Deep View Synthesis via Self-Consistent Generative Network

Zhuoman Liu, Wei Jia, Ming Yang, Peiyao Luo, Yong Guo, and Mingkui Tan

***Abstract*—View synthesis aims to produce unseen views from a set of views captured by two or more cameras at different positions. This task is non-trivial since it is hard to conduct pixel-level matching among different views. To address this issue, most existing methods seek to exploit the geometric information to match pixels. However, when the distinct cameras have a large baseline (*i.e.*, far away from each other), severe geometry distortion issues would occur and the geometric information may fail to provide useful guidance, resulting in very blurry synthesized images. To address the above issues, in this paper, we propose a novel deep generative model, called Self-Consistent Generative Network (SCGN), which synthesizes novel views from the given input views without explicitly exploiting the geometric information. The proposed SCGN model consists of two main components, *i.e.*, a View Synthesis Network (VSN) and a View Decomposition Network (VDN), both employing an Encoder-Decoder structure. Here, the VDN seeks to reconstruct input views from the synthesized novel view to preserve the consistency of view synthesis. Thanks to VDN, SCGN is able to synthesize novel views without using any geometric rectification before encoding, making it easier for both training and applications. Finally, adversarial loss is introduced to improve the photo-realism of novel views. Both qualitative and quantitative comparisons against several state-of-the-art methods on two benchmark tasks demonstrated the superiority of our approach.**

***Index Terms*—View synthesis, self-consistency, large baseline, generative model.**

I. Introduction

VIEW synthesis generates a novel (absent) camera view image from known camera views of the same scene, as shown in Fig. 1. It can be widely applied in video conferencing [1], virtual reality [2], and free-viewpoint TV [3], *etc*. In this paper, we focus on synthesizing a middle view from two different views in real industrial scenarios where the ideal view is hard to obtain due to hardware limitations. For example, in some video conferencing equipment, cameras are positioned symmetrically on each side of the screen with a large baseline (*i.e.*, the distance between two camera views [4]). Moreover, the baseline between the two cameras is often pre-defined for specific products.

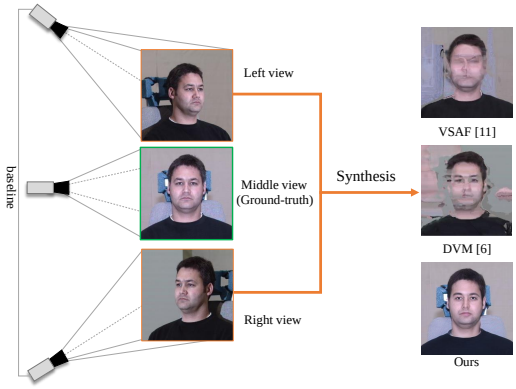


Fig. 1. A view synthesis example: middle view (frontal) synthesis from the left and right views with a large baseline (namely, a large distance between two camera views) on the Multi-PIE [16] dataset. The left and right views are captured by two distinct cameras and then used to synthesize the missing middle view. This task can be applied in a real industrial scenario — a real-time video conferencing system developed by the company with which the authors are working.

The view synthesis task, however, is extremely difficult due to the following challenges: 1) The large distance between two camera views may lead to huge occlusion. The model is hard to synthesize the novel views given limited information. 2) View synthesis is an ill-posed problem. Specifically, there exists an infinite number of middle/novel views that correspond to the same input views [5]. Thus, the space of the possible view synthesis functions can be extremely large, making it hard to find a good solution. Several recent works [6], [7], [8] attempt to solve the view synthesis problem by warping with a depth camera. However, depth images are difficult to obtain due to the limitation of the hardware. Therefore, some geometry-based view synthesis methods [9], [10], [11], [12], [13], [4] are proposed to synthesize novel views without the depth image. Such approaches add geometry constraints to preserve consistency between input views and the synthesized view. However, when the input views have huge occlusion, these geometry-based methods may learn mismatching corresponding map between the input views, or even fail to learn the correspondences. To overcome the drawbacks of geometry-based methods, some image-content-based methods [14], [9], [15] formulate the view synthesis task as a mapping from input views to the target view without geometry constraints. Despite these attempts, the space of the possible view synthesis functions is still extremely large, which makes it difficult to learn a good model.

This work was partially supported by the Science and Technology Program of Guangzhou, China, under Grant 202007030007, the Key-Area Research and Development Program of Guangdong Province (2018B010107001), National Natural Science Foundation of China (NSFC) 61836003 (key project), Guangdong Project 2017ZT07X183, Fundamental Research Funds for the Central Universities D2191240. *(Zhuoman Liu and Wei Jia contributed equally to this work.) (Corresponding author: Mingkui Tan.)*

Z. Liu is with School of Software Engineering, South China University of Technology, Guangzhou 510640, China, and also with CVTE Research, Guangzhou 510530, China, and also with Pazhou Laboratory, Guangzhou 510335, China (E-mail: liuzhuoman@cvte.com).

W. Jia and M. Yang are with CVTE Research, Guangzhou 510530, China (E-mail: jiawei@cvte.com; yangming@cvte.com).

P. Luo and Y. Guo are with School of Software Engineering, South China University of Technology, Guangzhou 510640, China (E-mail: is.luopeiyao@gmail.com; guoyongcs@gmail.com).

M. Tan is with School of Software Engineering, South China University of Technology, Guangzhou 510640, China, and also with Key Laboratory of Big Data and Intelligent Robot, Ministry of Education. (E-mail: mingkuitan@scut.edu.cn).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org., provided by the author. The material includes more experimental results. This material is 1.83MB in size.

To address the above issues, we propose a new view synthesis method, called Self-Consistent Generative Network (SCGN), to simultaneously produce photo-realistic novel views and preserve consistency among different views of the same scene. Specifically, to address the challenge brought by the large baseline between cameras, we propose View Synthesis Network (VSN) model that directly learns the mapping from the input side views to the resultant novel/middle view. To reduce the space of possible mapping functions, we design a self-consistency scheme that introduces an additional constraint by decomposing the synthesized novel view back into the original input side views, and propose a View Decomposition Network (VDN) to learn the decomposition mapping. To further improve the photo-realism of the synthesized view, we incorporate an adversarial loss and an image sharpness loss into the training objective to train the proposed model. Unlike cycle-consistency [17] that helps minimize the distribution divergence, our self-consistency builds a cycle to improve the pixel-wise prediction. With the self-consistency constraint, we are able to effectively reduce the space of possible mapping functions and thus obtain promising views synthesis performance. Extensive experiments on an indoor dataset and an outdoor dataset demonstrate the superiority of the proposed method over existing methods.

Our main contributions are summarized as follows.

- We propose a novel deep View Synthesis Network, called Self-consistent Generative Network (SCGN), which simultaneously synthesizes photo-realistic unseen views and preserves high consistency among different views of the same scene.
- We propose a View Decomposition Network (VDN) that reconstructs the input views from the synthesized view. In this way, different views are highly correlated with each other and the geometric pre-processing (*e.g.*, rectification) in existing methods becomes not necessary.
- Comprehensive experiments demonstrate the superior performance of the proposed method over existing methods both quantitatively and qualitatively. In particular, the proposed method is able to produce visually promising middle views on both the benchmark datasets and the real-world conferencing system[1].

II. Related Work

Multi-view synthesis. Synthesizing a novel view from multiple view images has long been studied. Debevec *et al.* [18] combines both image-based and geometry-based techniques to render novel views from multiple views. Sagonas *et al.* [19] considers frontal facial image synthesizing as an optimization problem. Traditional methods fail in occlusion situations and may generate artifacts in synthesized views. Thus, some approaches that combine different learning methods are proposed to tackle such bottlenecks.

Learning-based approaches tackle multi-view synthesis task via training a prediction model, *e.g.*, Convolutional Neural Networks (CNNs) [14], [10], [20], [21]. Dosovitskiy *et al.* [14] trains CNN to render images of chairs with different poses, lighting, *etc*. DeepStereo [21] synthesizes a novel view by interpolating from neighboring posed views of a scene. However, it is difficult to composite occluding content under large baselines. Similarly, Multi2Novel [13] and StereoMagnification [4] use multi-plane or multi-view to construct a plane-sweep volume. They have the same problem as DeepStereo. Considering the correlation among different views, Zhou *et al.* [10] proposes View Synthesis by Appearance Flow (VSAF) to synthesize new images of the same object from arbitrary viewpoints. However, VSAF requires viewpoint transformation information (in addition to the input images) and may lead to incorrect content due to occlusion. Park *et al.* [22] and Ji *et al.* [12] seek to improve VSAF by addressing these problems. Disocclusion-aware Appearance Flow Network (DOAFN) [22] is proposed to predict not only a novel view but also a visibility map to improve performance. Multi-Scale Adversarial Correlation Matching (MS-ACM) [23] models structures as self-correlation coefficients extracted from multi-scale feature maps. Unfortunately, both DOAFN and MS-ACM are not suitable for the multi-view synthesis task studied in this paper due to the limitation of a single view input. To render a novel view, View Independent Generative Adversarial Network (VI-GAN) [24] and Extreme View Synthesis [25] input additional camera pose, which is not required in our multi-view synthesis task. DVM [12] aims to synthesize novel views from multiple views without additional information beyond input image pairs. Our model outperforms DVM when dealing with a large baseline image pair.

Generative adversarial networks (GANs). Recently, many GANs [26], [27], [28] have been proposed to generate images, such as DCGANs [29], WGANs [30], and progressive GANs [31]. Inspired by GANs, Huang *et al.* [5] proposed a Two-Pathway Generative Adversarial Network (TP-GAN) to synthesize a facial view image from one side view while preserving the symmetric structure of faces. However, TP-GAN ignores data consistency and may result in meaningless images. Regarding this issue, Zhu *et al.* proposed cycle-consistency loss to preserve the content in image translation by enforcing double-sided consistency during training [17], [32]. Better than Pix2Pix [33], the double forward-backward processes qualify it for unsupervised tasks. However, unlike image translation, the multi-view synthesis task is often a supervised task in which we should exactly recover a novel view from two or more view images (or videos) obtained by cameras at distinct positions with more strict constraints. Focusing on addressing the multi-view synthesis task, in this paper, we use self-consistency (one cycle mapping from the synthesized view to side views) to ensure the input views (which may contain occluded contents) can be reconstructed.

III. Self-Consistent Generative Network

With the goal of addressing the challenges in the view synthesis task, *i.e.*, synthesizing novel views under large baselines with more occluded areas in paired views, and avoiding the limitations of geometric modules under large baselines, we propose self-consistency in our model.

[1]The collected dataset of the real-world conferencing is available at https://zhuomanliu.github.io/datasets/download.html.

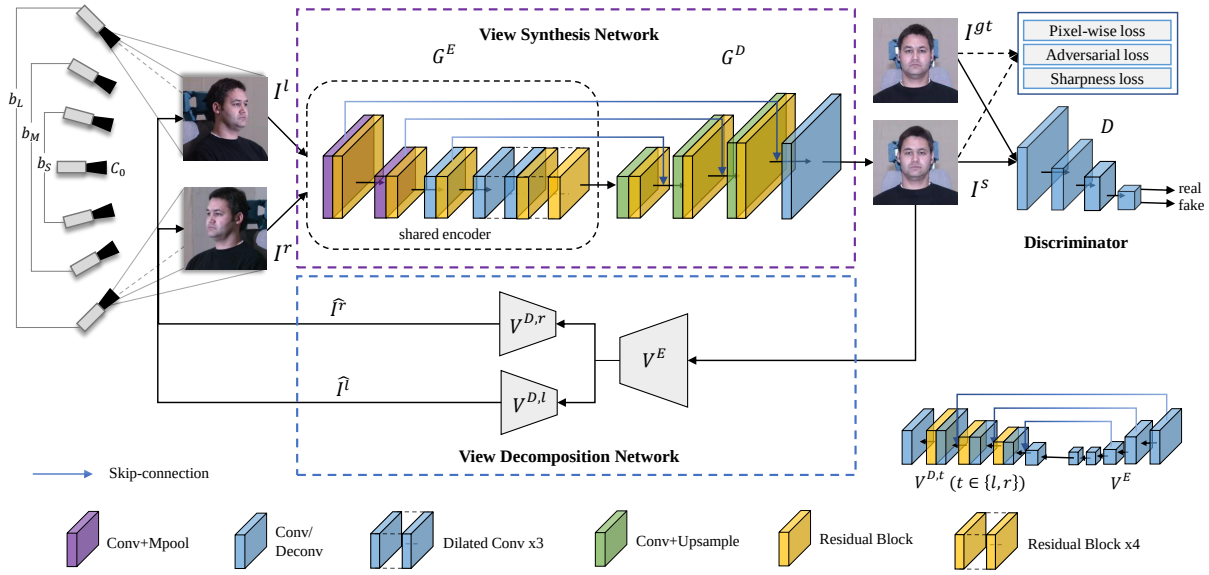


Fig. 2. General structure of Self-Consistent Generative Network (SCGN) for the task of **frontal view synthesis**. Here, camera C_0 is the reference camera that produces the ground-truth view image I^{gt}. Moreover, the connected pairs of cameras capture symmetrical views with different baselines, where b_S, b_M and b_L are view pairs with small, medium, and large baselines, respectively. SCGN seeks to recover the middle view from a given side view pair (I^l, I^r). SCGN consists of two major components, namely the encoder-decoder-based generator $\{G^E, G^D\}$, and the view decomposition network which enforces consistency between input views and recovered views. More network details can be found in Section III-D.

Without loss of generality, we introduce our method by focusing on synthesizing the absent frontal view from two stereo views, as shown in Fig. 1. Given a set of stereo view triplets $\{(I_i^l, I_i^r, I_i^{gt})\}_{i=1}^n$, we seek to learn a mapping $G : (I^l, I^r) \rightarrow I^{gt}$ to recover the ground-truth view I^{gt} from a given input view pair (I^l, I^r). This task is non-trivial due to the view correspondence issue of distinct views.

In this paper, we present a novel view synthesis method, called Self-Consistent Generative Network (SCGN). As shown in Fig. 2, our proposed method consists of two parts, namely a *View Synthesis Network* (VSN) for generating a frontal view from two stereo views, and a *View Decomposition Network (VDN)* for attempting to reconstruct two input stereo views from the synthesized frontal view. Here, VDN helps to address the occlusion problem caused by a large baseline. Furthermore, to ensure the photo-realism of the synthesized views, we further introduce a GAN based loss to train the model rather than the simple pixel-wise loss. The details of each part will be described in the following sections.

A. View Synthesis Network

As shown in the purple dotted block of Fig. 2, we employ an encoder-decoder network to implement the view synthesis network G, consisting of an encoder G^E and a decoder G^D. Both the encoder and the decoder networks are composed of a stack of residual blocks [34], allowing for faster convergence and better performance. To improve the representation ability of the embedding, in the encoder, we replace the last three convolutional layers with dilated convolution [35], [36] to increase the receptive field of the filters without increasing the number of weights.

Given the left and right views I^l and I^r, the synthesized view, denoted by I^s, can be computed by

$$I^s = G^D\left(G^E(I^l, I^r)\right), \tag{1}$$

where $G^E(I^l, I^r)$ denotes feature extraction from I^l and I^r using a weight-shared encoder. Specifically, we first use the encoder model to extract features from I^l and I^r independently. Then, we concatenate the features of I^l and I^r as the output of $G^E(I^l, I^r)$.

Reconstruction loss. To exactly recover the frontal view image, it is straightforward to use a pixel-wise loss to minimize the distance between the synthesized view I^s and the ground-truth I^{gt} on pixel level:

$$L_p(\theta_G) = \frac{1}{n}\sum_{i=1}^{n} \|I_i^s - I_i^{gt}\|_1, \tag{2}$$

where n denotes the number of images, $\|\cdot\|_1$ denotes ℓ_1-norm.

Sharpness loss. To improve the quality of the synthesized images, we integrate an image sharpness method into the loss function. First, to measure the sharpness of images, we exploit the sharpness criterion Q_S in LOGS [37] by computing the differences of the textural complexity between the synthesized image and its reblurred version obtained by a Gaussian smoothing filter. The textural complexity can be represented by the standard deviations of the pixels in the image. Following [37], we compute Q_S in a block-wise manner:

$$Q_S(I) = \frac{\sum_{i=1}^{i=Z} \sqrt{|\sigma_{1i}^2 - \sigma_{2i}^2|}}{Z}, \tag{3}$$

where σ_{1i}^2 and σ_{2i}^2 represent the standard deviations of the i-th block in the image and its blurred version. Here, $Z = \lfloor \frac{M}{k} \rfloor \cdot \lfloor \frac{N}{k} \rfloor$ denotes the total number of blocks, where k denotes

Algorithm 1 Training algorithm for SCGN

Require: Training stereo view triplets $\{I_i^l, I_i^r, I_i^{gt}\}_{i=1}^n$; batch size m; number of training iterations T; learning rate α.

1: **for** $t = 1, ..., T$ **do**
2: Sample a mini-batch of views $\{I_i^l, I_i^r, I_i^{gt}\}_{i=1}^m$.
3: Synthesize the frontal view I_i^s using Eq. (1).
4: Synthesize view pair $(\hat{I}_i^l, \hat{I}_i^r)$ from I_i^s using Eq. (9).
5: Update discriminator parameters θ_D using Eq. (11).
6: Update VSN parameters θ_G using Eq. (13).
7: Update VDN parameters θ_V using Eq. (14).
8: **end for**

the block size, M and N denote the height and width of the image I, respectively. Then, we construct the sharpness loss L_{sharp} based on Q_S:

$$L_{sharp} = \frac{1}{n}\sum_{i=1}^{n} \|Q_S(I_i^{gt}) - Q_S(I_i^s)\|_1, \tag{4}$$

where I^s denotes the synthesized view and I^{gt} denotes the ground-truth view.

Adversarial loss. To improve the photo-realism of the synthesized views, we propose to train the network in an adversarial manner. Specifically, VSN can be regarded as a generator G for synthesizing a frontal view I^s that is as photo-realistic as the real one I^{gt}. To enable VSN to synthesize a good-quality frontal view, we also introduce discriminator D to distinguish the generated frontal view from a real frontal view.

Let θ_G and θ_D be the model parameters of the generator G and the discriminator D, respectively. Following [26], the adversarial network can be trained by solving the following minimax problem:

$$\min_{\theta_G}\max_{\theta_D} L_{gen}(\theta_G, \theta_D), \tag{5}$$

with $L_{gen}(\theta_G, \theta_D)$ being

$$\begin{aligned} L_{gen}(\theta_G, \theta_D) = &\mathbb{E}_{I^{gt}\sim P_{I^{gt}}}[\log D(I^{gt})] \\ &+ \mathbb{E}_{I^s\sim P_{I^s}}[\log(1 - D(I^s))], \end{aligned} \tag{6}$$

where $P_{I^{gt}}$ and P_{I^s} are the distributions of the ground-truth and synthesized image, respectively.

In the training, the discriminator D can be learned by minimizing the following loss:

$$L_{disc}(\theta_D) = -\frac{1}{n}\sum_{i=1}^{n} \log D(I_i^{gt}) - \log(1 - D(I_i^s)), \tag{7}$$

where $D(I^s)$ is the probability that a synthesized image is a real frontal view. For better gradient behavior, we minimize $-\log D(I^s)$ instead of $\log(1 - D(I^s))$ [26]. For the generator G, we can define the **adversarial loss** as follows:

$$L_{adv}(\theta_G) = -\frac{1}{n}\sum_{i=1}^{n} \log D\left(G^D\left(G^E(I_i^l, I_i^r)\right)\right). \tag{8}$$

During training, the adversarial loss will be combined with other losses to update the parameters θ_G of generator G.

B. View Decomposition Network

For the ill-posed problem that there exists an infinite number of middle/novel views that correspond to the same input views [5], we propose a self-consistency scheme to reduce the space of possible view synthesis functions. Specifically, we propose a View Decomposition Network (VDN) that reconstructs the input side views from the predicted middle/novel view, as shown in the blue block of Fig. 2.

The VDN consists of an encoder V^E and two separate decoders V^D for the two side views. Specifically, VDN introduces a decomposition mechanism to decompose the generated frontal image I^s from VSN backward into $(\hat{I}^l, \hat{I}^r)$, *i.e.*,

$$\hat{I}^l = V^{D,l}(V^E(I^s)) \quad \text{and} \quad \hat{I}^r = V^{D,r}(V^E(I^s)). \tag{9}$$

In addition, by regenerating the side views, VDN here can ensure the validity of the generated occluded area. This is the reason why the VSN in our model does not need to contain a rectification module or any transformation operations. In combination with the forward generation network (*i.e.*, VSN) that learns the translation from (I^l, I^r) to I^s, VDN can backtrack to the original source and enforce forward-backward constraints on input view pairs. The predicted left and right views $(\hat{I}^l, \hat{I}^r)$ should be close to the real left and right input (I^l, I^r). We, therefore, minimize the distance between $(\hat{I}^l, \hat{I}^r)$ and (I^l, I^r) through L_{vc}:

$$L_{vc}(\theta_G, \theta_V) = \frac{1}{n}\sum_{i=1}^{n} \|\hat{I}_i^l - I_i^l\|_1 + \|\hat{I}_i^r - I_i^r\|_1. \tag{10}$$

Noted that the proposed self-consistency has several difference with cycle-consistency [17]. Firstly, cycle-consistency uses cycles to help minimize distribution divergence without ground truth while our self-consistency builds a cycle to improve the pixel-wise prediction together with reconstruction loss. Secondly, cycle-consistency learns two symmetric mappings between the images in two domains while our self-consistency learns two asymmetric mappings, *i.e.*, a synthesis mapping and a decomposition mapping. In practice, the proposed self-consistency scheme is able to significantly improve the performance by incorporating the constraint w.r.t. the decomposition mapping (See Table IV, Table V, and Table VI in the paper).

C. Training Details

To train SCGN, we need to update the parameters θ_G for VSN, θ_V for VDN and θ_D for discriminator D. Following GANs [26], we adopt an alternating optimization scheme to train SCGN using mini-batch stochastic gradient descent (SGD), as shown in Algorithm 1. Let α be the learning rate for SGD. In each iteration, for discriminator D, we update θ_D by minimizing the loss L_{disc} according to

$$\theta_D = \theta_D - \alpha\nabla_{\theta_D} L_{disc}. \tag{11}$$

For VSN, we should update θ_G by minimizing the following loss function:

$$L_G(\theta_G) = L_p + \lambda_1 L_{vc} + \lambda_2 L_{adv} + \lambda_3 L_{sharp}, \tag{12}$$

where λ_1, λ_2, and λ_3 are balancing parameters. Specifically, we consider the loss function with the L_{sharp} term as a variant of the loss function without the L_{sharp}. We further discuss this variant in Section V-D. Thus, the update can be made by

$$\theta_G = \theta_G - \alpha \nabla_{\theta_G} L_G. \tag{13}$$

Last, for VDN, we update θ_V by minimizing the loss L_{vc} according to

$$\theta_V = \theta_V - \alpha \nabla_{\theta_V} L_{vc}. \tag{14}$$

D. Details of the Network Structure

1) View synthesis network: We build the view synthesis network G following the scheme of an encoder-decoder network. The details are shown in Table I.

Encoder. The weight-shared encoder G^E takes the left and right views as inputs respectively. For the encoder G^E, each convolutional layer is followed by a leaky rectified linear unit (leaky ReLU). To better leverage spatial information and large distance information, dilated convolution is also employed in the encoder. Then, the encoded features of the left and right views $\{ec6\_l, ec6\_r\}$ are concatenated along the channel dimension and taken as the inputs of the decoder G^D. We also introduce residual blocks to our model. Specifically, each max-pooling layer is followed by one residual block and the final layer of the encoder (*i.e.*, ec6) is followed by four residual blocks.

Decoder. For the decoder G^D, we adopt ReLU as the non-linear activation function after each convolutional layer except for the final convolution layer $dc5$. In layer $dc5$, tanh is adopted to keep the output within the normalized data range. We obtain $\{ecfeatk\_l, ecfeatk\_r\}$ ($k \in \{1, 2, 3\}$) by applying 1×1 kernels to the outputs of $\{eck\_l, eck\_r\}$ ($k \in \{1, 2, 3\}$), respectively. We insert skip-connection between the encoder and the decoder and obtain the input of the next layer by concatenating $\{ecfeatk\_l, ecfeatk\_r\}$ ($k \in \{1, 2, 3\}$) with upk ($k \in \{1, 2, 3\}$).

2) View decomposition network: Unlike the view synthesis network G, the view decomposition network V is a fully convolutional network with the structure shown in Table II.

It is noteworthy that, we obtain $\{dec5\_l, dec5\_r\}$ by applying two 1×1 kernels to the output of $dec5$ in the encoder V^E. Then, the two decoders individually process $\{dec5\_l, dec5\_r\}$ to acquire their decomposed views. Leaky ReLU, residual blocks, and skip-connections are introduced in this network, similar to the view synthesis network, to ensure the effectiveness of our model.

3) Discriminator network: We show the detailed structure of the discriminator network D in Table III. Each convolutional layer is followed by a leaky ReLU. The fully connected layer on top of the convolutional layers is used to estimate the probability that the middle view I^s or I^{gt} is real.

TABLE I
DETAILED STRUCTURE OF THE VIEW SYNTHESIS NETWORK. THE LAYER TYPES "CONV", "MAXPOOL", "DCONV", AND "UPSAMPLE" REPRESENT "CONVOLUTION", "MAX-POOLING", "DILATED CONVOLUTION" AND "UPSAMPLING" RESPECTIVELY. k DENOTES THE KERNEL SIZE, s IS THE STRIDE OF THE LAYER AND r DENOTES THE DILATION RATE OF DILATED CONVOLUTION. THE DEFAULT INPUT OF EACH LAYER IS THE OUTPUT OF THE PREVIOUS LAYER, EXCEPT FOR THOSE LAYERS SPECIFIED BY THE COLUMN "INPUT".

Shared Encoder					
Layer	Type	k	s	r	Output Size
ec1	conv	7	1	-	224×224×32
ep1	maxpool	3	2	-	112×112×32
ec2	conv	5	1	-	112×112×64
ep2	maxpool	3	2	-	56×56×64
ec3	conv	3	1	-	56×56×128
ec4	dconv	3	1	2	56×56×128
ec5	dconv	3	1	2	56×56×128
ec6	dconv	3	1	2	56×56×128

Decoder					
Layer	Type	Input	k	s	Output Size
dc1	conv	ec6_l, ec6_r	3	1	56×56×128
up1	upsample	dc1	-	-	56×56×128
dc2	conv	up1, ecfeat3_l, ecfeat3_r	3	1	56×56×64
up2	upsample	dc2	-	-	112×112×64
dc3	conv	up2, ecfeat2_l, ecfeat2_r	5	1	112×112×32
up3	upsample	dc3	-	-	224×224×32
dc4	conv	up3, ecfeat1_l, ecfeat1_r	7	1	224×224×32
dc5	conv	dc4	3	1	224×224×3

TABLE II
DETAILED STRUCTURE OF THE VIEW DECOMPOSITION NETWORK. "DECONV" REPRESENTS THE DECONVOLUTION LAYER.

Encoder					**Decoder**				
Layer	Type	k	s	Output Size	Layer	Type	k	s	Output Size
dec1	conv	7	2	112×112×16	ddc1	deconv	3	2	28×28×128
dec2	conv	5	2	56×56×32	ddc2	deconv	3	2	56×56×64
dec3	conv	3	2	28×28×64	ddc3	deconv	5	2	112×112×32
dec4	conv	3	2	14×14×128	ddc4	deconv	7	2	224×224×16
dec5	conv	3	1	14×14×256	ddc5	deconv	3	1	224×224×3

TABLE III
DETAILED STRUCTURE OF THE DISCRIMINATOR NETWORK.

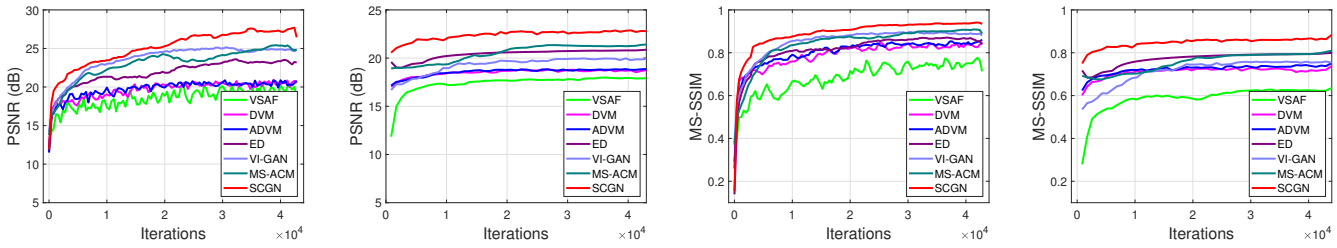
Layer	Type	k	s	Output Size
disc1	conv	5	2	112×112×32
disc2	conv	5	2	56×56×64
disc3	conv	5	2	28×28×128
disc4	conv	5	2	14×14×256
fc5	fc	-	-	1

IV. EXPERIMENTS

To demonstrate the effectiveness and robustness of the proposed method, we compare SCGN with several state-of-the-art methods in both indoor and outdoor scene synthesis settings. Specifically, we conduct multi-view synthesis experiments on Multi-PIE [16] and KITTI [38] datasets for the indoor and outdoor scene synthesis tasks, respectively.

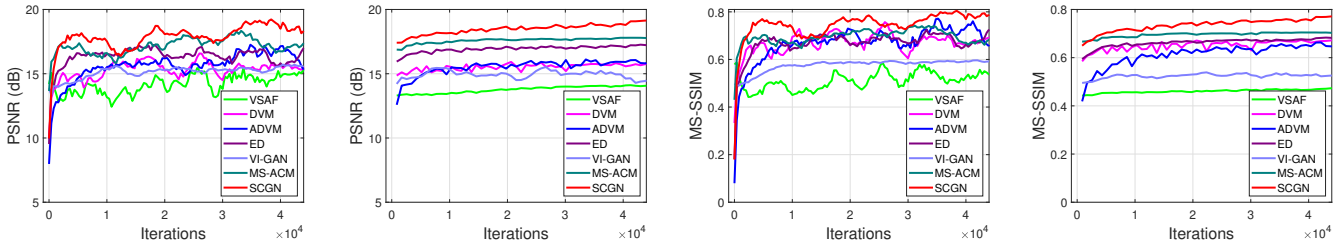
We also apply our method in real conferencing systems[2]. We collect frames (roughly $5K$ triplets) containing multiple human subjects with their upper bodies in a conferencing scenario. We use 80% for fine-tuning and 20% for testing. The data for fine-tuning and testing share the same backgrounds. The same subject with different clothing or motions may appear in different frames. Thus, there is no overlap in image level data. More details of the data collection are described in Section IV-G1.

[2]The demos and the implementation of the proposed SCGN are available at https://github.com/zhuomanliu/SCGN.



(a) Training PSNR (b) Testing PSNR (c) Training MS-SSIM (d) Testing MS-SSIM

Fig. 3. Performance comparison of different view synthesis methods on Multi-PIE measured by PSNR and MS-SSIM metrics. Both the training and testing curves are reported in this figure.



(a) Training PSNR (b) Testing PSNR (c) Training MS-SSIM (d) Testing MS-SSIM

Fig. 4. Performance comparison of different view synthesis methods on KITTI measured by PSNR and MS-SSIM metrics. Both the training and testing curves are reported in this figure.

A. Implementation Details and Datasets

For convenience, we use the same experimental settings for experiments on both Multi-PIE and KITTI datasets. Specifically, for the optimization, we use Adam [39] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ and normalize all images to the range $(-1, 1)$ to train the model while set $\lambda_1 = 0.01$, $\lambda_2 = 0.001$ and $\lambda_3 = 0.01$ to balance the losses in Eq.(12). We train the proposed SCGN for $371,400$ iterations with a batch size of 1. The learning rate is set to 1×10^{-4} and 1×10^{-5} for generator and discriminator, respectively, and decays by 0.1 at iteration $185,700$. During training, the original images are center cropped to $min(H, W) \times min(H, W)$, where H and W indicate the height and width of the original image, respectively. Then, we resize it to 224×224. We implement our proposed method on TensorFlow [40] and conduct all experiments on a single Nvidia TitanX GPU.

Multi-PIE. The indoor dataset Multi-PIE contains more than $750,000$ images of 337 people under 15 symmetrical viewpoints. For all of the experiments on Multi-PIE in this paper, we divide the dataset into 270 people for training and 67 people for testing, and make image triplets that include the frontal view (*i.e.*, the 0° pose captured by the central camera [3]) and two symmetrical side views (from 6 different baselines $\{b_S = \pm 15^\circ, b_M = \pm 30^\circ, b_L = \pm 45^\circ, b_{60} = \pm 60^\circ, b_{75} = \pm 75^\circ, b_{90} = \pm 90^\circ\}$) for training and evaluation. Furthermore, the cropped image triplets include not only the facial region but also the complex background.

KITTI. The outdoor dataset KITTI provides 22 odometry and image sequences of urban city scenes. Our experimental settings on KITTI are the same to the standard setting of view synthesis and have been widely used in view synthesis methods [10], [21], [13], [41]. The KITTI [38] dataset contains the frame sequences captured by the camera on a car traveling through urban city scenes. When we select two frames from a sequence, they can be seen as two views captured by the cameras at different positions. Actually, it is consistent with the standard setting of view synthesis that we seek to produce a novel view from multiple views captured by the cameras at different positions. To demonstrate the robustness and superiority of SCGN under the scenes with complex backgrounds, we conduct experiments on this dataset. According to the settings of VSAF [10] on KITTI, we first randomly sample a frame as ground truth and then select two symmetric frames within the sequence that are separated by $\pm K$ frames as the input image pair, where k is randomly sampled from the set $\{1, 2, ..., 7\}$. We split the first 11 sequences into 9 for training and 2 for testing and randomly collect paired frames with different baselines.

B. Comparing Methods

On Multi-PIE and KITTI, we compare SCGN with several state-of-the-art methods, including View Synthesis by Appearance Flow (VSAF) [10] , Deep View Morphing (DVM) [12], View Independent Generative Adversarial Network (VI-GAN) [24], Multi-Scale Adversarial Correlation Matching (MS-ACM) [15], and Extreme View Synthesis (EVS) [42]. Since the source code of DVM, MS-ACM and VI-GAN are not public, we reimplement DVM and MS-ACM on TensorFlow [40] and VI-GAN on PyTorch [43].

We also consider the widely used image-content-based method Encoder-Decoder (ED) [12] in the comparisons. ED is inspired by the Encoder-Decoder Network (EDN) in DVM.

[3] http://www.cs.cmu.edu/afs/cs/project/PIE/MultiPie.

We employ the ED as a baseline to evaluate the content-based method. To this end, we modify the outputs and architecture of the decoders in the EDN to have the same input and output settings as our method. We also extend the adversarial training scheme to DVM [12] and obtain its GAN-based variant, called adversarial DVM (ADVM).

Furthermore, to demonstrate the excellent performance of our method, we further compare SCGN with some generative methods, *e.g.*, Pix2Pix [33] and CycleGAN [17], and geometry-based methods that leverage known camera poses or plane-sweep volumes, *e.g.*, Multi2Novel [13] and Stereo-Magnification [4], on KITTI.

C. Evaluation Metrics

For quantitative comparisons, we adopt the *Peak Signal-to-Noise Ratio* (PSNR), *Multi-Scale Structural Similarity* (MS-SSIM) [31], and *Inception Score* (IS) [44] as performance metrics. The PSNR measures the amount of signal loss w.r.t. a reference and the MS-SSIM measures the similarity between the generated images and the reference images. The inception score measures both the single image quality and the diversity over a large number of samples. For all the above metrics, the larger the metric value is, the better the performance of the method is. We also adopt *the mean of MSE* (mMSE) (used in DVM [12]) and L_1 *error* (used in VSAF [10]) for fair comparisons. For these two metrics, the smaller the metric value is, the better the performance of the method is.

Furthermore, several view synthesis quality assessment methods [45], [46], [47], [48] have been shown very effective for view synthesis quality evaluation. Specifically, we compare the performances of our SCGN model with the considered methods in terms of LOGS [37], which is a view synthesis quality metric. For the LOGS, the higher metric value indicates better quality.

D. Training Convergence

In this experiment, we compare the training and testing convergence of different methods on both Multi-PIE and KITTI datasets in terms of PSNR and MS-SSIM. The experimental results of the indoor and outdoor datasets are shown in Fig. 3 and Fig. 4, where (a), (b) and (c), (d) show the convergence results in terms of PSNR and MS-SSIM, respectively, during both training and testing periods.

From Fig. 3 and Fig. 4, our SCGN shows faster convergence than other methods in terms of both PSNR and MS-SSIM. As for the testing performance, our SCGN consistently outperforms other methods during the whole training procedure. These results demonstrate the superior performance of the proposed method over competing approaches.

E. View Synthesis Results on Indoor Scenes

We compare our SCGN with state-of-the-arts on Multi-PIE and report the results in Table IV. Noted that the results on Multi-PIE for EVS are unavailable because of the lack of camera pose in the Multi-PIE. From Table IV, we observe that, first, SCGN consistently outperforms other methods on all evaluation metrics under different baseline settings. Our model obtains the highest PNSR and inception score, suggesting that the synthesized results are more photo-realistic. Moreover, high MS-SSIM values and high LOGS values with low error values (*e.g.*, mMSE, L_1 *error*) show that SCGN generates accurate results. Second, SCGN significantly outperforms the considered methods, especially when the baseline is very large, *i.e.*, b_M, b_L, b_{60}, b_{75}, and b_{90}. In other words, our method is able to effectively predict the frontal view from two distant side views. In contrast, geometry-based methods (*e.g.*, VSAF, DVM, ADVM, EVS) obtain poor metric values and fail to synthesize high-quality frontal views under large baselines. Furthermore, ED performs better than these geometry-based methods, which shows that view synthesis under large baselines benefits from methods based on image content.

TABLE IV
PERFORMANCE COMPARISON WITH STATE-OF-THE-ART VIEW SYNTHESIS METHODS UNDER THREE BASELINES b_S, b_M, b_L ON MULTI-PIE.

Baseline $\mathbf{b_S}$						
Method	PSNR	MS-SSIM	Inception Score	mMSE	L_1	LOGS
VSAF [10]	19.56	0.7199	-	134.50	0.171	0.1979
DVM [12]	21.26	0.8686	-	131.67	0.123	0.2581
ADVM	20.89	0.8587	1.68±0.12	131.76	0.125	0.2544
ED [12]	23.30	0.8940	-	130.63	0.083	0.2505
VI-GAN [24]	21.22	0.7910	1.72±0.19	131.87	0.101	0.2436
MS-ACM [15]	25.16	0.9096	1.62±0.38	129.23	0.064	0.2501
SCGN (ours)	**26.36**	**0.9620**	**1.97±0.22**	**128.95**	**0.054**	**0.2719**
Baseline $\mathbf{b_M}$						
Method	PSNR	MS-SSIM	Inception Score	mMSE	L_1	LOGS
VSAF [10]	17.35	0.6273	-	136.39	0.191	0.1799
DVM [12]	18.39	0.6744	-	135.52	0.180	0.2069
ADVM	18.25	0.7010	1.67±0.16	135.37	0.185	0.1746
ED [12]	21.07	0.8270	-	132.54	0.103	0.2278
VI-GAN [24]	20.35	0.7565	1.72±0.17	132.82	0.111	0.2416
MS-ACM [15]	22.51	0.8464	1.66±0.41	130.71	0.086	0.2182
SCGN (ours)	**22.83**	**0.8578**	**1.97±0.22**	**130.40**	**0.076**	**0.2459**
Baseline $\mathbf{b_L}$						
Method	PSNR	MS-SSIM	Inception Score	mMSE	L_1	LOGS
VSAF [10]	16.27	0.5257	-	139.01	0.215	0.1860
DVM [12]	16.77	0.6431	-	139.12	0.224	0.2144
ADVM	17.09	0.6436	1.67±0.22	138.94	0.223	0.2034
ED [12]	20.51	0.7741	-	133.30	0.112	0.2229
VI-GAN [24]	19.66	0.7277	1.77±0.27	133.92	0.121	0.2346
MS-ACM [15]	21.57	0.8174	1.66±0.39	131.50	0.096	0.2177
SCGN (ours)	**21.75**	**0.8268**	**1.91±0.24**	**131.37**	**0.087**	**0.2407**

From the visual comparison results in Fig. 5, VSAF and DVM struggle to produce plausible frontal views when increasing the baseline from small to large. For the modified version ADVM with an additional adversarial loss, the image quality still suffers due to the limitations of 2D geometric approximation. The Encoder-Decoder (ED) exhibits good performance when given a small baseline but fails to produce plausible images under a large baseline. In contrast, SCGN is able to recover photo-realistic frontal views when given different baselines. We also apply our method to the view synthesis tasks on the Multi-PIE dataset with three large baselines, *i.e.*, b_{60}, b_{75}, and b_{90}. We show the results in Fig. 7 and Table V. From these results, our SCGN significantly outperforms the considered methods. From Fig. 15, our method effectively preserves the consistency among different views.

To show the effectiveness of our method for synthesizing photo-realistic frontal views, we exhibit the detailed structure and texture of the results produced by different methods in Fig. 6. Clearly, SCGN is able to produce quality images with sharper face structures and finer details in the background.

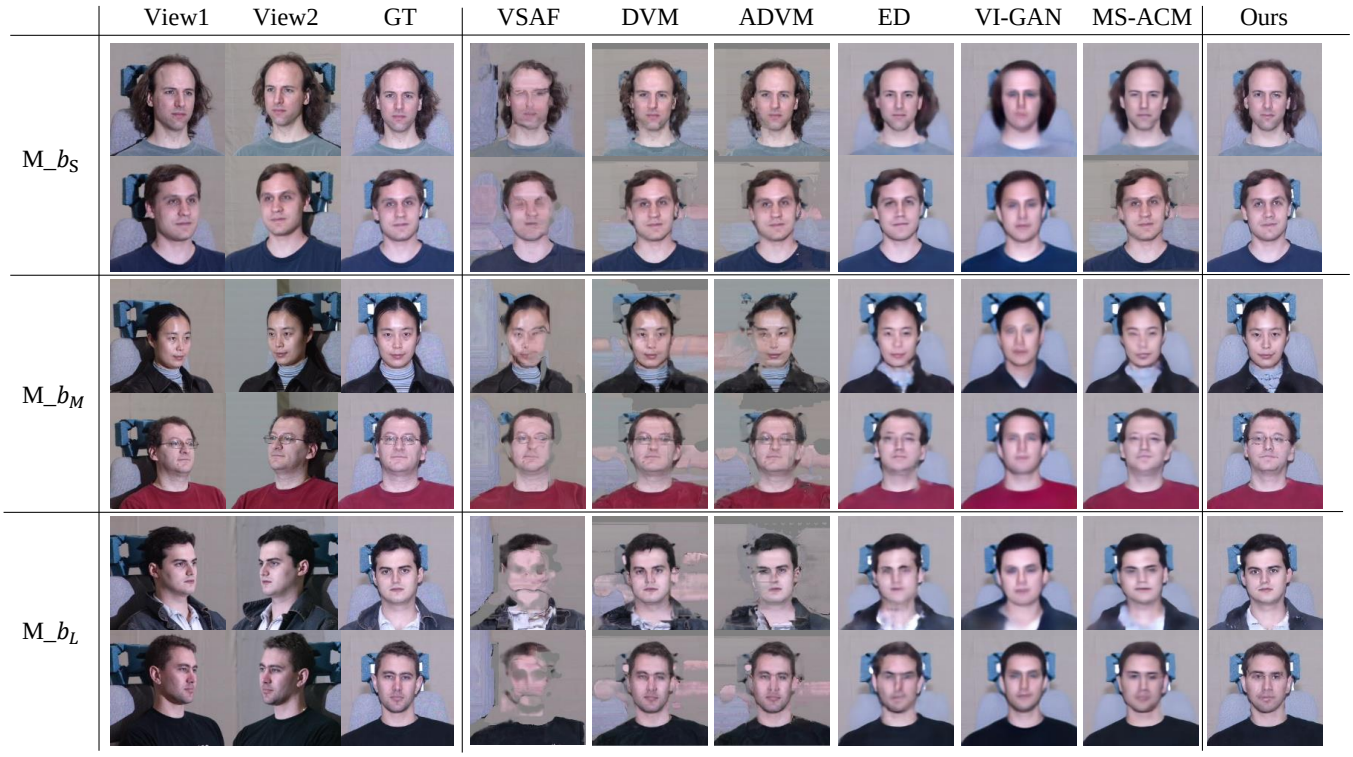


Fig. 5. Visual comparisons of the synthesized view images under different baselines, where M represents Multi-PIE and $\{b_S, b_M, b_L\}$ are three different baselines. For Multi-PIE, {View1, View2, GT} represent {left, right, middle} views.

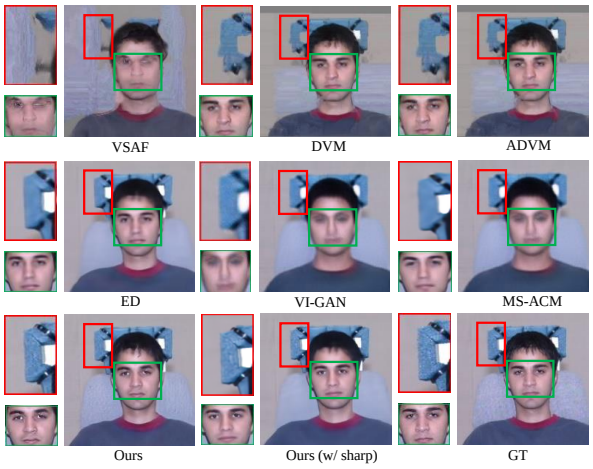


Fig. 6. Visual comparison of image details generated by different methods on Multi-PIE. The red boxes and the green boxes emphasize the local details of background and face, respectively.

TABLE V
PERFORMANCE COMPARISON WITH STATE-OF-THE-ART VIEW SYNTHESIS METHODS UNDER THREE BASELINES b_{60}, b_{75}, b_{90} ON MULTI-PIE.

Method	PSNR	MS-SSIM	Inception Score	mMSE	L_1	LOGS
Baseline b_{60}						
VSAF [10]	10.18	0.0599	-	172.78	0.405	0.2005
DVM [12]	18.47	0.6985	-	135.67	0.169	0.2220
ADVM	16.33	0.5990	1.66±0.22	140.58	0.245	0.1620
ED [12]	19.73	0.7362	-	134.50	0.126	0.2107
VI-GAN [24]	19.25	0.6973	1.71±0.23	134.41	0.130	0.1965
MS-ACM [15]	**20.85**	0.7780	1.52±0.20	**132.29**	0.104	0.1997
SCGN (ours)	20.84	**0.7929**	**1.84±0.33**	132.32	**0.102**	**0.2443**
Baseline b_{75}						
Method	PSNR	MS-SSIM	Inception Score	mMSE	L_1	LOGS
VSAF [10]	15.97	0.5563	-	139.38	0.201	0.1763
DVM [12]	15.48	0.4513	-	143.32	0.267	0.1817
ADVM	17.17	0.5876	1.78±0.21	138.50	0.182	0.2191
ED [12]	18.22	0.6602	-	137.32	0.152	0.2058
VI-GAN [24]	18.57	0.6666	**1.80±0.30**	135.62	0.140	0.1962
MS-ACM [15]	**19.45**	0.7331	1.60±0.24	**134.09**	0.122	0.2030
SCGN (ours)	19.41	**0.7459**	1.71±0.24	134.11	**0.121**	**0.2317**
Baseline b_{90}						
Method	PSNR	MS-SSIM	Inception Score	mMSE	L_1	LOGS
VSAF [10]	8.99	0.1165	-	187.32	0.549	0.2156
DVM [12]	14.61	0.4409	-	146.66	0.316	0.1994
ADVM	14.92	0.4825	**2.36±0.36**	145.47	0.297	0.2140
ED [12]	17.68	0.6528	-	138.43	0.160	0.2075
VI-GAN [24]	18.53	0.6782	1.88±0.38	136.13	0.142	0.1972
MS-ACM [15]	19.27	0.7282	1.65±0.27	134.85	0.127	0.1840
SCGN (ours)	**19.29**	**0.7418**	1.74±0.28	**134.65**	**0.123**	**0.2258**

F. View Synthesis Results on Outdoor Scenes

Quantitative comparisons using the evaluation metrics are shown in Table VI and Table VII. From Table VI, SCGN outdistances the state-of-the-art view synthesis methods on PSNR and MS-SSIM, although the inception score of ADVM is slightly higher than that of our method. From Table VII, our method consistently outperforms the other methods according to several metrics, which verifies its effectiveness. It is worth noting that the image-to-image translation task is different from view synthesis because the latter has to synthesize a novel view from two distinct views with more strict constraints (*e.g.*, generation of occluded contents), and it is hard to handle the synthesis task under large baselines for the compared geometry-based methods in Table VII which work well under stereo settings or other small baselines.

We show the experimental results of visual comparison in Fig. 8. For VSAF and the Encoder-Decoder (ED), the synthesized frontal views contain many deformations and blurs. For DVM and ADVM, artifacts still appear in some regions even through the generated images look realistic on the whole. In contrast, SCGN maintains robustness and performs

Fig. 7. Visual comparisons of the synthesized view images under different baselines, where M represents Multi-PIE and $\{b_{60}, b_{75}, b_{90}\}$ are three large different baselines. For Multi-PIE, {View1, View2, GT} represent {left, right, middle} views.

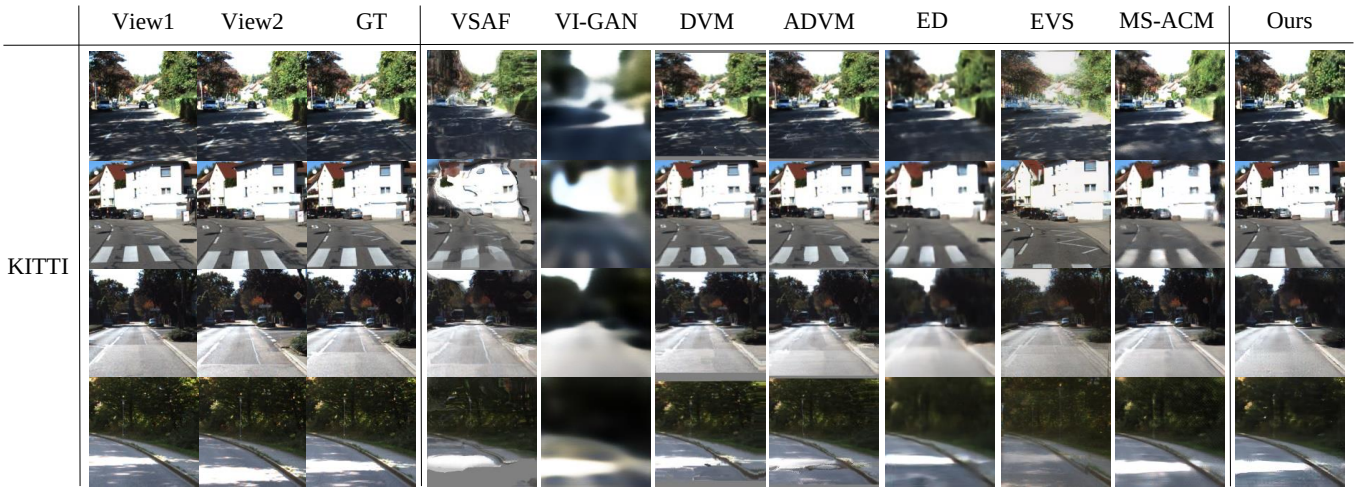


Fig. 8. Visual comparisons of the synthesized view images on KITTI, where {View1, View2, GT} represent {last, next, current} frames.

well in scenes with rich texture, complex background and different light conditions. We also show image details such as shadows on the road in Fig. 10, which further demonstrates the superiority of our methods in terms of details.

G. Results on Real-world Conferencing System

1) Details about the conferencing dataset: For demo evaluation, we set up the experimental environment by placing two cameras on the left side and the right side, and capture view pairs as our inputs. Furthermore, to capture the frontal view as the ground-truth, we place an additional camera at the center of the screen on the same horizontal line as the other cameras.

2) Implementation details: We train the SCGN model on our conferencing dataset with a pre-trained model which is trained on Multi-PIE. Note that the pre-trained data are with similar settings of our conferencing dataset, *i.e.*, people with a background in symmetrical image pairs. We also record two conferencing demos to further demonstrate the effectiveness and robustness of our SCGN. One of the demos shows the real conferencing system with less actions (named as *SCGN_demo_talk.mp4*), and the other shows a scene with much more rapid movements (named as *SCGN_demo_move.mp4*).

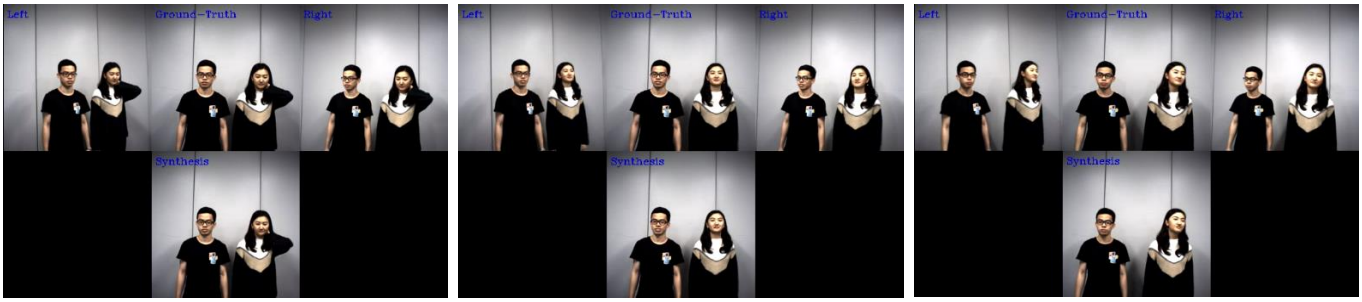


Fig. 9. Visual results of our demo in a real-world video conferencing system with a pre-defined baseline between the left and right cameras.

Fig. 10. Visual comparisons of image details generated by different methods on KITTI. The red boxes and the green boxes emphasize the local details of tree texture and shadow on the road, respectively.

TABLE VI
PERFORMANCE COMPARISON WITH STATE-OF-THE-ART VIEW SYNTHESIS METHODS UNDER DIFFERENT BASELINES ON KITTI.

Method	PSNR	MS-SSIM	Inception Score	mMSE	L_1	LOGS
VSAF [10]	13.87	0.4533	-	151.34	0.258	0.3417
VI-GAN [24]	15.05	0.5294	1.83± 0.06	146.31	0.233	0.1239
DVM [12]	15.48	0.6552	-	144.25	0.205	0.4051
ADVM	16.30	0.6861	2.96±0.29	141.37	0.179	0.3750
ED [12]	17.28	0.6859	-	139.28	0.159	0.3365
EVS [42]	14.74	0.5135	-	148.93	0.250	0.3776
MS-ACM [15]	**19.35**	0.7715	**3.78±0.50**	135.37	0.124	0.3817
SCGN (ours)	19.20	**0.7772**	2.41±0.25	**129.48**	**0.031**	**0.4097**

TABLE VII
COMPARISONS WITH ADDITIONAL STATE-OF-THE-ART METHODS ON KITTI. ALL METHODS ARE TRAINED AND TESTED IN A PAIRED SETUP.

Method	PSNR	MS-SSIM	L_1
Pix2Pix [33]	12.59	0.6943	0.141
CycleGAN [17]	8.17	0.5112	0.226
Multi2Novel [13]	10.36	0.3904	0.413
StereoMagnification [4]	11.87	0.3792	0.194
SCGN (ours)	**19.20**	**0.7772**	**0.031**

3) View synthesis results on our demos: As shown in Fig. 9, for most scenes, our SCGN performs effectively and synthesizes views as photo-realistic as the captured frontal view. The good-quality visual results of the demos that even including two people with occlusions demonstrate the robustness of our SCGN. From the talking demo with fewer actions, we observe that the synthesized views are excellent with the small changes of inputs, and more details and visualized results of the moving demo are shown in the supplementary.

TABLE VIII
ABLATION STUDY OF EACH MODEL COMPONENT ON KITTI. WE COMPARE THE RESULTS IN TERMS OF PSNR, MS-SSIM, AND INCEPTION SCORE WHERE m REPRESENTS THE MODIFIED VERSION.

VSN	VDN	adv	PSNR	MS-SSIM	Inception Score
✓			18.96	0.7382	-
m			18.92	0.7351	-
✓		✓	18.80	0.7533	**2.45±0.22**
✓	m	✓	18.81	0.7432	2.38±0.20
✓	✓		19.16	0.7511	-
✓	✓	✓	**19.20**	**0.7773**	2.41±0.25

TABLE IX
ABLATION RESULTS (PSNR) OF VDN ON MULTI-PIE.

VSN	VDN	adv	$\mathbf{b_S}$	$\mathbf{b_M}$	$\mathbf{b_L}$
✓		✓	25.52	22.59	21.54
✓	m	✓	25.59	22.69	21.68
✓	✓	✓	**26.36**	**22.83**	**21.75**

V. FURTHER EXPERIMENTS

We conduct further experiments on KITTI and Multi-PIE to demonstrate the effectiveness of each component of SCGN, including the View Synthesis Network (VSN), the View Decomposition Network (VDN), the adversarial loss, and the sharpness loss.

A. Effect of View Synthesis Network

We investigate the effect of VSN by comparing the original version of VSN (w/o VDN & adv) and the modified version of VSN (mVSN). Compared to VSN, mVSN removes the max-pooling layers of the encoder and the upsampling layers of the decoder. In addition, we train these two versions of VSN using only the L_p loss in Eq. (12). As shown in Table VIII, all of the evaluation metrics show that the original version of VSN outperforms the mVSN, which demonstrates the necessity of the feature compression and extraction mechanism in VSN.

B. Effect of View Decomposition Network

We investigate the effect of VDN by removing the VDN component (w/o VDN) and removing the L_{vc} in Eq. (12). Furthermore, we modify the separated decoders $\{V^{D,l}, V^{D,r}\}$

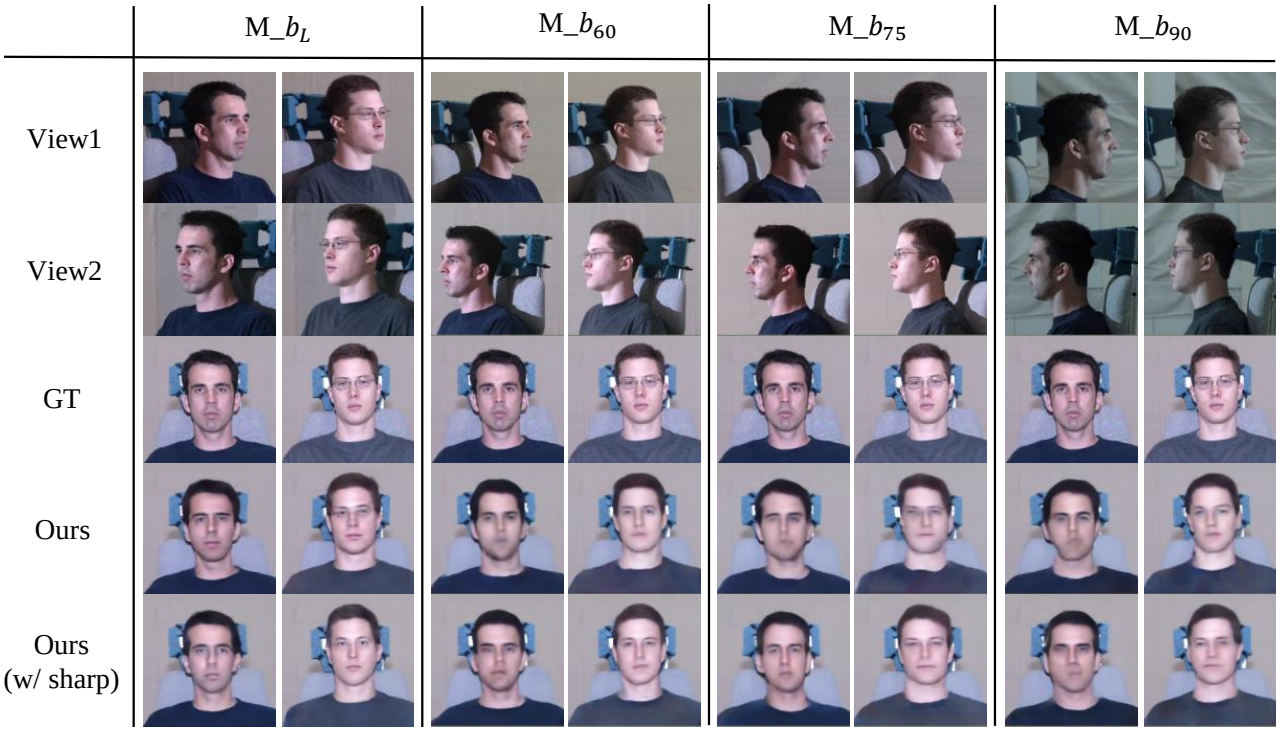


Fig. 11. Visual comparisons of our SCGN and a variant with the sharpness loss L_{sharp} (w/ sharp) under different baselines, where M represents Multi-PIE and $\{b_L, b_{60}, b_{75}, b_{90}\}$ are four different baselines in Multi-PIE..

Fig. 12. Performance comparisons of SCGN with and without the adversarial loss L_{adv} on KITTI dataset.

to be a single weight-shared decoder (mVDN) to evaluate the effect of the separated decoders in VDN.

As shown in Table VIII, our model with VDN (SCGN) significantly outperforms the model without VDN on KITTI in terms of both PSNR and MS-SSIM although the inception score of our model is slightly lower than that without VDN. Our model with VDN (SCGN) also has higher PSNR than that without VDN on different baselines of Multi-PIE as shown in Table IX. Our model (SCGN) also outperforms the mVDN in terms of all evaluation metrics. These results demonstrate that using the two separate decoders to obtain the decomposed side views is more effective than using a single decoder in VDN.

We also show the images regenerated by VDN in Fig. 15. From this figure, VDN is able to decompose the synthesized

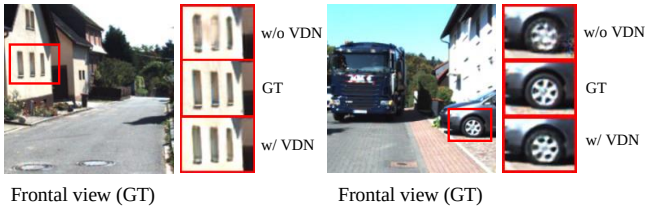


Fig. 13. Performance comparisons of SCGN with (w/) and without (w/o) View Decomposition Network (VDN). The red boxes emphasize the local details of windows and wheels, respectively.

view into the original side views and recover the content details on both Multi-PIE and KITTI. Moreover, from Fig. 13, compared to the synthesized results from a variant of SCGN without VDN, the windows and wheels synthesized by SCGN with VDN are more realistic than those of the ground-truth (GT) with less deformation. All of the above comparisons show that VDN can help resolve the correspondence matching issue and compensate for the lack of rectification, so that VSN does not need to perform any geometric processing in advance, and can directly learn the photo-realistic synthesized views based on image content.

C. Effect of the Adversarial Loss L_{adv}

We investigate the effect of adversarial learning by removing L_{adv} (*i.e.*, Eq. (8)) from the training procedure. As shown in Table VIII, the PSNR, MS-SSIM, and inception score of SCGN without adversarial loss is slightly higher than that with adversarial loss. The adversarial loss introduces diversity to improve the photo-realism, leading to lower evaluation

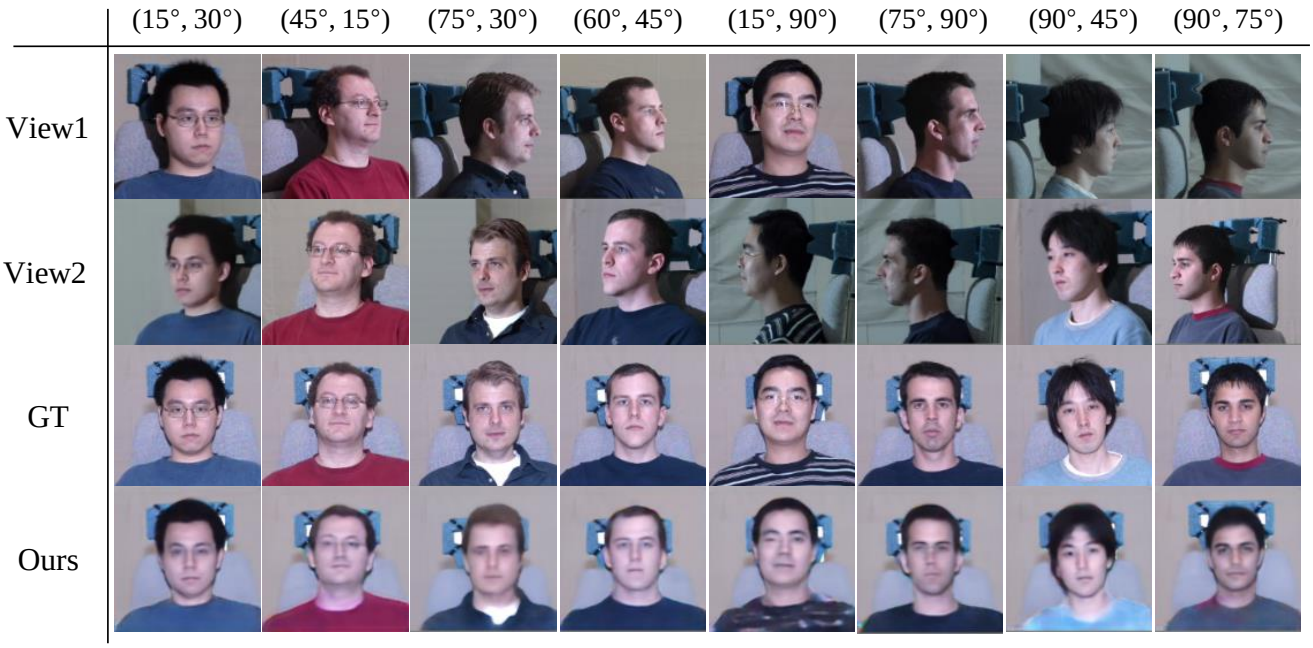


Fig. 14. Visual comparisons of the synthesized view under asymmetric input views, where $(\cdot, \cdot)$ represents input baseline of View1 and View2.

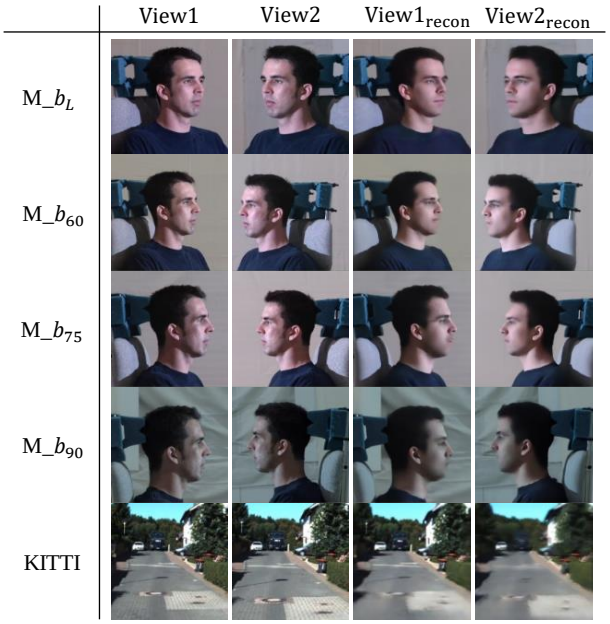


Fig. 15. Regenerated results of VDN on Multi-PIE with large baselines, *i.e.*, b_L ($\pm 45°$), b_{60} ($\pm 60°$), b_{75} ($\pm 75°$), and b_{90} ($\pm 90°$) and on KITTI.

metrics. From visual qualitative comparisons in Fig. 12, it can be seen that SCGN synthesizes rich details such as a clear lane line, tree texture, and shadow on the road, while the SCGN without the adversarial loss synthesizes smooth results.

D. Effect of the Sharpness Loss L_{sharp}

To investigate the effect of L_{sharp} in Eq. 4, we consider 4 view synthesis settings on large baselines, *i.e.*, b_L ($\pm 45°$), b_{60} ($\pm 60°$), b_{75} ($\pm 75°$), and b_{90} ($\pm 90°$). We show the results in Table X and Fig. 11. From the results, the model with the sharpness term outperforms the baseline model in most cases.

TABLE X
ABLATION STUDY OF SHARPNESS LOSS L_{sharp} ON LARGE BASELINES b_L, b_{60}, b_{75}, b_{90} IN MULTI-PIE. b_L, b_{60}, b_{75}, b_{90} INDICATE BASELINE $\pm 45°$, $\pm 60°$, $\pm 75°$, $\pm 90°$ RESPECTIVELY.

Baseline b_L						
L_{sharp}	PSNR	MS-SSIM	Inception Score	mMSE	L_1	LOGS
✗	21.75	0.8268	**1.91±0.24**	137.37	**0.087**	0.2329
✓	**22.03**	**0.8349**	1.83±0.30	**131.10**	**0.087**	**0.2391**
Baseline b_{60}						
L_{sharp}	PSNR	MS-SSIM	Inception Score	mMSE	L_1	LOGS
✗	20.84	0.7929	**1.84±0.33**	132.32	0.102	**0.2443**
✓	**20.93**	**0.7969**	1.78±0.32	**132.23**	**0.101**	0.2383
Baseline b_{75}						
L_{sharp}	PSNR	MS-SSIM	Inception Score	mMSE	L_1	LOGS
✗	19.41	0.7459	1.71±0.24	134.11	0.121	0.2317
✓	**19.55**	**0.7473**	**1.75±0.31**	**133.90**	**0.117**	**0.2374**
Baseline b_{90}						
L_{sharp}	PSNR	MS-SSIM	Inception Score	mMSE	L_1	LOGS
✗	19.29	0.7418	**1.74±0.28**	134.65	0.123	0.2258
✓	**19.42**	**0.7458**	1.68±0.25	**134.42**	**0.120**	**0.2342**

TABLE XI
COMPARISON OF THE AVERAGE INFERENCE LATENCY AND THE PERFORMANCE OF DIFFERENT METHODS ON KITTI DATASET.

Method	VSAF [10]	DVM [12]	ADVM	ED [12]	SCGN
Inference Latency (s)	0.037	0.052	0.064	0.068	**0.036**
PSNR	13.87	15.48	16.30	17.28	**19.20**

TABLE XII
ABLATION STUDY OF ASYMMETRIC BASELINE IN MULTI-PIE, WHERE ASYM. REPRESENTS TRAINING SCGN WITH ASYMMETRIC INPUTS VIEWS.

Asym.	PSNR	MS-SSIM	Inception Score	mMSE	L_1
✗	21.75	**0.8212**	1.86±0.26	131.99	**0.094**
✓	**21.77**	0.8179	**2.14±0.06**	**131.48**	0.095

E. Discussion on Asymmetric Input Views

We apply our method to the view synthesis tasks with asymmetric input views on the Multi-PIE dataset. In the experiments, we randomly sample angles from the range between $15°$ to $90°$ to construct the asymmetric views. From the results in Table XII and Fig. 14, our SCGN is able to produce photo-realistic views from asymmetric input views.

TABLE XIII
EFFECT OF λ_1 ON THE PERFORMANCE OF SCGN.

λ_1	Multi-PIE			KITTI		
	PSNR	MS-SSIM	IS	PSNR	MS-SSIM	IS
0.1	23.73	0.8722	**2.34±0.30**	18.03	0.6957	2.50±0.30
0.01	**23.95**	**0.8774**	2.19±0.24	**19.20**	**0.7773**	2.41±0.25
0.001	23.08	0.8511	2.06±0.27	18.80	0.7433	**2.53±0.23**

TABLE XIV
EFFECT OF λ_2 ON THE PERFORMANCE OF SCGN.

λ_2	Multi-PIE			KITTI		
	PSNR	MS-SSIM	IS	PSNR	MS-SSIM	IS
0.01	23.65	0.8655	**2.24±0.30**	18.53	0.7338	2.32±0.18
0.001	**23.95**	**0.8774**	2.19±0.24	**19.20**	**0.7773**	**2.41±0.25**
0.0001	23.06	0.98498	1.99±0.22	18.49	0.7262	2.28±0.21

TABLE XV
EFFECT OF λ_3 ON THE PERFORMANCE OF SCGN.

λ_3	Multi-PIE			KITTI		
	PSNR	MS-SSIM	IS	PSNR	MS-SSIM	IS
0.1	19.71	0.7924	1.80±0.22	17.02	0.6840	2.52±0.32
0.01	**23.95**	**0.8774**	**2.19±0.24**	**19.86**	**0.7706**	**3.56±0.41**
0.001	23.08	0.8511	2.06±0.27	19.73	0.7605	3.53±0.53

F. Effect of λ_1, λ_2, and λ_3 on the Performance of SCGN

In this section, we investigate the effect of λ_1, λ_2, and λ_3 in Eq. (12) on Multi-PIE and KITTI. Table XIII shows the experimental results with different λ_1 values when $\lambda_2 = 0.001$ and $\lambda_3 = 0.01$. The results for $\lambda_1 = 0.01$ are better than the others in terms of PSNR, MS-SSIM, and Inception Score (IS) on both datasets. We also evaluate our method with different λ_2 values when $\lambda_1 = 0.01$ and $\lambda_3 = 0.01$. Table XIV shows that SCGN with $\lambda_2 = 0.001$ achieves the best performance on three metrics. In addition, we investigate our method with different values of λ_3 when $\lambda_1 = 0.01$ and $\lambda_2 = 0.001$. From Table XV, our method performs the best when $\lambda_3 = 0.01$ in terms of three metrics. As a result, we suggest setting $\lambda_1 = 0.01$, $\lambda_2 = 0.001$, and $\lambda_3 = 0.01$ for SCGN by default.

G. Comparison of the Inference Latency of Different Methods

In this section, we show the average inference latency of different methods on the KITTI dataset using a single Nvidia TitanX GPU. We show the comparison results of latency and performance in Table XI. From these results, our method exhibits the fastest inference speed (27 fps) but yields the best performance above all the other compared methods.

VI. CONCLUSION

We have presented a simple but effective view synthesis network to synthesize unseen frontal and middle views from two side views with a large camera baseline without geometric processing. Specifically, we propose a view decomposition network by learning an inverse mapping from the synthesized view back to the input view pair to preserve content consistency; this mapping can take the place of rectification and solve the pixel-level matching problem. To improve the photo-realism of images, we further introduce an adversarial loss to increase the likelihood that the synthesized images will be indistinguishable from the real views. As a result, the proposed method can simultaneously produce photo-realistic unseen views and preserve the view consistency among all views of the same scene. Using different baselines, the proposed method consistently outperforms the other methods in terms of both quantitative and visual comparisons.

REFERENCES

[1] N. Atzpadin, P. Kauff, and O. Schreer, "Stereo analysis by hybrid recursive matching for real-time immersive video conferencing," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 3, pp. 321–334, 2004.

[2] D. Scharstein, "Stereo vision for view synthesis," in *IEEE Conference on Computer Vision and Pattern Recognition*, 1996, pp. 852–858.

[3] K.-J. Oh, S. Yea, and Y.-S. Ho, "Hole filling method using depth based in-painting for view synthesis in free viewpoint television and 3-d video," in *Picture Coding Symposium*. IEEE, 2009, pp. 1–4.

[4] T. Zhou, R. Tucker, J. Flynn, G. Fyffe, and N. Snavely, "Stereo magnification: Learning view synthesis using multiplane images," in *SIGGRAPH*, 2018.

[5] R. Huang, S. Zhang, T. Li, and R. He, "Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis," in *IEEE International Conference on Computer Vision*, 2017.

[6] X. Jin, Z. Liu, Q. Li, and Q. Dai, "Depth assisted adaptive workload balancing for parallel view synthesis," *IEEE Transactions on Multimedia*, vol. 20, no. 11, pp. 2891–2904, Nov 2018.

[7] B. Ceulemans, S. Lu, G. Lafruit, and A. Munteanu, "Robust multiview synthesis for wide-baseline camera arrays," *IEEE Transactions on Multimedia*, vol. 20, no. 9, pp. 2235–2248, Sep. 2018.

[8] B. Graham, D. Novotny, and J. Reizenstein, "Perspectivenet: A scene-consistent image generator for new view synthesis in real indoor environments," in *Conference on Neural Information Processing Systems*. Curran Associates, Inc., 2019, pp. 7599–7610.

[9] M. Tatarchenko, A. Dosovitskiy, and T. Brox, "Multi-view 3d models from single images with a convolutional network," in *European Conference on Computer Vision*, 2016, pp. 322–337.

[10] T. Zhou, S. Tulsiani, W. Sun, J. Malik, and A. A. Efros, "View synthesis by appearance flow," in *European Conference on Computer Vision*, 2016, pp. 286–301.

[11] J. Flynn, M. Broxton, P. Debevec, M. DuVall, G. Fyffe, R. Overbeck, N. Snavely, and R. Tucker, "Deepview: View synthesis with learned gradient descent," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2019.

[12] D. Ji, J. Kwon, and M. McFarland, "Deep View Morphing," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[13] S.-H. Sun, M. Huh, Y.-H. Liao, N. Zhang, and J. J. Lim, "Multi-view to novel view: Synthesizing novel views with self-learned confidence," in *European Conference on Computer Vision*, 2018, pp. 155–171.

[14] A. Dosovitskiy, J. Tobias Springenberg, and T. Brox, "Learning to generate chairs with convolutional neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1538–1546.

[15] Y. Zhang, D. Zou, J. S. Ren, Z. Jiang, and X. Chen, "Structure-preserving stereoscopic view synthesis with multi-scale adversarial correlation matching," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5860–5869.

[16] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker, "Multi-pie," in *Image and Vision Computing*, vol. 28, no. 5, 2010, pp. 807 – 813.

[17] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *IEEE International Conference on Computer Vision*, 2017.

[18] P. E. Debevec, C. J. Taylor, and J. 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*. ACM, 1996, pp. 11–20.

[19] C. Sagonas, Y. Panagakis, S. Zafeiriou, and M. Pantic, "Robust statistical face frontalization," in *IEEE International Conference on Computer Vision*, 2015, pp. 3871–3879.

[20] M. Liu, X. He, and M. Salzmann, "Geometry-aware deep network for single-image novel view synthesis," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4616–4624.

[21] J. Flynn, I. Neulander, J. Philbin, and N. Snavely, "Deepstereo: Learning to predict new views from the world's imagery." in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[22] E. Park, J. Yang, E. Yumer, D. Ceylan, and A. C. Berg, "Transformation-grounded image generation network for novel 3d view synthesis," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[23] Y. Zhang, D. Zou, J. S. Ren, Z. Jiang, and X. Chen, "Structure-preserving stereoscopic view synthesis with multi-scale adversarial correlation matching," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2019.

[24] X. Xu, Y.-C. Chen, and J. Jia, "View independent generative adversarial network for novel view synthesis," in *IEEE International Conference on Computer Vision*, October 2019.
[25] I. Choi, O. Gallo, A. Troccoli, M. H. Kim, and J. Kautz, "Extreme view synthesis," in *IEEE International Conference on Computer Vision*, October 2019.
[26] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Conference on Neural Information Processing Systems*, 2014, pp. 2672–2680.
[27] Y. Guo, Q. Chen, J. Chen, Q. Wu, Q. Shi, and M. Tan, "Auto-embedding generative adversarial networks for high resolution image synthesis," *IEEE Transactions on Multimedia*, 2019.
[28] J. Cao, Y. Guo, Q. Wu, C. Shen, J. Huang, and M. Tan, "Improving generative adversarial networks with local coordinate coding," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.
[29] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *International Conference on Learning Representations*, 2016.
[30] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International Conference on Machine Learning*, vol. 70, 2017, pp. 214–223.
[31] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," in *International Conference on Learning Representations*, 2018.
[32] Y. Guo, J. Chen, J. Wang, Q. Chen, J. Cao, Z. Deng, Y. Xu, and M. Tan, "Closed-loop matters: Dual regression networks for single image super-resolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5407–5416.
[33] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1125–1134.
[34] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.
[35] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *International Conference on Learning Representations*, 2016.
[36] J. Li, Z. L. Yu, Z. Gu, H. Liu, and Y. Li, "Dilated-inception net: multi-scale feature aggregation for cardiac right ventricle segmentation," *IEEE Transactions on Biomedical Engineering*, vol. 66, no. 12, pp. 3499–3508, 2019.
[37] L. Li, Y. Zhou, K. Gu, W. Lin, and S. Wang, "Quality assessment of dibr-synthesized images by measuring local geometric distortions and global sharpness," *IEEE Transactions on Multimedia*, vol. 20, no. 4, pp. 914–926, April 2018.
[38] A. G. . P. L. . R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.
[39] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015.
[40] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: a system for large-scale machine learning," in *OSDI*, 2016, pp. 265–283.
[41] T. Habtegebrial, K. Varanasi, C. Bailer, and D. Stricker, "Fast view synthesis with deep stereo vision," in *Computer Vision, Imaging and Computer Graphics Theory and Applications*, 2019, pp. 792–799.
[42] I. Choi, O. Gallo, A. Troccoli, M. H. Kim, and J. Kautz, "Extreme view synthesis," in *IEEE International Conference on Computer Vision*, 2019, pp. 7781–7790.
[43] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Conference on Neural Information Processing Systems*, 2019, pp. 8024–8035.
[44] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Conference on Neural Information Processing Systems*, 2016, pp. 2234–2242.
[45] M. Yang, C. Zhu, X. Lan, and N. Zheng, "Efficient estimation of view synthesis distortion for depth coding optimization," *IEEE Transactions on Multimedia*, vol. 21, no. 4, pp. 863–874, April 2019.
[46] Y. Zhou, L. Li, S. Wang, J. Wu, Y. Fang, and X. Gao, "No-reference quality assessment for view synthesis using dog-based edge statistics and texture naturalness," *IEEE Transactions on Image Processing*, vol. 28, no. 9, pp. 4566–4579, 2019.
[47] Y. Huang, L. Li, H. Zhu, and B. Hu, "Blind quality index of depth images based on structural statistics for view synthesis," *IEEE Signal Processing Letters*, vol. 27, pp. 685–689, 2020.
[48] L. Li, Y. Zhou, J. Wu, F. Li, and G. Shi, "Quality index for view synthesis by measuring instance degradation and global appearance," *IEEE Transactions on Multimedia*, 2020.

Zhuoman Liu is currently a researcher with Guangzhou Shiyuan Electronic Technology Co., Ltd (CVTE) in Guangzhou, China. She received the bachelor degree in Software Engineering from South China University of Technology in 2019. Her main research interests include deep learning and computer vision.

Wei Jia is currently a senior researcher at Guangzhou Shiyuan Electronic Technology Co., Ltd (CVTE) in Guangzhou, China. She leads a 3D vision group and her research interests include 3D imaging, 3D reconstruction, neural networks, and deep learning. She received her Ph.D degree in Computer Science from Dundee university, UK, in 2012. She received her Master degree in Computer Science from University of Bristol, UK, in 2006. She received her bachelor degree in Computer Science from Harbin Engineering University, China, in 2004.

Ming Yang is currently the CTO of Guangzhou Shiyuan Electronic Technology Co., Ltd (CVTE) in Guangzhou, China. He received his B.S. and Ph.D. degree from Sun Yat-sen University in 2009 and 2014, respectively. He joined CVTE Research in 2014. His research interests include machine learning and interactive computer vision.

Peiyao Luo is a M.S. candidate with the School of Software Engineering at South China University of Technology. She also received her bachelor degree in Mechatronic Engineering from the same university in 2018. Her research interests include deep learning and computer vision.

Yong Guo is a Ph.D. candidate with the School of Software Engineering at South China University of Technology. He also received his bachelor degree in Software Engineering from the same university in 2016. His research interests include deep learning and computer vision.

Mingkui Tan is currently a professor with the School of Software Engineering at South China University of Technology. He received his Bachelor Degree in Environmental Science and Engineering in 2006 and Master degree in Control Science and Engineering in 2009, both from Hunan University in Changsha, China. He received the Ph.D. degree in Computer Science from Nanyang Technological University, Singapore, in 2014. From 2014-2016, he worked as a Senior Research Associate on computer vision in the School of Computer Science, University of Adelaide, Australia. His research interests include machine learning, sparse analysis, deep learning and large-scale optimization.