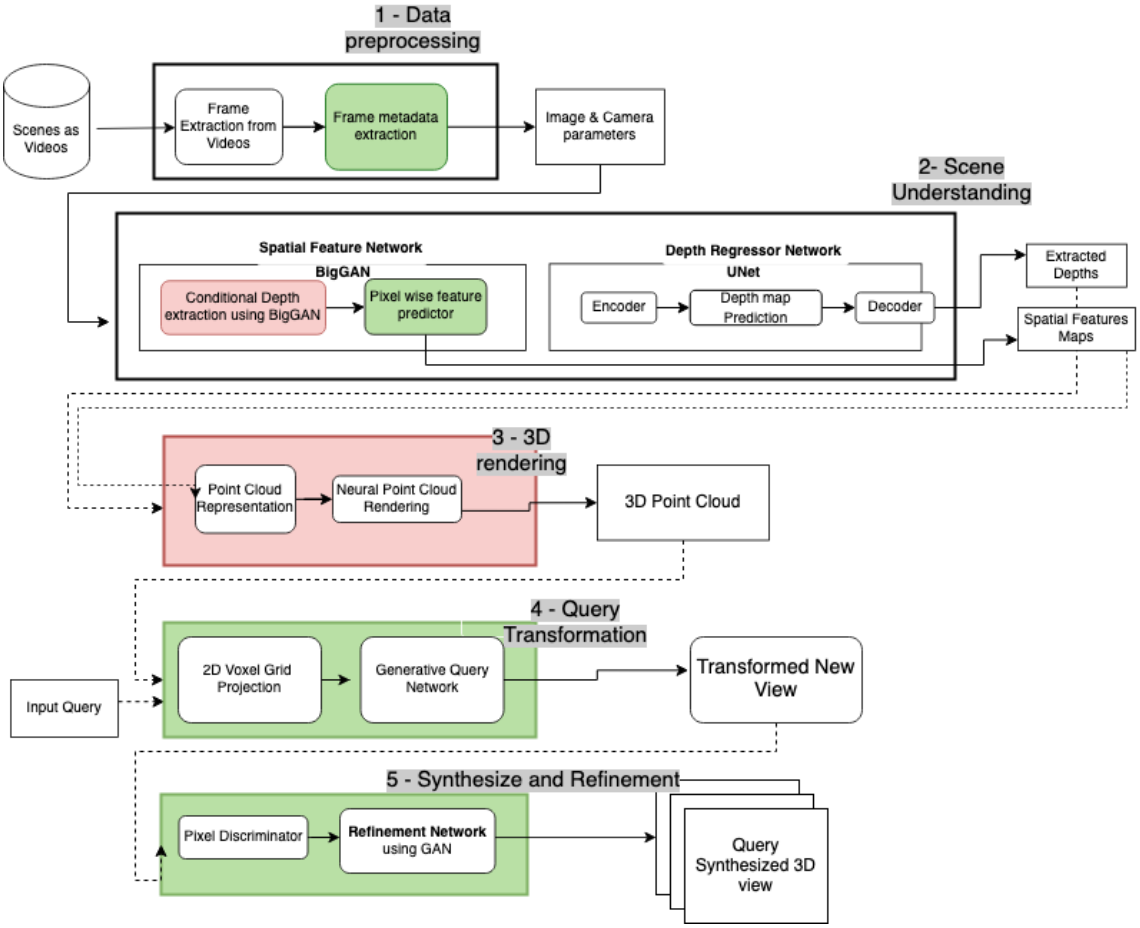

Figure 1. SQ3D - Net Architecture

represented by these feature maps. It is necessary to model 3D structures in order to capture the relative motion of visible objects under a view transform. Understanding the geometry is important for scene understanding which further facilitates the 3D rendering. The proposed work makes use of ResNet blocks. The block is used to increase the resolution of the features using an upsample layer and can progressively learn complex features in each layer. The ResNet blocks are stacked together to form the embedding network in this case. The modified BigGAN is utilized for spatial features reasoning by learning and mapping the higher dimensional features. The discriminator is trained using hinge loss, and the shared embedding layer is linearly projected to gains and biases. Large batch sizes provide better gradient information for updating the models. To directly link the latent input point to particular layers deep in the network, skip connections were introduced to the generator model. In each training cycle, the BigGAN [3] updates the discriminator model twice before updating the generator model.

5.2.2. Depth Predictor

At the same resolution, the depth network assesses the 3D structure of the input picture. To synthesize plausible completions of partially visible objects, semantics must be recognized. UNet, which has the ability for pixel-level localization, is precisely used for depth prediction. The fine-grained details learned in the encoder part of the UNet are used in predicting the depths of the images in the decoder part. Particular attention to the edges and preserving the sharp discontinuities is essential for depth map prediction. The Encoder Block consists of a sequence of Leaky ReLU, convolution (stride 2, padding 1, kernel size 4), and batch normalization layers. The Decoder Block is made up of stacked layers of ReLU, bilinear upsampling, convolution (stride 1, padding 1, kernel size 3), and batch normalization. The per-pixel depth is inferred using the U-Net, and the pixel is mapped to 3D using known intrinsics. The depth network learns the depth via end-to-end training on reprojection losses.

5.3. 3D Scene Rendering

The learned representations are used to generate the 3D model as features of the point cloud using differential rendering. We combine spatial features F and predicted depths D in order to produce a 3D point cloud of feature vectors P. A differentiable neural point cloud renderer is implemented for rendering the 3D scene. Combining the low-level image features with the high-level semantic and 3D object models will facilitate a robust network. Initially, the predicted depths and the spatial features are projected into a 3D point cloud. A Point cloud is defined as a cluster of three-dimensional points distributed in a 3D space. These 3D points all have predictable positions indicated by specific (x, y, z) coordinates, as well as other characteristics like RGB color values. Given the input view transform T, this point cloud needs to be viewed from the target viewpoint. This requires rendering the point cloud with a neural renderer. The neural point cloud renderer proposed by the NeRF [9] is utilized in this work. A differentiable renderer renders point clouds by scattering points throughout an area and collecting via alpha-compositing, mitigating the harsh rasterization decisions.

5.4. Query Transformation

Given the query pose, the 3D image is transformed to map the new viewpoint. Initially, the renderer projects P onto a 2D grid for a given input transformation T. The work models the query network inspired by Generative Query Network [4]. GQN learns the context from observations of the world around it. The GQN learns about likely situations and their geometrical characteristics in this way without any human labeling of scene contents.. The representational network produces the context from scene understanding. And, then the query is given to the generational network, which produces the transformed new view. Further, the rotation and translation for the newly transformed view are enabled by Quartenion

5.5. Synthesize and Refinement

The missing portions hidden from the original view are synthesized automatically using GANs to generate the final 3D output. Our refinement network decodes the projected features to fill in the blanks and provide a realistic output image that is both semantically and geometrically accurate. A Quantized Variational Autoencoder (VQ-VAE) is utilized to refine the query-transformed view and conditionally synthesize the missing regions of the scene. The technique of Autoregressive Outpainting is employed to outpaint using image-specific orderings. The process begins with pixels adjacent to the visible region and moving outward. The model outpaints a vector-quantized embedding space and performs image outpainting on the latent space of the VQ-VAE. To tackle the ambiguity in the inpainting task, batch normalization is injected with noise. And also the

spectral normalization following each convolutional layer. The reprojection of the original and outpainted pixels is blended using an adversarially trained refinement module, which forecasts a residual to its input.

6. EXPERIMENTS AND RESULTS

6.1. Performance Metrics

SSIM Structure Similarity Matrix (SSIM) is used as a metric to measure the similarity between two given images. The range is a value between -1 and +1, where the SSIM is calculated between 2 given images. The number +1 denotes that the two photographs are identical or extremely similar, whereas -1 denotes that the two images are highly dissimilar. These numbers are frequently changed to fall between [0, 1], where the extremes have the same meaning. The three features extracted from the images—Luminance, Contrast, and Structure—are used to compare the two images.

Luminance is given by averaging over all pixel values, and the formula is,

$$\mu_x = \frac{1}{N}\sum_{i=1}^{N} x_i$$

The luminance comparison function, $l(x, y)$ which is a function of μx and μy is,

$$l(x, y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}$$

where $C1$ is the constant to ensure stability

Contrast is the standard deviation (square root of variance) of all pixel values and the formula is,

$$\sigma_x = \frac{1}{N-1}\sum_{i=1}^{N} (x_i - \mu_x^2)$$

The contrast comparison function $C(x, y)$ is given by,

$$C(x, y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}$$

Structure - The structural comparison is performed using a condensed formula, which divides the input signal by its standard deviation to get a result with a unit standard deviation, enabling a more accurate comparison.

$$\frac{x - \mu_x}{\sigma_x}$$

The structure comparison function s(x, y) is then a function of σx and σy where σ denotes the standard deviation.

$$s(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x + \sigma_y + C_3}$$

where σ_{xy} is defined as follows

$$\sigma_{xy} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y)$$

The overall SSIM can be calculated as follows,

$$SSIM = [l(x,y)]^{\alpha} . [c(x,y)]^{\beta} . [s(x,y)]^{\gamma}$$

PSNR is the peak signal-to-noise ratio between two images, measured in decibels, is computed by the PSNR block. The quality of the original and compressed images is contrasted using this ratio. The quality of the compressed or reconstructed image improves with increasing PSNR.

When comparing the quality of image compression, the mean-square error (MSE) and peak signal-to-noise ratio (PSNR) are used. While the MSE reflects the cumulative squared error between the original and compressed picture, the PSNR provides a measure of the peak error. The error is inversely correlated with the MSE value.

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(O(i,j) - D(i,j))^2$$

O denotes the original image matrix data. D represents the matrix data of the degraded image. m is the number of rows of pixels, and i denotes the index of that row. n is the number of columns of pixels, and j denotes the index of that column.

$$PSNR = 10log_{10}\frac{(L-1)^2}{MSE}$$

Here, L is the total number of intensity levels that can exist in an image with a minimum intensity level of 0.

Implementation: On a single GPU, it took 60ms for the forward pass to process a batch of four images of dimensions 256*256 and 10ms for the backward pass. The models were trained on Pytorch for 30K iterations. The train, test, and validation data were split into 30K, 12K, and 7K images, respectively.

6.2. Comparison Systems

We compare the performance of the proposed architecture with the previous works done on 3D Scene Rendering.

NeRF (Baseline) [9] presents a method for synthesizing novel views of complex scenes by optimizing underlying continuous volumetric scene functions. The algorithm represents a scene using a fully-connected (non-convolutional) deep network with a single continuous 5D coordinate (the spatial location (x, y, z) input and viewing direction (θ, ϕ)) and whose output is

the volume density and view-dependent emitted radiance at that specific spatial location. The authors of this work on multiple views of a single object. Their prime idea is a differentiable rendering procedure based on classical volume rendering techniques, which was used to optimize these representations from standard RGB images.

DeepVox [13] By integrating native 3D operations into the neural network architecture, the model seeks to address the basic constraints of 2D generative models currently in use. The authors deliberately encode these operations in the network design and execute reasoning directly in 3D space rather than having the network acquire intuitive ideas from 3D vision, such as perspective. Without explicitly modeling the geometry of the scene, the DeepVoxels technique tries to transform posed input photographs of a scene into a durable latent representation. In order to create previously unknown viewpoints of a 3D scene without having access to the original set of input frames, this representation may then be used for the job of novel view synthesis.

GQN [4] asserts that a single brain architecture is capable of understanding, representing, and perceiving synthetic situations without the need for any human labeling of their contents. Additionally, it has the capacity to build a potent neural renderer that can generate precise and reliable pictures of scenes from novel inquiry views. The GQN learns representations that adapt to and succinctly record the significant aspects of its surroundings, such as the location identities and colors of many objects. GQN uses analysis-by-synthesis to perform “inverse graphics” and learns this behavior by itself and in a generally applicable manner. The network is jointly adapted in the proposed work for the relational query network.

SQ3D-Net (Proposed work) The results using the proposed novel framework SQ3D-Net are compared with other systems for their robustness. Unlike most methods where the 3D view of a single object is rendered, the authors work on Indoor and Outdoor Scenes to account for the increased real-life utility of the latter. A Differential rendering is used instead of Naive rendering as proposed by the [base paper]. Generational Query Network forms the basis for understanding and processing the queries posited by [4]. The 3D transformed view according to the input query is then refined to fill the missing regions by enhanced understanding of the Input scene.

6.3. Result Analysis

The SQ3D-Net models a new approach for view synthesis using a single view input image and incorporates query transformation into 3D scene rendering. The improved results are attributed to the scene understanding module, which enabled semantically and geometrically precise construction of new views.

Table 1 illustrates the results obtained using SQ3D-Net and other comparable systems considered. We train our models, ablations, and baselines on two datasets. We evaluate models based on how effectively they comprehend the data to improve the outcomes analysis through the 3D scene structure and the scene semantics. The performance metrics vary proportionally, ↑ - indicating that the higher, the better the model. The baseline NeRF synthesizes the new view from multiple input images with a PSNR of 20.1, and SQ3D-Net outperforms this with a PSNR of 23.5. The proposed work performs better than Vincent et al. [13] - a single-view synthesis model with an SSIM of 0.76. Apart from the rendering capability, the SQ3D-Net produces convincing results for input query transformation.

Table 1. Results

Model	SSIM ↑	PSNR↑
NeRF	0.76	20.1
DeepVox	0.72	19.5
SQ3D-Net	0.76	23.5

7. CONCLUSION

An end-to-end model has been introduced in this work. The prominent modules are scene understanding and 3D rendering, with the key component being a differentiable neural point cloud renderer. The model was validated on indoor and outdoor scenes with relatively complex arrangements compared to other view synthesis tasks. It may be used to create plausible clips that follow a predetermined course and can be applied straight to images with greater qualities. We consider the new challenge of synthesizing rich and complete scenarios from a single image. 3D awareness is considered an important technique for appreciable results and generalisability. For instance, the application of our system to two views is made possible by our model's 3D awareness. For instance, our model's 3D awareness enables the application of our system to two views. The authors' long-term objectives include strengthening the model for high-resolution photos and making it applicable to different input resolutions. The work can be extended to deal with complex input queries with a wider range of inputs. Enhancing the work with explainability can provide a reasonable 3D rendering of the provided 2D indoor and outdoor scenes.

REFERENCES

[1] Deepvoxels: Learning persistent 3d feature embeddings. In: Proceedings - 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2019. pp. 2432–2441. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE Computer Society, United States (Jun, 2019).

[2] Brock, A., Donahue, J., Simonyan, K.: Large scale gan training for high fidelity natural image synthesis.

[3] Brock, A., Donahue, J., Simonyan, K.: Large scale GAN training for high fidelity natural image synthesis. In: International Conference on Learning Representations (2019)

[4] Eslami, S.M.A., Rezende, D.J., Besse, F., Viola, F., Morcos, A.S., Garnelo, M., Ruderman, A., Rusu, A.A., Danihelka, I., Gregor, K., Reichert, D.P., Buesing, L., Weber, T., Vinyals, O., Rosenbaum, D., Rabinowitz, N., King, H., Hillier, C., Botvinick, M., Wierstra, D., Kavukcuoglu, K., Hassabis, D.: Neural scene representation and rendering. Science 360(6394), 1204–1210 (2018).

[5] Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: The KITTI dataset. The International Journal of Robotics Research 32(11), 1231–1237 (2013)

[6] Iizuka, S., Simo-Serra, E., Ishikawa, H.: Globally and locally consistent image completion 36(4) (2017)

[7] Liu, A., Tucker, R., Jampani, V., Makadia, A., Snavely, N., Kanazawa, A.: Infinite nature: Perpetual view generation of natural scenes from a single image. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (October 2021)

[8] Liu, S., Chen, W., Li, T., Li, H.: Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV). pp. 7707–7716. IEEE Computer Society, Los Alamitos, CA, USA (Nov, 2019).

[9] Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I. p. 405–421. Springer-Verlag, Berlin, Heidelberg (2020).

[10] Nguyen-Phuoc, T., Li, C., Theis, L., Richardt, C., Yang, Y.L.: Hologan: Unsupervised learning of 3d representations from natural images. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (October 2019)
[11] Park, T., Liu, M., Wang, T., Zhu, J.: Semantic image synthesis with spatially adaptive normalization. CoRR abs/1903.07291 (2019)
[12] Rombach, R., Esser, P., Ommer, B.: Geometry-free view synthesis: Transformers and no 3d priors (2021)
[13] Sitzmann, V., Thies, J., Heide, F., Nießner, M., Wetzstein, G., Zollh¨ofer, M.: Deepvoxels: Learning persistent 3d feature embeddings. In: Proc. Computer Vision and Pattern Recognition (CVPR), IEEE (2019)
[14] Yang, Z., Dong, J., Liu, P., Yang, Y., Yan, S.: Very long natural scenery image prediction by outpainting. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (October 2019)