Master Thesis

Learning-based novel view synthesis from sparse 3D point clouds and images

（疎な３次元点群と画像のセットからの自由視点画像生成）

Chuyi Wang
Advisor: Professor Yoichi Sato
Submission Date: Jan 30th, 2020

Department of Information and Communication Engineering
Graduate School of Information Science and Technology
The University of Tokyo

Advisor
Prof. Yoichi Sato

Abstract

Novel view synthesis is one of the challenging topics in computer vision, which aims to synthesize images from unknown viewpoints with images seen from captured viewpoints. Nowadays, with the development of Virtual Reality techniques, it has become an increasing concern since rendering of free viewpoint images is one of the basic problems in Virtual Reality. If we want to roam in a scene, it means that we need to render the scene from different viewpoint. Traditional 3D modeling and rendering could construct a photo-realistic scene with manual modeling software. But high computational cost is inevitable since we usually need large illumination calculation and complex geometric mapping and cropping. The computation cost is high. Therefore, people began to pay more attention about whether we could get a virtual, unknown viewpoint from a collection of images directly. That is what novel view synthesis technique mainly focus on.

Image-based rendering technology has been applied to novel view synthesis problem over the last two decades. Earlier works try to render new images without 3D geometry information by using simplified forms of the plenoptic function, while later works utilized structural information to help interpolate novel image from known image pixels directly. In addition, recent advances in deep learning have also made it possible to improve the performing of novel view synthesis by leveraging a deep learning network. Different from traditional approaches, deep learning model provides an end-to-end rendering and could learn to synthesize pixels from large amount of images directly without complex formulas and computation. This kind of methods utilize the ability of neural network to adapt wide-baseline cases. in recent years, a lot of works in novel view synthesis began to combine geometry-based method and learning-based method. In order to build a realistic novel view of an unknown scene, [PKKS19] and [CGT+19] proposed pointcloud-based method to synthesize novel views by using image descriptors and semantic labels respectively. However, there still remains some limitations. Artifacts are generated from the quality of auxiliary information, especially for the virtual viewpoints which are far away from existed viewpoints. Inspired by these, we aim to leverage original image priors directly to improve the synthesis results. We would like to propose a two-staged synthesis network based on images and a sparse point cloud from structure-from-motion(SfM) technique, which preserve basic structural information of the scene. For the synthesis stage, we build a point cloud-conditional generative adversarial network to generate an initial novel view. Then for the refinement stage, we leverage image priors selected from nearby viewpoints to correct the poor reconstruction and artifacts. We acquire visually pleasing results with this two-staged network.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Overview

Figure 1.1: Novel view synthesis [Alj17]. The problem setting is that for one scene, if we have a sparse set of images from blue cameras, we would like to get rendering of other virtual views from orange cameras.

Novel view synthesis is one of the long-standing problems in computer vision and has developed rapidly in recent years due in part to application in virtual reality. Classical view synthesis problem aims to generate novel views from single or multiple views of a scene. Basically, the problem setting is generating pixels of a novel view from pixels of neighboring views.

Many approaches try to address this problem with view interpolation. A lot of earlier works employed image-based rendering methods and have presented various solutions, such as light field rendering [LH96, GGSC96], image warping [SD96] and estimation for shape or visual appearance [VBR+99, ZKU+04, SAC+13]. Recent works focus on combining geometry information with image warping together. Well-performed image-based rendering usually require strong priors to recover the holes generated by uncertain geometry and occlusions. Hence, this kind of methods are usually largely hand-built and of high computational cost. However, it is still not an easy work to remove artifacts especially for complex objects such as tearing, aliasing and distortion by explicitly modeling the color, occlusion and stereo for each pixel in novel view.

Furthermore, with the development of deep learning techniques, researchers begin to apply deep networks to overcome these limitations. As deep networks have already performed well for super-resolution [DLHT14], denoising [XXC12], style-transfer [GEB16] and distribution estimation [GPAM+14, MO14]. Actually, novel

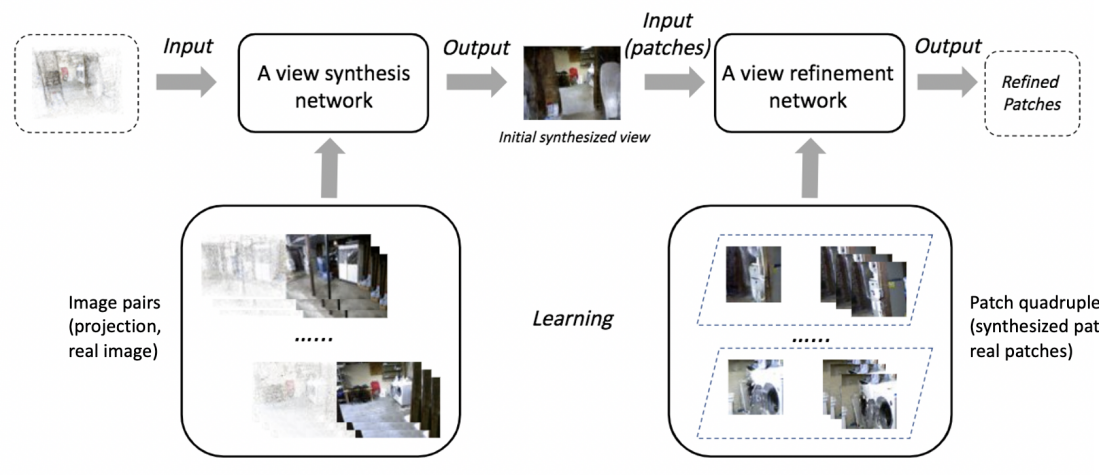


Figure 1.2: The concept of our novel view synthesis model.. We attempt to build a novel view synthesis model consists of two sub-networks, which has the potential to learn virtual views from a set of images. When given a projection view, the synthesis network could generate an initial synthesized result, then we refine it with the refinement network at patch level.

view synthesis can be simply seen as a learning task by training a model that generate pixels for a novel view. Objective function are formulated by comparing synthesized views to the ground truth. Recent works have applied deep learning to view interpolation. Zhou *et al.* [ZTF+18] proposed a method to extract a layered representation of the scene and perform extrapolation from two input views. Choi *et al.* [CGT+19] further improved the extrapolation range and generate novel views father away.

However, unlike these works, our goal is to synthesize an arbitrary novel view of a scene, as seen in Figure 1.2. We achieve the goal by utilizing a relatively sparse image set of the scene. Our problem setting is more like [PKKS19], we aim to train a novel two-staged neural networks for scene reconstruction by synthesizing arbitrary views of the scene without knowing any auxiliary information or neighbouring views. Technically, our model can be divided into synthesis network and refinement network. 1) We acquire 3D shape from image collection by constructing a sparse point cloud and invert it to synthesize novel views from arbitrary viewpoint with synthesis network. 2) For each target views, we select relevant patches from ground truth near by, and jointly train the refinement network at the patch level. We demonstrate the effectiveness of refinement network.

1.2 Challenges and Contributions

One fundamental challenge of novel view synthesis is the difficulty of acquiring accurate geometry information. Early works on novel view synthesis employed image

Figure 1.3: Occlusion problem. A 3D point cloud projection with its corresponding ground truth. It is obvious that the blue box in the red circle should be occluded in the view and points belong to it should be removed from the projection, otherwise occlusion problem would happen.

interpolation methods decades ago. These methods interpolate pixels of novel views between corresponding pixels of input images [CW93] and space rays [LH96], or conducting a weighted combination with geometry information [BBM+01, DTM96]. This kind of approaches works based on the premise that additional depth or 3D geometry are given as input.

However, the most common situation is that only images are known. Therefore, in order to obtain additional information, one solution is to estimate depth information from images directly [KWR16, CDSHD13, PZ17]. Another approach is to design an intermediate representation including geometry information of the scene, as Zitnick *et al.* [ZKU+04] and Zhou *et al.* [ZTF+18] have done in their works. However, there still remains some limitations. For depth estimation, artifacts could be generated by depth uncertainty. What is more, as shown in Figure 2.5, it is difficult to know full 3D information of all visible geometry due to occlusion problem from input images even with an intermediate representation. Therefore, the geometry for visible surfaces may be ambiguous and uncertain. These problems occur more often in extrapolation rather than interpolation from input images. For methods using intermediate representation of the scene, Zhou *et al.* limit the virtual camera to translate along the baseline within a certain range. Furthermore, Choi *et al.* [CGT+19] allow more flexible movement and larger magnification with referenced input images.

Recently, some pointcloud-based view synthesis methods have been proposed [CGT+19, PKKS19]. [PKKS19] build a network with three sub-networks for scene reconstruction by using SIFT descriptor. However, the SIFT descriptor of each 3D point is sampled from an arbitrary source image in the image set, which means it could lead to unknown distortions and inconsistencies. [CGT+19] train an appearance encoder and rerenderer with semantic labelings of high quality, but it is not an easy work to obtain good segmentation mask and ghost pole may appear in synthesized images.

Figure 1.4: Challenges of repairing artifacts caused by depth uncertainty. [CGT+19] (a)A 3D point cloud generated from the depth map of red camera (b)(c)(d) Three different views from the blue, red, green cameras. It is obvious that there would be large artifacts due to depth uncertainty if the viewpoint is far away from the red one.

In address these problems, we propose a two-staged novel view synthesis method and aim to generate novel views from an arbitrary appointed viewpoint. In our case, what we have is a sparse set of images and a point cloud generated from them. Furthermore, in order to solve the artifacts caused by photometric inconsistencies and distortions, we directly leverage original images nearby the virtual viewpoint.

Our main contributions are summarized as follows:

- We further define the novel view synthesis problem with sparse set of images and propose a approach based on two neural networks: synthesis network and refinement network. As shown in Figure 1.2, the synthesis network generate initial novel views from point cloud projections, while the refinement network further refine the artifacts with ground truth patches.
- We reconstruct the scene roughly with a sparse SfM point cloud extracted from images and present a method to find neighbouring viewpoints to help us refine the synthesis results.
- We validate the effectiveness of utilizing original image patches to correct artifacts in synthesized images rather than using auxiliary information such as image descriptors or segmentation masks.

1.3 Thesis Outlines

The outline of this thesis is organized as follows. In Chapter 2, we make an introduction of recent related works on novel view synthesis problem, including image-based rendering methods and deep learning-based methods. In Chapter 3, we present our approach in detail, including the data set preparing, the network of generation stage and refinement stage. In chapter 4, we evaluate our approach by comparing results from generation stage and refinement stage with qualitative analysis. Then we made

a discussion in chapter 5 and show more visualization results. At last in Chapter 6, we made a summary of this thesis.

2 Related Work

We made an overview of recent novel view synthesis works in this section. We classify them into two main categories: images-based rendering and learning-based rendering, and we especially focus on the latter one.

2.1 Image-based rendering

Over the last two decades, people have been trying to create free viewpoint scenes roaming without time-consuming geometric modeling processes. The difficulty is that, without 3D modeling process, how could we generate images of arbitrary viewpoints from discrete cameras. Image-based rendering is the technique to solve this issue. Intuitively, it should be able to get the visual content from discrete input images and re-project them to a novel view.

2.1.1 Rendering without 3D geometry information

Earlier works try to solve this problem without using 3D geometry information. Traditional rendering follows plenoptic function to render the observed the objects in the scene [MB95, LH96]. With a plenoptic function, we could actually get the panorama of specific viewpoint. Furthermore, we are able to simplify the plenoptic function with Concentric Mosaics [SH99], which has only three parameters and is much easier to be constructed. In addition, Debevec *et al.* [DDB+15] proposed a method to realize scene roaming with constructing a spherical light field capture system. Therefore, novel views can be rendered within the sphere outlined by fish eye cameras.

However, this kind of method can only render novel view within a certain range, since it is difficult to render the viewpoints far away from the rotation center. In addition, the complicated capturing process has also increase the cost of data collection.

2.1.2 Rendering with 3D geometry information

Intuitively, it is inappropriate to ignore 3D geometry information contained in images due to the high complexity of Real-world scenes. With the development of information acquisition technology, we could obtain spatial relationship between

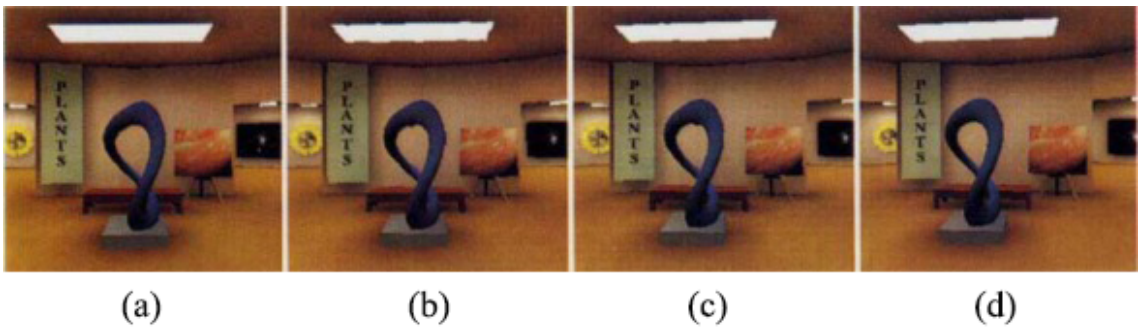


Figure 2.1: Image interpolation with correspondence. (a) and (d) are images as correspondence, (c) and (d) are two middle views interpolated from correspondence.

different objects from 2D images, which would be an auxiliary information for rendering. This 3D information can be described in many different forms in rendering process, such as depth maps or depth distributions [SGHS98, CBL99, PZ17], triangle mesh [BBM+01, Rad99, VBK02] and 3D point cloud [CDSHD13, LKM14], or utilizing only correspondence [CW93, SD96, NZS+16] as shown in Figure 2.1. In rendering, color information and 3D information can be projected to any viewpoints according to the camera positions and parameters. However, there still remains some difficulties. First, there will be missing regions in rendering results since not all pixels can be found in existed viewpoints. Second, visibility and de-occlusion problem may lead to poor reconstruction. In order to solve these problems, we need some post-processing to complete missing 3D geometry information and deal with discontinuity and inconsistencies.

Chen and Williams [CW93] has proposed a method interpolate arbitrary viewpoints with optical flow. They use morphing to generate intermediate frames. However, this method performs well only in the case where two referenced viewpoints are relatively close and there is a lot of overlapping area in input images.

In order to deal with the situation where input viewpoints are far away from each other, Nie *et al.* [NZS+16] have proposed a method which is a combination of homography fitting and homography propagation. In detail, for the matched super-pixels, they calculate a homography with correspondence estimation method. For other poor matched super-pixels, they use propagation to obtain their homographies. Therefore complex 3D reconstruction and accurate depth map are not necessary in rendering process, so that the computational cost is not high. However, de-occlusion problem cannot be solved without considering depth relationship between different objects.

Chaurasia *et al.* [CDSHD13] proposed a method that associate depth synthesis algorithm to sample depth value for areas with poor recovery of depth. They assumed that pixels in the same super-pixel share same depth value. After obtaining the complete depth map, they use shape-preserving warping to handle inconsistency

problem and rendering error. However, obviously, the shortcoming of this method is that depth estimation may be wrong if there is no corresponding super-pixels with enough depth samples in the image.

Different from using depth map or 3D point cloud extracted by multi-view geometry techniques, soft 3D reconstruction estimates a distribution of depth values instead of fixed one for each pixel. Penner *et al.* [PZ17] have proposed a method based on soft 3D reconstruction and visibility function. They first estimate an initial depth map with multi-view stereo. In order to obtain the depth distribution, they build several layers of depth planes for different values and record vote-value and vote-confidence for each pixel. Then, the visibility information can derive from depth distribution, which could help render more visually pleasing novel views.

2.2 Learning-based rendering

With the rapid development of deep learning, researchers have paid more attention to applying deep learning techniques to novel view synthesis problems. As learning-based methods still take images as a fundamental element, it is of great importance to utilize and invert image features which could improve the performance of rendering network. We would like to introduce three recent works in detail in this session which are most related to our work and inspire us a lot.

Pittaluga *et al.* [PKKS19] present a novel view synthesis method based on point clouds obtained from structure from motion(SfM) [SF16]. They use not only the SfM point cloud, but also SIFT descriptors to reconstruct details in a view. What is more, in order to handle visibility problems better, they construct a three-stage training based on three different neural networks: VisibNET, CoarseNET and RefineNET. First they project each 3D points in the point cloud from SfM to specific viewpoints with camera parameters. For each 2D pixel in images, 3D point attributes such as color information and SIFT descriptor are associated. Therefore, the input to the networks could be a multi-dimensional nD array. The VisibNET is responsible for visibility estimation. As the point cloud is relatively sparse, it is impossible to estimate if a 3D point belongs to foreground objects without knowing the underlying geometry of a scene. After removing 3D points belong to surfaces which are occluded, they build a CoarseNET to perform novel view rendering and train this network by combining L1 pixel loss and L2 perceptual loss. At the last stage, a RefineNET is trained to further improve the quality of synthesis by adding an adversarial loss. Each of the sub-networks is trained separately and shares similar U-Nets architecture composed by an encoder and a decoder. Figure 2.2 has shown details of their network architecture with three sub-networks.

Meshry *et al.* [MGK+19] have also proposed a neural rendering method based on 3D representations. As shown in Figure 2.3, They perform a staged training as well. The first stage is to train a encoding network to estimate appearance vectors for

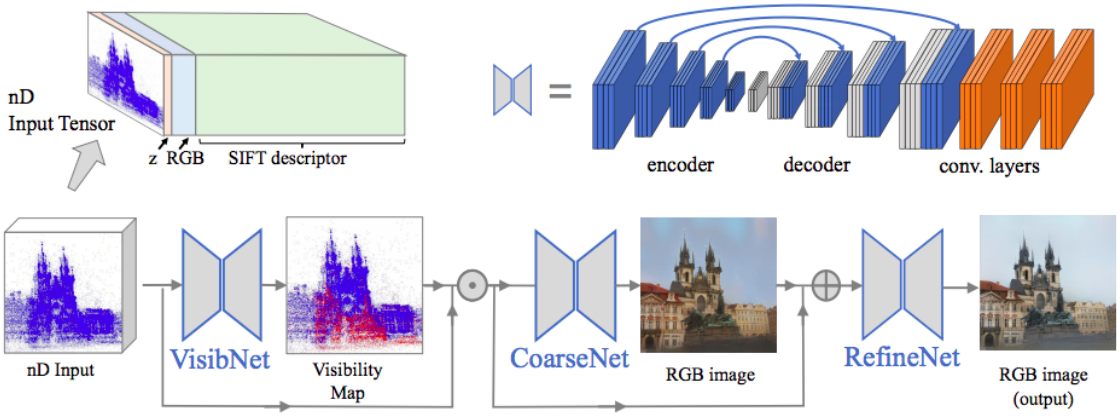


Figure 2.2: Network Architecture of [PKKS19]. The network consists of three sub-networks: VISIBNET, COARSENET and REFINENET. The input to this network is a multi-dimensional array composed by depth value, color and SIFT descriptor. All of the three sub-networks share similar U-Nets architecture and symmetric skip connections between encoder and decoder. There are also extra convolutional layers concatenated to the end of the decoder, which could help interpret high-dimensional information.

each views. Then for the rendering stage, a neural network is trained with fixed appearance vectors. Both of the networks are jointly fine tuned at last. What is more, after rendering a given viewpoint with appearance embedding, they also condition the network on a desired semantic labeling. This could prevent artifacts due to the encoding variations of transient objects. They expect better performance with uncontrolled images from internet rather than images from carefully collected data sets.

However, the limitation is that their method relies heavily on semantic, especially for the regions of the image lack geometry information, such as the ground and the sky. Figure 2.4 has shown some artifacts due to errors in predicted segmentation. Therefore, it is difficult to validate this method if we don't have ground truth segmentation mask or cannot guarantee the accuracy of the semantic labeling network.

Choi *et al.* [CGT+19] have also proposed a solution to extreme view synthesis with a depth probability volume estimation and a refinement network, which aims to render novel views from small number of input images. First they generate depth probability volume for each novel viewpoints by warping and fusing of input depth information, then to get an initial novel view from its depth volume. For artifacts caused by depth discontinuities, they build a refinement network by leveraging candidate patches selected from input views. The key is that patches in input views are extracted by regions re-projected from same depths of corresponding parts in target novel view. In order to solve distortion and deformation problems, patches

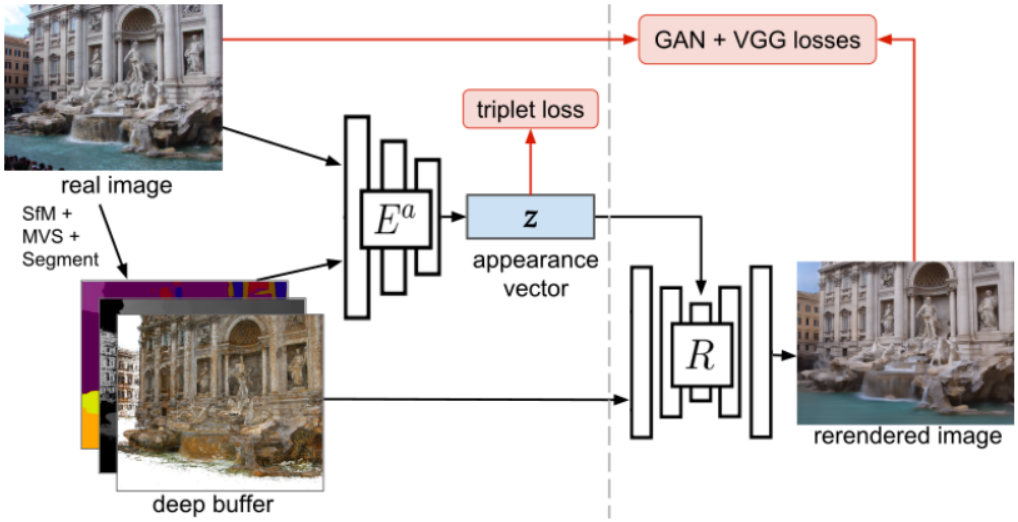


Figure 2.3: Network Architecture of [MGK$^+$19]. Meshry *et al.* proposed a staged method. The first step is to pre-train a encoder to estimate appearance vector by using semantic labels, SfM output and deep buffer. Then train a rerender model by generative adversarial loss and reconstruction loss. Finally they fine-tuned the model together.

are warped with the homography calculated from the depth plane. In the training stage, they take poor construction from initial synthesized views as well as all candidate patches as input to a U-Net architecture network. The model is trained by perceptual loss [JAFF16] and ADAM [KB14] optimization algorithm.

Our work shares the same idea with [PKKS19] and [MGK$^+$19] that synthesis novel views based on sparse point clouds. According to this, we present a two-stage synthesis network. For the synthesis stage, we build a point cloud-conditional generative adversarial network to obtain an initial novel view. Then for the refinement stage, we leverage image priors selected from three nearest viewpoints to refine initial synthesis.

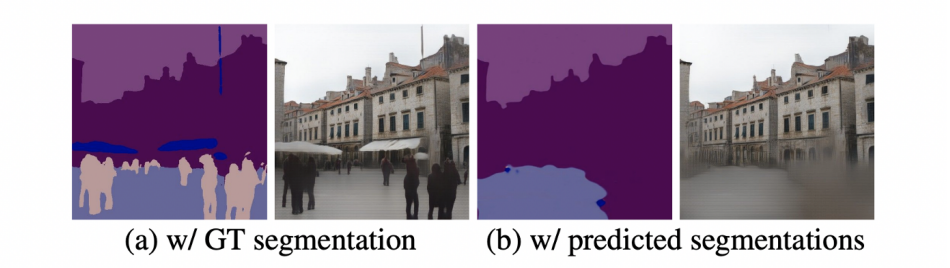


Figure 2.4: Example semantic labelings and output views of [MGK+19]. It is obvious that output view is better by using ground truth segmentation mask. Artifact appears on the bottom when generated from predicted segmentations since there is a misclassification of ground and building.

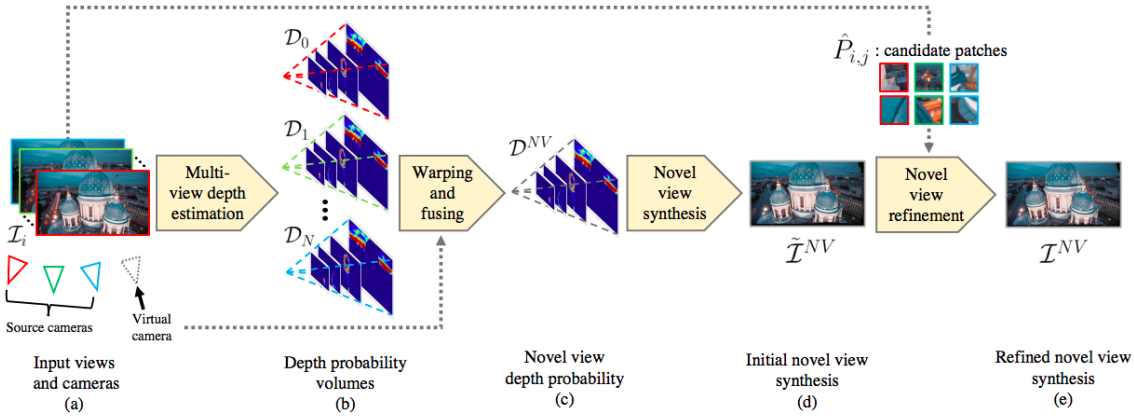


Figure 2.5: Network Architecture of [CGT+19]. Choi *et al.* proposed a method for novel view synthesis from multiple input views (a). First generate depth probability volumes by multi-view depth estimation for each view (b). Warp and fuse the input depth probability volumes to get a novel view depth probability with given camera parameters. (c). Generate an initial novel view directly. (d). Refine the result from last stage with a refinement network to generate the final novel view (e). The refinement network is trained at patch-level where patch selection is guided by the depth distribution.

3 Proposed Method

In this section, we would like to describe the overview of our 2-stage training model for novel view synthesis first. Then we introduce the details of the framework, especially the proposed refinement network which helps refine the initial synthesis by involving patches selected from nearby viewpoints.

Our goal is to learn a generative model for novel view synthesis. Given a set of images, we first obtain sparse Structure-from-Motion point cloud as is done in [PKKS19, MGK+19] and project the 3D point cloud into arbitrary viewpoints, which could be an initial construction of detailed views of the scene. Figure 3.1 has shown some examples of projected 3D points associated with source images.

The view synthesis model generate a initial synthesized view based on these paired images. However, in general the resulting images will present some artifacts, blurs and distortion, since we are not concerned with visibility problems and sparse point cloud cannot preserve complex structures of the scene. Therefore, at this stage we could only roughy generate initial estimates that contain general color information and geometry information. Furthermore, the view refinement model is trained to recover apparent artifact. Since complex structures are easily to be locally deformed, we train this model to repair poor construction regions at the patch level. For a key

Figure 3.1: The illustration of real images with corresponding point cloud projections. The sparse point cloud with 54,665 vertices is generated from Structure-from-Motion. It could be projected into arbitrary viewpoints with specific camera parameters.

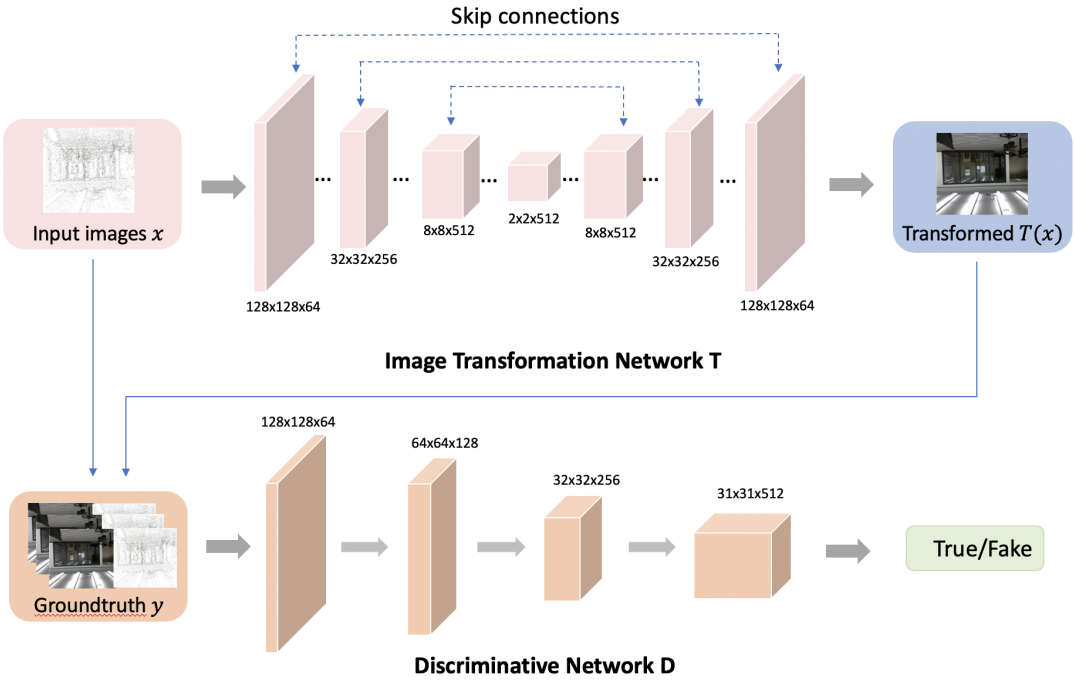


Figure 3.2: The illustration of our synthesis network. The image translation network T is a U-Net based framework with an encoder and a decoder. There are 8 convolutional layers for both encoder and decoder of T and 5 convolutional layers for discriminator D. We also build skip connections from the layers of the encoder to the decoder. For the generator T, input is projected point cloud x and output is a synthesized image $T(x)$. For the discriminator D, input is projected point cloud x combined with a synthesized image $T(x)$ or a ground-truth image y, and the output is a prediction to estimate whether the image pair is real or fake.

pixel in a initial estimate, we extract corresponding 64×64 patches from viewpoints which are close to the novel viewpoint. The target of the refinement work is to output a final results with less artifacts. We will illustrate each module in detail in the following sections.

3.1 Synthesis of a Novel View

Similar to [PKKS19] and [MGK+19], we introduce a generative model based on point cloud projections. In order to generate realistic images effectively, Our model bases the famous pix2pix framework [IZZE17], an image-to-image translation model with conditional Generative Adversarial Network [MO14]. Generative adversarial networks (GANs) is a generative model that aim to learn distribution of true data without formulating a complex equation. GANs learn a mapping from random noise to output image.

In our case, the goal of synthesis stage is to map point cloud projections x to real images y with a translator T, $T : x \rightarrow y$, which can be seen as a domain transfer process essentially.

The model takes groundtruth RGB images and the corresponding point cloud projection as input. Specifically, The image translator T is trained to produce output images that are hardly distinguished from real images. The input is projected point cloud x and output is synthesized images $T(x)$; The discriminator D is trained to detect fake images from translator T. The input image pairs are the combinations of point cloud projections x and synthesized images $T(x)$ or ground-truth images y, while the output is a prediction to estimate whether the associated image is a real or fake pair to the point cloud projection. Both image translator and discriminator are trained simultaneously while optimizing the objective function of both T and D. The objective of translator T can be expressed as:

$$T_{loss} = \log(1 - D(T(x))) + \lambda \mathcal{L}_{L1}(T) \tag{3.1}$$

Where we use L1 loss to encourage less blur:

$$\mathcal{L}_{L1}(T) = \mathbb{E}_{x,y}[\|y - T(x)\|_1] \tag{3.2}$$

The objective of discriminator D can be expressed as:

$$D_{loss} = -\log(D(x, y)) - \log(1 - D(x, T(x))) \tag{3.3}$$

As is shown in Figure3.2, we use a framework based on U-Net [RFB15] with skip connections to build the image translation network T, which could deliver shared low-level information between input and output. Specifically, there are 8 layers in both encoder and decoder. We add skip connections between layer i in encoder and

layer $8 - i$ in decoder with simple concatenations of all channels. For the discriminator D, we build 5 convolutional layers and adopt PatchGAN [DU18] to learn high frequency structure at patch level. The discriminator is trained to estimate if each 256×256 patch in the input image is real or fake. Follow the standard training approach from [GPAM+14], we alternatively train the translator T and discriminator D by optimizing the objective functions above.

3.2 Image Refinement

Images generated from synthesis stage still suffer from apparent artifacts. Most notably, these include regions where point cloud is relatively sparse, so that the generator have not received sufficient information and structures may be locally deformed. In order to address these artifacts, we train a refinement network to improve the quality of images furthermore.

3.2.1 Find nearby viewpoints

In order to refine regions apparently affected by artifacts, a possible approach is to train a refine network at patch level. We could consider the denoising operation that uses synthesized and ground truth patches as input at training time, and tries

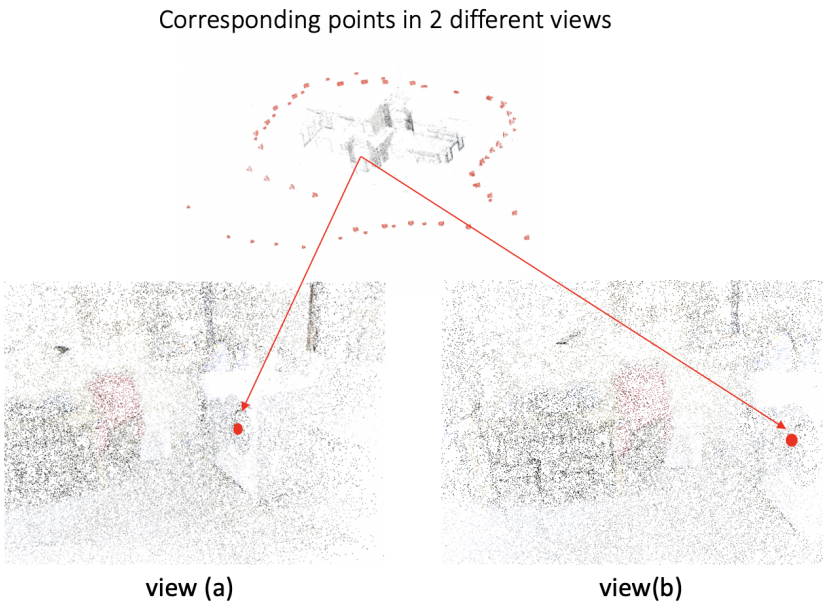


Figure 3.3: The distance between two cameras is equivalent to average euclidean distance between key points in image coordinate. The locations of 3D points could be calculated by the camera poses of different viewpoints.

to output refined results by optimizing an appropriate objective function. However, this kind of method tends to utilize only generic image priors and easily neglect the valuable information of input images at inference. Intuitively, we should consider a refinement operation with utilizing more ground truth images together at both training and inference time. Since our goal is to synthesize novel views from a set of images, it is unreasonable to disregard the valuable information the images carry, especially for those located close to the target virtual view.

Therefore, we turn to find the way of utilizing ground truth patches to improve our synthesis results. First of all, the problem is to select valuable images among the whole image set. Intuitively, cameras should capture similar views when appearing close to each other. That means for a novel viewpoint, we could find cameras close to the target camera to get image priors with relatively small distortion.

Although the location of cameras is determined by camera parameters including camera intrinsics and camera extrinsics, it is not visually or intuitively understandable to calculate distance between camera matrices directly. Therefore, instead we measure the distance between cameras by utilizing point cloud projections. For example, as shown in Figure 3.3, a single point in the 3D point cloud can be projected to different viewpoints. It is obvious that the distance between view(camera) k and view(camera) l is equivalent to average euclidean distance:

$$d_{k,l} = \frac{1}{n}\sum_{i=0}^{n-1}\left\|(x_i^k, y_i^k) - (x_i^l, y_i^l)\right\| \tag{3.4}$$

Where (x_i^k, y_i^k) indicates the image coordinate in view k of $p_i \in \{p_1, p_2, ..., p_n\}$, where $\{p_1, p_2, ..., p_n\}$ are shared points appear in both view k and view l, views $v_k, v_l \in \{v_1, v_2, ..., v_n\}$. In order to obtain the 2D image coordinate from the 3D world coordinate, we apply camera parameters to coordinate system transformation.

$$Z_c \begin{bmatrix} x_i^k \\ y_i^k \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & x_0 & 0 \\ 0 & f_y & y_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R^k & T^k \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix} \tag{3.5}$$

(3.5) describes the process of transforming, where R^k and T^k are rotation matrix and translation matrix of camera$k$, f is camera focal length, while (X_i, Y_i, Z_i) denotes the world coordinate of point p_i and (x_0, y_0) indicates the principal point of image coordinate. After obtaining image coordinates, points placed in areas that are outside of the frusta of input cameras are removed from the point sequence.

For each novel view, we could find its three nearest views among input groundtruth images and select corresponding patches from those views for training. For each point in the point sequence, we extract its surrounding region(a 64x64 box) from initial synthesized images and referenced groundtruth views separately. Intuitively,

relevant patches from groundtruth views can inform the refinement network about correct content and structure information. We describe our refinement network and its training in detail in the next section.

3.2.2 Refinement network

We show our refinement network and strategy in Figure3.4. We take quadruplets of patches consist of a patch $P_{i,j}^{NV}$ from initial synthesized views $\mathcal{I}_i$ and three candidate patches $P_{i,j}^{1}, P_{i,j}^{2}, P_{i,j}^{3}$ from referenced views as input. The candidate patches contributed to each $P_{i,j}^{NV}$ are selected from referenced views nearest to view $\mathcal{I}_i$.

The refinement network is developed based on a U-Net architecture since it has performed well on many vision problems by propagating information to higher resolution layers with multiple feature channels. Rather than concatenating patches, we would like to make each of the patches go through the encoder network independently. Before concatenation with the output of synthesized patch, features generated from different candidate patches should be aggregated with different weights according to their locations. Intuitively, the camera is closer, the more information it contributes. Therefore, we define aggregate weights for each candidate patch by the reciprocal of distance to target view i:

$$w_i = \frac{{d_{i,k}}'}{{d_{i,k}}' + {d_{i,l}}' + {d_{i,m}}'},\ where\ {d_{i,k}}' = \frac{1}{d_{i,k}} \tag{3.6}$$

We perform three kinds of pooling operation for aggregation in the refinement network to show the contribution of each view:

Max-pooling: Apply a max filter over the features of candidate patches to down-sample an representation from all relevant patches, for each value of the feature map: $max(f_{v_1}, f_{v_2}, f_{v_3})$

Average-pooling: Apply a average filter over the features of candidate patches to down-sample an representation from all relevant patches, for each value of the feature map: $\frac{1}{3}\sum_{i=1}^{3} f_{v_i}$

Weighted-pooling: Apply a weighted filter over the features, where we use the viewpoint distance to estimate the contribution from each candidate views: $\sum_{i=1}^{3} w_i f_{v_i}$

There are seven convolutional layers in encoder and four of them are down-sampling layers where we use skip connections to corresponding layers in decoder. In detail, for each skip connection, features of synthesized patches and aggregated features of candidate patches are concatenated together. We train the refinement network using a Mean Square Error(MSE) loss with ADAM at the present stage.

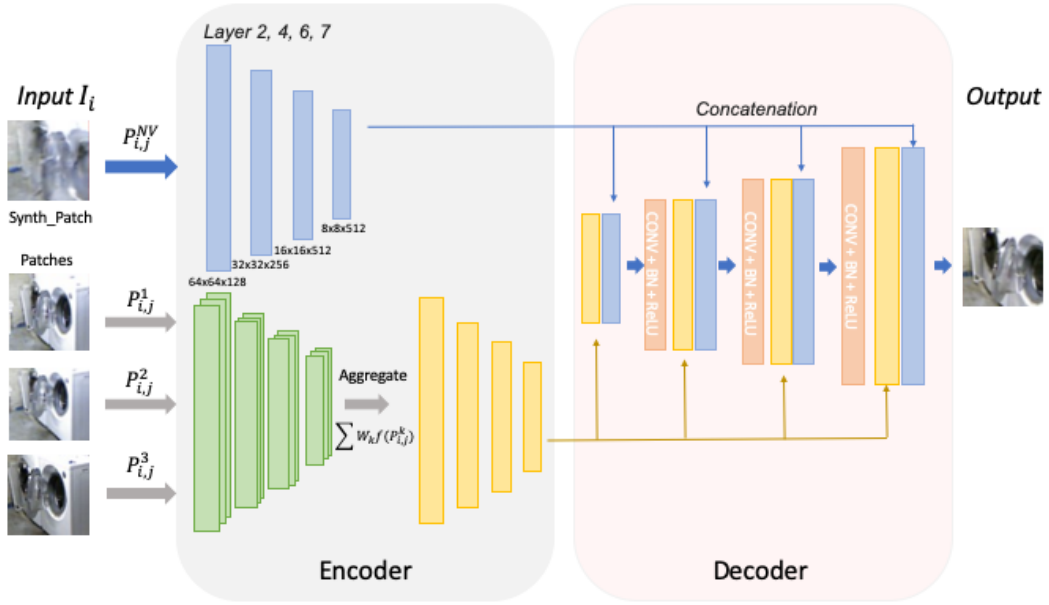


Figure 3.4: The illustration of our refinement network. The U-Net based framework is composed by an encoder and a decoder. For the encoder, we take a patch of synthesized images from synthesis stage as input, and a variable number of corresponding patches from three of the cameras nearest the target viewpoint. All patches go through the encoder separately. The encoder has 7 convolutional layers and we extract features of layer 2, 4, 6, 7, which is down-sampling layers. Then we perform aggregation with aggregate weights, defined by the distance between target views and referenced views. Intuitively, the closer the camera is, the more feature information it will contribute. For the decoder, we use skip connections. The input are feature sets of patches from the four down-sampling layers of synthesized images and candidate views. Features of the synthesized patch and features of candidate patches after a weighted-pooling operation are concatenated in the decoder at layer0, 2, 4, 6 to generate final refined patch. We perform Batch Normalization after each convolutional layer in both encoder and decoder

3.3 Implementation Details

We generate sparse point cloud from image collection first by COLMAP [SF16] , and obtain point cloud projection of each viewpoint with camera parameters $\boldsymbol{P}_{in}$ and $\boldsymbol{P}_{ex}$. An example of the output projection can be seen in Figure (3.3).

At the synthesis stage, the input images are randomly cropped with a 256x256 box at each epoch for training. The pairwise framework is trained with 200 epochs. We set learning rate as 2E-4 and batch size as 1. There are 8 layers in both encoder and decoder of image translation network while 5 layers in discriminative network. All the layers have kernels with a size of 4x4 and strides with a size of 2x2.

At the Refinement stage, We select key points from point sequence by adding an interval value 200 to decrease the overlapping, and obtain about 60 patches for each view. We use PyTorch [PGC+17] to implement our framework. All the input patches are resized to 64x64. Skip connections are used from the four down-sampling layers(layer2, 4, 6, 7) of the encoder to the decoder(layer0, 2, 4, 6). Layers are followed with batch normalization. We set learning rate as 1E-4 and 1E-3 with weight decay, our model is trained and tested separately.

4 Experiments

We evaluate our proposed method on the NYU-Depth V2 data set [SHKF12], which consists of a variety of indoor scenes recorded by Microsoft Kinect. We report our results of synthesis stage and refinement stage by a validation set. We would like to present some visual results and provide a numerical evaluation of images from two stages, such as SSIM [WBSS04] and PSNR.

(a) Samples of NYU-Depth raw data

(b) Samples of basement data set

Figure 4.1: (a) Samples of the RGB image and the raw depth image of different scenes from NYU-Depth data set. (b) Samples of Basement data set. Only RGB images are used in our experiments.

4.1 Datasets

We train the model with the NYU-Depth V2 data set that consists of 464 new scenes taken from three different cities. The data set is comprised of several video sequences and records a variety of indoor scenes. Both RGB and Depth images from the Kinect are sampled from video frames and the sampling rate lies between 20 and 30 FPS. According to our problem setting, we only use the raw RGB images to synthesize realistic novel views in the experiments. We take two of the indoor scenes from the NYU-Depth V2 data set: study room scene with 595 images and basement scene with 1407 images. We train our model on basement data set and test on both. We also divide the basement image set into two different set: 1000 pairs for training and 407 pairs for testing. Figure 4.1 has shown several example images. We describe our data processing for synthesis stage and refinement stage in detail.

Data processing at synthesis stage: For our purpose, at synthesis stage we require pairs of images and their corresponding point cloud projections. In order to create image pairs, we need to estimate camera motion for each image and construct a sparse point cloud at first. Actually, for camera tracking there are few well-performed available tools. We use a structure-from-motion technique in computer vision to estimate camera poses, such as COLMAP [SF16] which is optimized for image collections. We obtain a sparse point cloud with 54,115 points.

Data processing at refinement stage: At refinement stage, we need quadruplet of synthesized patches together with their three candidate patches from nearby cameras. In detail, for each synthesized images, we extract patches by cropping a key point-centered 64x64 bounding box. In order to find corresponding patches in candidate views, we need to find key points that could be visually seen in the target view as well as candidate views. There are in total 54,115 key points in the point cloud, for each quadruplet, about 20,000 points could be seen simultaneously. As described in section 3.2.1, we select key points from seen point sequence by adding an interval value 200 to decrease the overlapping, and obtain about 4x60 patches for each target view in the end. In the experiment, we use 20011 quadruplets of synthesized patches and their candidate patches from 300 images of basement for training, while 6846 quadruplets for testing.

4.2 Baseline methods

Most previous works about novel view synthesis concentrate on predicting novel views from one or multiple given views directly. For instance, DeepStereo [FNPS16] tries to imagine the frame between two static images. Pittaluga *et al.* [PKKS19] present a pointcloud-based method but mainly focus on scene reconstruction and have not shown quantitatively evaluation for novel view synthesis.

According to our problem setting, we aim to generate novel views from arbitrary viewpoint with a sparse set of images. As there is no previous work focus on arbitrary view synthesis for a complicated scene, we take the pix2pix method as our baseline and compare the results with it to validate the effectiveness of our 2-stage training. We evaluate the methods both on views included in the testing set and virtual views set by virtual camera poses.

4.3 Performance comparison

In this section, we present a numerical evaluation of our proposed method and show some visual examples. To obtain a quantitative evaluation, we use two different image quality evaluation metrics PSNR and SSIM [WBSS04]. Both of them compare synthesized images with ground truth images and higher value means higher image quality. We evaluate our proposed method on testing set of basement scene from NYU-Depth V2 Data Set.

Figure 4.2 shows samples of the results from our synthesis network and refinement network. It is obvious without refinement, some regions of the results are strongly affected by artifacts, especially those with abundant details, such as the washing machine in Figure 4.2 (a). What is more, some of the distortions are removed and occlusion relations between objects have been corrected. Table 4.1 shows quantitative comparison with specific PSNR and SSIM values. Comparing with Baseline-Pix2Pix, our 2-stage work presents higher PSNR and SSIM values. It is clear that the refinement network training at patch level does indeed improved the quality of synthesis. To further evaluate the performance of refinement work and validate the effectiveness of guiding patches, we also train our refinement network with another two aggregation methods. As shown in 4.3, our weighted-pooling method with distance between views performs more visually pleasing results than max-pooling and average-pooling.

Method	**Mean SSIM**	**Mean PSNR**
Baseline-Pix2Pix	0.433	18.03dB
Max-pooling	0.559	19.64dB
Average-pooling	0.563	19.96dB
Our weighted-pooling	**0.580**	**19.99dB**

Table 4.1: Quantitative analysis of proposed method: Baseline refers to the novel views produced by pix2pix image-to-image translation method without any refinement. Max-pooling refers to performing aggregation by conducting max-pooling over the features of the three candidate patches. Average-pooling refers to performing aggregation by regarding the three candidate patches as equal. The accuracy is measured by both SSIM and PSNR quality metrics.

Metrics	**0-0.25px** (1838)	**0.25-0.5px** (2280)	**0.5-0.75px** (789)	**0.75-1px** (1513)	**1-1.25px** (354)	**1.25-1.5px** (63)	**>2px** (9)	**Avg.** (6846)
Mean SSIM	**0.587**	0.584	0.576	0.570	0.565	0.566	**0.606**	0.580
Mean PSNR	19.14dB	19.79dB	20.69dB	20.81dB	**21.00dB**	18.47dB	18.06dB	19.99dB

Table 4.2: Quantitative comparison between different image:We divide the test images into 7 different groups by average distance from target view to its three corresponding candidate views. We show number of images in each group and evaluate the image quality for each group with PSNR and SSIM.

Method	**[PKKS19] 20%**	**[PKKS19] 40%**	**[PKKS19] 60%**	**Ours Avg.**
Mean SSIM	0.539	0.605	0.631	0.580

Table 4.3: Quantitative comparisons with [PKKS19], where 20%, 60% and 100% indicates the utilization of SIFT features in their work.

In addition, we also show some detailed quantitative comparison on synthesized novel views of different groups divided by average distance from target view to its corresponding candidate views in Table 4.2. The distance are measured in pixels. Intuitively, the smaller the average distance is, the better final results we should get. Our results generally follow the rule but still have some exception, which we would explain in detail in next section.

We compare against the results from [PKKS19] in Table 4.3. It has shown that our results performs better when they use 20% of the SIFT features in training. Other works by [ZTF+18] and [CGT+19] focus on interpolation or extrapolation from many fewer neighbouring inputs, whose problem setting has larger difference with us.

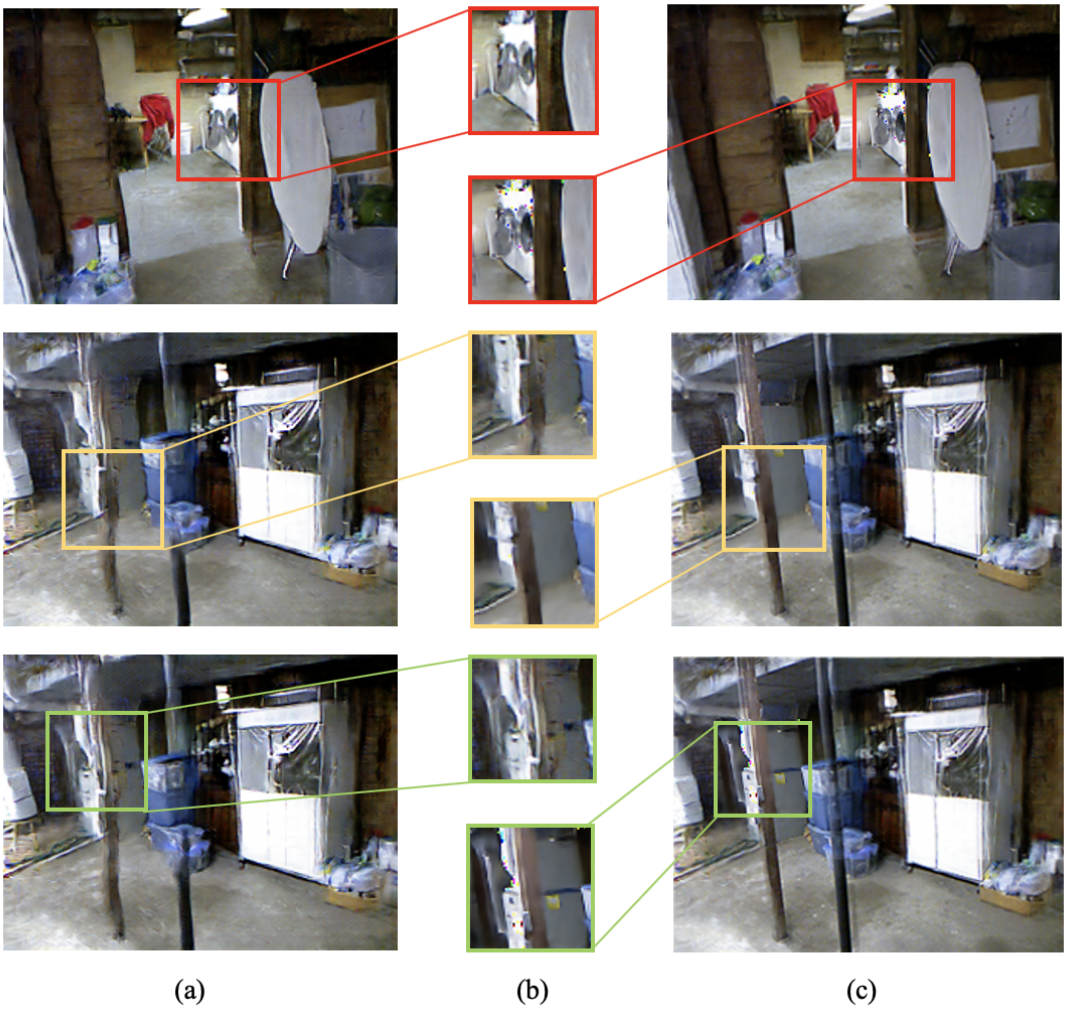
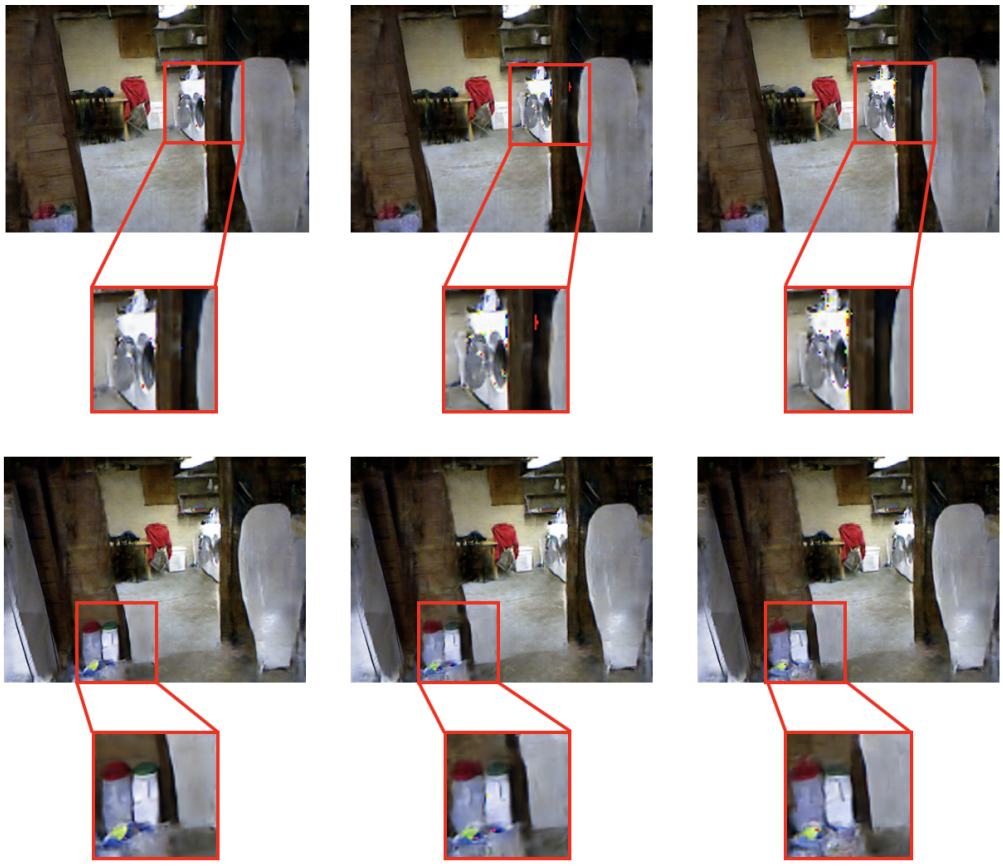


Figure 4.2: Samples of the results from our synthesis network and refinement network. (a) are initial synthesized novel views from synthesis network. (c) are refined novel views generated from refinement network. (b) are selected patches (initial patches and refined patches). It is obvious that some distortions are removed after refinement. Also, the occlusion relation of the blue box and the pillar is corrected in refined patch compared to initial one.



(a) Weighted-pooling (b) Max-pooling (c) Average-pooling

Figure 4.3: Example results generated with different pooling operations. Weighted-pooling method performs more visually pleasing results than other methods by removing distortions and noise.

5 Discussion

5.1 Analysis of experimental results on Basement dataset

We have shown that our proposed 2-stage network has achieved better performance by training with image priors from nearby views in the section of experiments. We test our model on Basement dataset where two different rooms are included.

Obviously, performance of novel view synthesis is strongly affected by transformation of camera pose. It is much more difficult to generate novel views under larger viewpoint changes. We measure the viewpoint changes by average distance in pixel from target view to its three corresponding candidate views. Intuitively, we should get better results when the distance is relatively small. However, as shown in Table 4.2, the group of distance(>2) obtain highest score with SSIM Metric, while the group of distance(1-1.25) get higher value than others with PSNR Metric even if we could see little visual differences on them.

There are two main reasons. First, the experimental results are affected by the data size. For instance, the numerical result of the group with viewpoint distance larger than 2px is not meaningful since the group size is too small. Second, there are still limitations on assessment metrics. Since PSNR is defined via the mean squared error(MSE), it performs better in assessing the quality of noisy images rather than discriminating structural content, while SSIM is more sensitive to structural information. As is shown in Table 4.2, without considering the meaningless data, structural content of novel view is better reconstructed when candidate views are closer, which not means less noise(higher PSNR value) in images.

5.2 Analysis of different pooling skills

To validate the effectiveness of training with candidate patches, we also perform three different aggregation method in our refinement network and get numerical results respectively: Max-pooling, Average-pooling and Weighted-pooling as described in section 3.2.2. Table 4.1 shows quantitative results. Our weighted-pooling operation achieved best performance under both SSIM and PSNR perceptual metrics. Figure 5.1 shows visualization samples generated with different pooling operation.

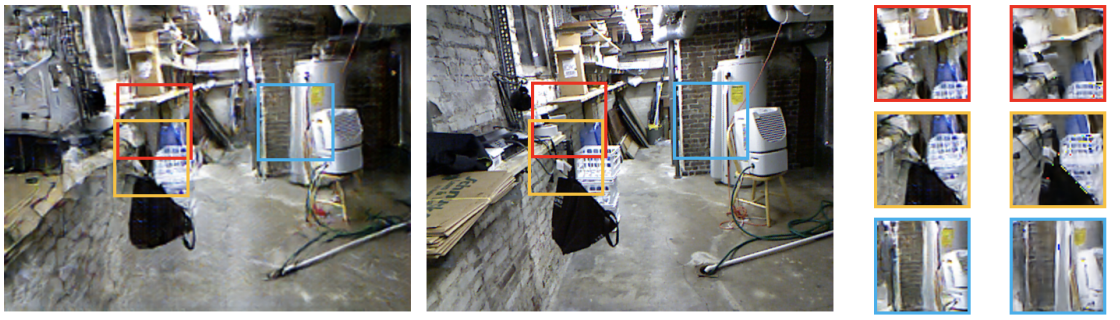
Figure 5.1: Results generated with different pooling operations. It shows that when compare to ground truth, weighted-pooling method suppresses others and performs better in removing noise and preserving detail contents of the washing machine.

Comparing to groundtruth, we could see that structural content has been better preserved and less noise is produced in synthesized result. Therefore, closer view would contribute more to the novel views. Also, it is believed that larger visual difference would be seen if there is an increasing difference between viewpoint distances.

5.3 Limitations

As shown in Figure 5.2 and Figure 5.3, although we have synthesized better results with refinement stage compared to baseline, there is still some noise in final output. The refinement network can mainly fix artifacts such as distortion and structure errors. However, it is struggle to fix natural artifacts and is unable to hallucinate regions where projected points are rarely found.

As our point cloud is generated from a image collection of the scene, it is obvious that there is no point corresponded with pixels in areas outside of input images. We set our novel view by translating or rotating the cameras. Therefore, there might be regions in novel views, where point cloud is extremely sparse or even no point is projected. The synthesis would fail in this kind of cases.



(a) Initial synthesized view (b) Ground truth (c) Initial (d) Refined

Figure 5.2: Samples of limitations. When comparing (c) with (d), we could find that to a certain extent, refined patches have smaller distortions and higher structural similarity with the groundtruth. However, some detail information is lost at the same time so that the image becomes more blurry.

Figure 5.3: Samples of failure cases. Image pairs of point cloud projections (left) and synthesized results (right). **First row**: original viewpoint; **Second row**: virtual viewpoint generated by a translation offset {-2,0,-4}; **Third row**: virtual viewpoint by a translation offset {-2,0,-2}. It is clear that poor reconstruction and strong artifacts are found in areas with fewer projected points.

6 Conclusion and Future work

We propose a new method for novel view synthesis in this thesis and further define the problem setting. We specifically target the case that rendering a scene from a novel viewpoint when given some other views of it. Most of previous works focus on generating novel views by interpolation and extrapolation with input images and the position of virtual camera often differs little from the input. However, there would be missing regions in synthesized views due to occlusion and depth uncertainty. To deal with this problem, existing works use depth-synthesis approach [CDSHD13] or formulating a soft 3D reconstruction explicitly [PZ17]. Recent approaches use deep learning techniques to learn novel views directly by leveraging large image priors. This kind of methods try to correct artifacts at a high level.

Our work focuses on a increasingly common case, so that the challenges lie on two aspects: 1) Find most valuable image priors from large amount of input. 2) Overcome traditional difficulties of novel view synthesis, such as the artifacts caused by occlusion problem and uncertain geometry. To tackle the first challenge, we first build a sparse point cloud with Structure-from-Motion techniques from image collection to get a rough construction for the scene. Key points in the world coordinate could be projected to different image coordinate with different camera poses, forming a geometry representation for each view. Intuitively, views from cameras that close to the target camera should share similar components with target view. Therefore, we use these intermediate representations to measure the distance between cameras and seek out valuable image priors. To solve the second challenge, we propose a two-stage framework consists of synthesis network and refinement network, to predict and refine novel views respectively. We combine traditional geometric method with deep learning techniques and validate on a real scene from NYU-Depth V2 data set. The visualization results and quantitative evaluation have proved the effectiveness of our approach. As a discussion, we showed more visualization results to reveal how patches help correct artifacts.

We also analyzed the limitations of our method and would like to solve those in our future work. First, there still remains some artifacts caused by occlusion. To handle this, we could estimate point visibility at synthesis stage by adding a depth buffer to our network input, and depth value is calculated from the 3D point cloud. However, it is still difficult to perfectly remove all points lie on surfaces which are occluded from that viewpoint, since the points belong to foreground may not exist in the original point cloud. Next, in order to obtain higher quality images with less blur and noises, we could train our refinement network with a perceptual loss rather than

L2 loss to preserve high-frequency information. The second challenge for us is how to fix the regions of strong artifacts and poor reconstruction due to sparse projected points. We would like to further build an end-to-end neural network and train all parameters simultaneously with one loss, and target to achieve better performance of generating more realistic views.

Acknowledgments

First, I would like to express my sincere gratitude to my supervisor Prof. Yoichi Sato for his continuous support of my Master's study and research, for his patience, kindness and immense knowledge, and his enthusiasm is also infectious. During these two years, he offered us plenty of free space and his guidance helped me with my research in all the time. Meanwhile, his thinking breadth, depth and flexibility inspired me a lot.

Besides my supervisor, I am also grateful to our Research Associate Yusuke Matsui. He always inspired me in our discussion whenever I encountered difficulty or came up with a new idea. Also, I would like to thank my tutor Naoya Kato as well, for his help both in study and life when I first arrived in Japan.

I thank all of my fellow lab mates in Sato Lab and all my friends I met here. I have enjoyed a rich and colorful campus life in University of Tokyo while moving forward with them all together in past two years.

At last, I would like to thank my family: my parents Zhijin Wang and Junxi Chai, for giving birth to me and supporting me all the time, and also my roommate Mian Wu and Chenlei Juan, my idol Yibo Wang, my cat Zhuizhui, for their company in my daily life.

References

[Alj17] Aljosa Smolic. Computer vision course. `https://v-sense.scss.tcd.ie/lectures/computer-vision/`, 2017. Online; accessed 8 January 2017.

[BBM+01] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 425–432, 2001.

[CBL99] Chun-Fa Chang, Gary Bishop, and Anselmo Lastra. Ldi tree: A hierarchical representation for image-based rendering. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 291–298, 1999.

[CDSHD13] Gaurav Chaurasia, Sylvain Duchene, Olga Sorkine-Hornung, and George Drettakis. Depth synthesis and local warps for plausible image-based navigation. *ACM Transactions on Graphics (TOG)*, 32(3):1–12, 2013.

[CGT+19] Inchang Choi, Orazio Gallo, Alejandro Troccoli, Min H Kim, and Jan Kautz. Extreme view synthesis. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7781–7790, 2019.

[CW93] Shenchang Eric Chen and Lance Williams. View interpolation for image synthesis. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 279–288, 1993.

[DDB+15] Paul Debevec, Greg Downing, Mark Bolas, Hsuen-Yueh Peng, and Jules Urbach. Spherical light field environment capture for virtual reality using a motorized pan/tilt head and offset camera. In *ACM SIGGRAPH 2015 Posters*, pages 1–1. 2015.

[DLHT14] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *European conference on computer vision*, pages 184–199. Springer, 2014.

[DTM96] Paul E Debevec, Camillo J Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 11–20, 1996.

[DU18] Ugur Demir and Gozde Unal. Patch-based image inpainting with generative adversarial networks. *arXiv preprint arXiv:1803.07422*, 2018.

[FNPS16] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5515–5524, 2016.

[GEB16] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.

[GGSC96] Steven J Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F Cohen. The lumigraph. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 43–54, 1996.

[GPAM+14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[IZZE17] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[JAFF16] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016.

[KB14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[KWR16] Nima Khademi Kalantari, Ting-Chun Wang, and Ravi Ramamoorthi. Learning-based view synthesis for light field cameras. *ACM Transactions on Graphics (TOG)*, 35(6):1–10, 2016.

[LH96] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 31–42, 1996.

[LKM14] Christian Lipski, Felix Klose, and Marcus Magnor. Correspondence and depth-image based rendering a hybrid approach for free-viewpoint video. *IEEE Transactions on Circuits and Systems for Video Technology*, 24(6):942–951, 2014.

[MB95] Leonard McMillan and Gary Bishop. Plenoptic modeling: An image-based rendering system. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 39–46, 1995.

[MGK+19] Moustafa Meshry, Dan B Goldman, Sameh Khamis, Hugues Hoppe, Rohit Pandey, Noah Snavely, and Ricardo Martin-Brualla. Neural rerendering in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6878–6887, 2019.

[MO14] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[NZS+16] Yongwei Nie, Zhensong Zhang, Hanqiu Sun, Tan Su, and Guiqing Li. Homography propagation and optimization for wide-baseline street image interpolation. *IEEE transactions on visualization and computer graphics*, 23(10):2328–2341, 2016.

[PGC+17] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[PKKS19] Francesco Pittaluga, Sanjeev J Koppal, Sing Bing Kang, and Sudipta N Sinha. Revealing scenes by inverting structure from motion reconstructions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 145–154, 2019.

[PZ17] Eric Penner and Li Zhang. Soft 3d reconstruction for view synthesis. *ACM Transactions on Graphics (TOG)*, 36(6):1–11, 2017.

[Rad99] Paul Rademacher. View-dependent geometry. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 439–446, 1999.

[RFB15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[SAC+13] Qi Shan, Riley Adams, Brian Curless, Yasutaka Furukawa, and Steven M Seitz. The visual turing test for scene reconstruction. In *2013 International Conference on 3D Vision-3DV 2013*, pages 25–32. IEEE, 2013.

[SD96] Steven M Seitz and Charles R Dyer. View morphing. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 21–30, 1996.

[SF16] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4104–4113, 2016.

[SGHS98] Jonathan Shade, Steven Gortler, Li-wei He, and Richard Szeliski. Layered depth images. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 231–242, 1998.

[SH99] Heung-Yeung Shum and Li-Wei He. Rendering with concentric mosaics. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 299–306, 1999.

[SHKF12] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *European conference on computer vision*, pages 746–760. Springer, 2012.

[VBK02] Sundar Vedula, Simon Baker, and Takeo Kanade. Spatio-temporal view interpolation. *Rendering Techniques*, 28:65–76, 2002.

[VBR+99] Sundar Vedula, Simon Baker, Peter Rander, Robert Collins, and Takeo Kanade. Three-dimensional scene flow. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 722–729. IEEE, 1999.

[WBSS04] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[XXC12] Junyuan Xie, Linli Xu, and Enhong Chen. Image denoising and inpainting with deep neural networks. In *Advances in neural information processing systems*, pages 341–349, 2012.

[ZKU+04] C Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. *ACM transactions on graphics (TOG)*, 23(3):600–608, 2004.

[ZTF+18] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817*, 2018.