

# A Portable Multiscopic Camera for Novel View and Time Synthesis in Dynamic Scenes

Tianjia Zhang, Yuen-Fui Lau, and Qifeng Chen

**Abstract**—We present a portable multiscopic camera system with a dedicated model for novel view and time synthesis in dynamic scenes. Our goal is to render high-quality images for a dynamic scene from any viewpoint at any time using our portable multiscopic camera. To achieve such novel view and time synthesis, we develop a physical multiscopic camera equipped with five cameras to train a neural radiance field (NeRF) in both time and spatial domains for dynamic scenes. Our model maps a 6D coordinate (3D spatial position, 1D temporal coordinate, and 2D viewing direction) to view-dependent and time-varying emitted radiance and volume density. Volume rendering is applied to render a photo-realistic image at a specified camera pose and time. To improve the robustness of our physical camera, we propose a camera parameter optimization module and a temporal frame interpolation module to promote information propagation across time. We conduct experiments on both real-world and synthetic datasets to evaluate our system, and the results show that our approach outperforms alternative solutions qualitatively and quantitatively. Our code and dataset are available at <https://yuenfui.lau.github.io/>.

## I. INTRODUCTION

Novel view synthesis (NVS) has been a trending research topic for 3D scene rendering and modeling, especially with the progress led by neural radiance fields (NeRF) [1]. NVS has potential real-life applications such as AR/VR in Metaverse, robotic vision system, aerial surveying, and autonomous driving [2]. For instance, Block NeRF [3] has presented a 3D reconstruction approach to renders a 3D map on a city. Such an application can also be further extended to autonomous driving in terms of route planning and real-time environmental adaption. Currently, most existing approaches focus on novel view synthesis using multi-view images captured in a static scene or a monocular video of moving objects, which does not utilize multiscopic information over time. So we might want to ask: can we build a multiscopic camera tailored for novel view and time synthesis in dynamic scenes in the real world?

NVS in dynamic scenes is generally challenging, and most existing methods make some assumptions of captured input data. For instance, HyperNerf [4] assumes that the dynamics from a scene can be represented by a 2D latent vector, which limits the freedom of potential dynamic movements in a scene. Nerfies can only model a small amount of deformation of an object (i.e., head) [5].

In this work, we aim at solving novel view and time

Authors are with the Hong Kong University of Science and Technology, Hong Kong, China. T. Zhang (tzhangbl@connect.ust.hk) is with the Individualized Interdisciplinary Program and the Robotics Institute. Y. Lau (yflauad@connect.ust.hk) is with the Department of Mathematics. Prof. Q. Chen (cqf@ust.hk) is with the Department of Computer Science and Engineering and the Department of Electronic and Computer Engineering.

synthesis in a dynamic environment without any assumption about object movements, such as a room with walking persons, using a portable multiscopic camera that is accessible to general users. Our camera captures multi-view images at every time step (30 frames per second), which can provide abundant information for robust novel view and time synthesis. The portable multiscopic sensor is constructed with five cameras following the *top-left-center-right-bottom* layout in the size of a laptop (about 30cm × 30cm) [6], [7]. The camera allows us to capture synchronized videos from five different perspectives.

With our multiscopic camera, we propose an end-to-end NeRF-based method to perform novel view and time synthesis in dynamic scenes. Similar to previous time encoding work on NeRF [8], we extend NeRF to model emitted radiance and density of a point as a function of a 4D space-time coordinate and a 2D viewing direction. Compared to previous methods [1], [9], [10], which rely on either calibration or structure-from-motion method to obtain camera parameters, we embed the camera parameters and time-variable into a learnable model to be optimized during the training process, inspired by Wang et al. [11]. To encourage information propagation across time, we apply a video frame interpolation model SloMo [12] to generate several intermediate images between two consecutive frames and enforce our temporal NeRF model to render photo-realistic images at an arbitrary time. As a result, our model can perform not only novel view synthesis tasks but also novel time synthesis generated from our model. Fig. 1 illustrates the objective of our multiscopic camera, and Fig. 2 shows the layout design and the physical prototype of our multiscopic camera.

To quantitatively evaluate our model for novel view and time synthesis, it is desirable to collect high-quality real-world images in a dynamic scene rendered from arbitrary viewpoints (beyond the five cameras) at any time, but such a data collection process requires lots of human effort and dedicated a large camera array system. Thus we build a synthetic dataset by applying a simulation engine to render images in a dynamic scene from any viewpoints. Our simulation system is based on the well-known Habitat-Sim system<sup>1</sup>, a modular intuitive and high-performance simulator for studying embodied AI. We can render high-quality images using 3D scene data with moving object (e.g., a flying chair) at a reasonable speed.

To summarize, our main contributions include

<sup>1</sup>Further details can be found at <https://github.com/facebookresearch/habitat-sim>.

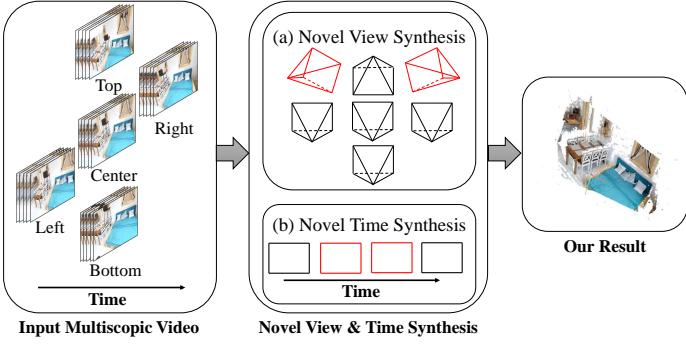


Fig. 1. **Illustration of the objective of our multiscopic camera.** From several multiscopic input videos or images, we learn a time-varying scene representation to model the geometry and appearance of the scene. From such representation, we can generate novel images (red line) at any pose (a) and time (b).

- A multiscopic camera prototype for novel view and time synthesis,
- A NeRF model trained end-to-end to perform novel view and time synthesis in dynamic scenes, at any camera pose and any time.
- A dataset of real-world data captured using the proposed camera and synthetic data rendered from customized simulators.

## II. RELATED WORK

**Novel View Synthesis.** Traditional methods on novel view synthesis are usually achieved by interpolation from densely distributed multiple perspectives. However, with the rise of deep learning, many researchers have applied various neural network structures to learn the synthesis process, and it has gradually become the mainstream research direction. The key component to achieve successful view synthesis is to reconstruct an accurate and compact scene representation that could reflect the scene's geometry and visual appearance. At present, the most commonly used methods are volumetric representations [13] and multiple-plane images (MPI) [14]–[16]. Volumetric representation can be used to represent very complex scenes and objects. Researchers train neural networks to predict the color of voxels [17], [18] from images and then use rendering techniques [19] to generate images. Additionally, MPI is a multi-layered scene representation consisting of a series of fronto-parallel planes at a fixed depth range in front of capturing the view. However, these methods are limited by discrete sampling. As the resolution increases, those methods cannot meet the needs of rendering ultra-high-definition images because the storage and loading time will increase sharply. [20] have explored the application of MPI method on free-viewpoint avatar rendering of a moving person. However, it targets at dynamic rendering of dynamic characters and cannot adapt to arbitrary dynamic scenes. The recent emergence of implicit representations [21], [22] can model the scenes as a continuous function of spatial locations and other important geometry parameters. Those parameters can then be optimised via training.

**Neural Radiance Field** NeRF [1] is a neural implicit representation for modeling a single scene. It encodes scenes and objects in the MLP so that continuous scenes can

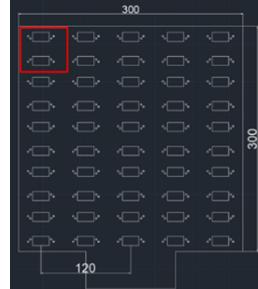


Fig. 2. **The physical structure of our multiscopic camera.** The five cameras are fixed to the supporting board of size  $30cm \times 30cm$ , following the bottom-center-left-right-top layout.

be represented at a relatively low cost. However, due to the limit of the capacity of the MLP itself and the overfitting optimization mechanism, NeRF can only be used to represent a single scene, which significantly limits its application. Many subsequent works were used to improve NeRF's representation ability and expand the scope of application in terms of generalization ability [23], dynamic and deformable scenes synthesis [24], applications under extreme environments [25], improving training and rendering efficiency [26], reconstruction on specific shape domains [27], [28] and so on. Our interest focuses more on works extending NeRF to generate free-viewpoint videos. Inspired from level-set methods which use slices to represent the movement of surfaces, Hyper-NeRF [4] addresses the problem by lifting canonical NeRF into a higher dimensional space and slicing a cross-section to model topologically changing shapes. Several methods have been explored on extending NeRF to dynamic scenes. [9], [10] combine flow models with NeRF to warp source frames to target frames and thus it could model the temporal dimension. [29] shares a similar idea with us in that both represent the dynamic scenes as a function of space and time. However, these methods emphasize processing monocular videos since time-synchronized multi-view data is difficult to collect. Recently, cameras with image and depth sensors have been mass-manufactured due to the great demand for 3D reconstruction and autonomous driving. Additional information which helps capture scene geometry [30] could compensate the scale drawbacks of using visual information alone. However, depth information is not good at dense sensing and preserving visual consistency. Our method demonstrates that aggregating homogeneous visual information from multiple color sensors could achieve impressive reconstruction and rendering.

**Multi-camera System and 3D Scene Simulator** Humans evolve two eyes to better sense 3D geometry of the world. The biological phenomenon inspires the development of binocular stereo cameras. Nowadays stereo cameras are widely used in robotics and computer vision. However, two views could not provide sufficient constrain to estimate dense correspondence matching. Mathematicians [31] have been exploring abstract multi-view geometry for years, laying a solid theoretical foundation for the application of multi-view sensor system. Many works create camera arrays [32] to en-

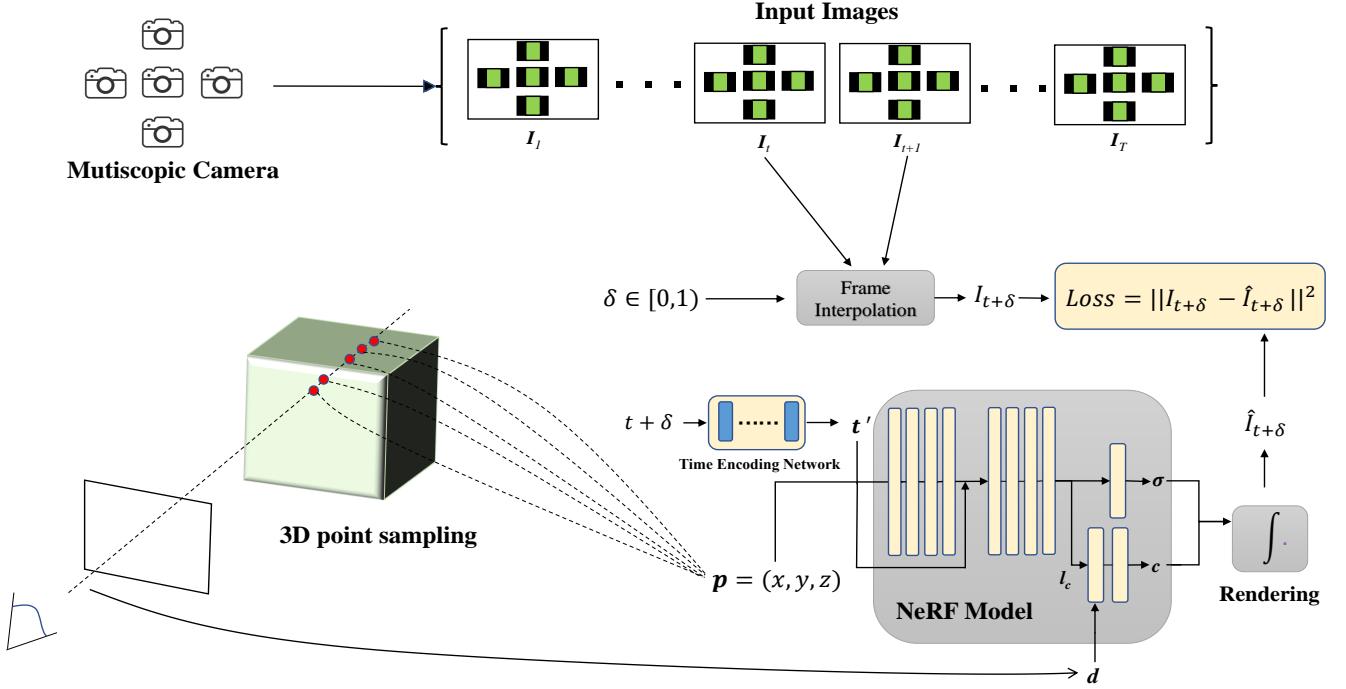


Fig. 3. The pipeline of our method demonstrates the training and testing process, where  $\delta \in [0,1]$ . To render an image  $\hat{I}_{t+\delta}$ , given optimized camera intrinsic and extrinsic parameters, an embedded latent time representation  $t'$  obtained from the time encoding network and 3D points ( $x, y, z$ ) sampled along camera rays are fed together into the NeRF Model with the viewing direction  $d$  to generate color radiance  $c$  and density  $\sigma$ . The color  $c$  and density  $\sigma$  are then aggregated together to produce image  $\hat{I}_{t+\delta}$ .

hance stereo matching between any two cameras and produce high-quality dense depth maps. However, those cumbersome camera systems are difficult to move. Other works [6], [7] use a single mobile camera to capture multiscopic data and apply the collected data to investigate depth estimation and stereo matching methods. Nonetheless, they cannot capture dynamic scene. [33] developed a spherical camera array consisting of 46 low-cost but wide-FOV mobile cameras. They designed extra software and hardware interface to synchronize internal timing. However, processing such a large amount of video data costs large memory and encodes redundant information. Our multiscopic camera system [6], [7] could balance portability, lightweight, and sufficient consistency constraints. Besides real-world data, synthetic data is also widely employed in researching synthesis methods. Some 3D datasets [34], [35] and corresponding simulators [36], [37] have been proposed to visualize three-dimensional scenes, but their interactivity capabilities are inferior. Recently, with the emergence of simulators such as Gibson [38] and Habitat-sim [39], [40] that focus on embodied AI tasks, it has become possible to train robots in simulators. Based on the powerful performance of the simulator, our method demonstrates that the simulators' data can be used for novel view and time synthesis which show great potential for computer vision tasks.

### III. METHOD

Our goal is to render images or videos at arbitrary viewpoints and time given a set of multiscopic videos without knowing other prior information. Mathematically, We denote  $\{(I_t^1, P_t^1), \dots, (I_t^N, P_t^N)\}_{t=1}^T$  the source images and corre-

sponding poses at time  $t$ , where  $T$  and  $N$  are the number of frames and the number of views ( $N = 5$ ), respectively. Our objective is to render a target image  $\hat{I}_t$  with the specified target pose  $P$  at time  $t$ . To address the problem, we formulate the neural radiance field as a function of a 4D space-time coordinate  $(x, y, z, t)$  and a 2D viewing direction  $d$ . The output consists of time-dependent radiance  $c_t$  and volume density  $\sigma_t$ . To exploit and integrate the visual consistency information across time, we apply the frame interpolation method [12] to generate intermediate frames, which serve as extra ground-truth to supervise the training process. If the camera parameters are unknown, we simultaneously optimize the intrinsic parameter in normal space and the extrinsic parameters in  $SE(3)$  space.

#### A. NeRF Model

NeRF [1] applies an MLP network to predict color radiance and density for an input 5D vector consisting of a 3D position vector and a 2D viewing direction vector. The color radiance represents the RGB value at the location, and the density represents the probability that a camera ray terminates at that location. The viewing direction is used to cope with viewing-dependent effects such as reflection and diffusion. However, this amount of information is insufficient for NeRF to render a correct image in a dynamic scene. The time dependence is essential for the sake of novel view synthesis via time. Hence, the input for our NeRF is slightly different from the standard one. Since it deals with a changing environment over time, an extra time coordinate  $t$  will be introduced to allow our model to represent time domain. Additionally, taking inspiration from a parameter-

ized pose learning network [11], we apply a latent time representation. The time coordinate  $t$  is warped using an MLP to get a higher level representation, allowing the model to distinguish between frames at each moment more easily. The latent time representation is given by

$$\mathbf{t}' = \mathcal{W}(t), \quad (1)$$

where  $\mathcal{W}$  is a time encoding network, and  $\mathbf{t}'$  is a latent representation of the time coordinate.

Since the density of a point solely relies on the space and time coordinate, the model divides the mapping relation into two parts: predicting density using only the space-time coordinate and predicting radiance color using both the space-time coordinate and viewing direction. Mathematically, the relation could be represented by a two-stage MLP:

$$\begin{aligned} \sigma, \mathbf{l}_c &= \text{MLP}_1(\mathbf{p}, \mathbf{t}'), \\ \mathbf{c} &= \text{MLP}_2(\mathbf{l}_c, \mathbf{d}), \end{aligned} \quad (2)$$

where  $\mathbf{p} = (x, y, z)$  is the 3D spatial coordinates;  $\text{MLP}_1$  comprises a 8-layer network which transforms the 4D input into the scalar density  $\sigma$  and a 256-D hidden feature vector  $\mathbf{l}_c$ ; the feature vector  $\mathbf{l}_c$  is concatenated with the unit viewing direction  $\mathbf{d}$  and then fed into the final layer  $\text{MLP}_2$ ; the radiance color  $\mathbf{c}$  is the output in terms of RGB value. Following NeRF [1], volume rendering is applied to produce the pixel color of a ray passing through the scene. Specifically, for one pixel, a camera ray is cast originated at the camera center. For each point in the ray we can predict its density and radiance using the model Eq. 2. The color of a pixel is obtained by integrating the weighted product of the radiance value and the density along the camera ray. However, it is difficult to computer the continuous integral. Fortunately, quadrature is a powerful tool to approximate the integral. Suppose a camera ray  $\mathbf{r}$  has origin  $\mathbf{o}$  and unit viewing direction  $\mathbf{d}$ , we sample  $Z$  points on the ray  $\mathbf{r}(z) = \mathbf{o} + z\mathbf{d}$  within the near and far scene bound  $z_n, z_f$ . The estimated color along the ray  $\mathbf{r}$  is calculated by

$$\hat{\mathbf{C}}(\mathbf{r}) = \sum_{i=1}^Z T_i (1 - e^{-\sigma_i \Delta_i}) \mathbf{c}_i, \quad (3)$$

where  $T_i = \exp(-\sum_{j=1}^{i-1} \sigma_j \Delta_j)$  and  $\Delta_i = z_{i+1} - z_i$  is the distance between  $z_{i+1}$  and  $z_i$ .

Our method synthesizes images at given poses and time steps. We do not render a full image during training since it would run out of memory. Instead, we randomly cast camera rays for corresponding image pixels using the optimised camera parameters and render RGB values with our model. In general, our losses are evaluated in the sample space using mean square error:

$$\mathcal{L}_{MSE} = \sum_t^T \sum_i^N \|I_t^i - \hat{I}_t^i\|^2, \quad (4)$$

where  $I_t$  is the sampled ground-truth image at step  $t$  and  $\hat{I}_t$  is the rendered image outcome.

### B. Frame Interpolation

It is intuitive to directly add the time variable to the input and transform the 4D space-time coordinates using the MLP network. However, we found that each dimension would

entangle with the other. Consequently, the simple mapping alone cannot model such a complex relationship and results in image degradation through time. We disentangle the time and space by using an optical-flow-based SloMo model to enforce pixel consistency in the temporal domain.

For the  $i$ -th sensor (out of  $N$ ) in our multiscopic camera, given two adjacent frames  $I_t$  and  $I_{t+1}$  (we regard  $I_t$  as  $I_t^i$  for simplicity in the following description), we apply the Super SloMo method [12] to render an intermediate frame  $I_{t+\delta}$  ( $0 \leq \delta < 1$ ) and use the 4D-input NeRF model (Eq. 2) to predict a frame using the same settings, we encourage the two generated frame to be similar to enhance temporal connection.

We exploit neural networks to predict the optical flow between the intermediate target and source frames. The frame is generated by interpolating the images warped from initial images using the flow field. However, the target frame image is not known initially, which indicates we cannot obtain the optical flow field from the source frames to the target frame. To deal with the issue, we assume that the motion of the dynamic scenes is slow and smooth. Therefore the target flow fields could be estimated from the weighted average of the bidirectional flow fields  $F_{t \rightarrow t+1}$  and  $F_{t+1 \rightarrow t}$ .

The estimation works poorly near the motion boundaries and cannot address the occlusion issue. As FlowNet2 [41] indicates, the cascaded CNN architecture is beneficial to refine the predicted results and reduce artifacts. In addition, the visibility maps, which represent whether a pixel is visible when moving from one step to another, are used to handle occlusions. Therefore, the refining process takes the coarse optical flow prediction and source color images as input and outputs the refined optical flow fields and visibility maps.

Since the input and output of the flow estimation and refinement process are all grid maps, the mapping relation follows the encoder-decoder architecture and can be modeled using a U-Net [42] convolutional neural network. In particular, given  $\hat{F}_{t+\delta \rightarrow t+1}$  and  $\hat{F}_{t+\delta \rightarrow t}$  that are coarse predicted flow fields and  $g(I_{t+1}, \hat{F}_{t+\delta \rightarrow t+1})$  and  $g(I_t, \hat{F}_{t+\delta \rightarrow t})$  that represent synthesizing potential target images by warping from source images  $I_t$  and  $I_{t+1}$  using the corresponding coarse predicted flow fields, respectively, we then feed them into another U-Net to generate the predicted flow fields, visibility maps  $V$ , and refined RGB images. Finally, those flow features output are aggregated together as demonstrated in Super SloMo approach [12] to produce the intermediate frame:

$$I_{t+\delta} = \frac{1}{\lambda} \odot [(1 - \delta)V_{t \rightarrow t+\delta} \odot g(I_t, \hat{F}_{t+\delta \rightarrow t}) + \delta V_{t+1 \rightarrow t+\delta} \odot g(I_{t+1}, \hat{F}_{t+\delta \rightarrow t+1})]. \quad (5)$$

where  $\lambda = (1 - \delta)V_{t \rightarrow t+\delta} + \delta V_{t+1 \rightarrow t+\delta}$  is normalization factor and  $\odot$  denotes the element-wise multiplication.

In addition, the temporal consistency loss is added to enforces our model which is learnt from the flow models inherently to find the correspondence of pixels across time. We randomly select two consecutive frames from a single input video. We use our model to render the pixels sampled from the intermediate frame given the estimated pose  $P$  and

time step  $t$ . At the same time, the flow model could interpolate between the selected frames and output the warped RGB values for the same pixels. The operations used in our model are all differentiable and can be performed end-to-end. Fig. 3 illustrates how to train our model from scratch and produce photo-realistic novel images from the trained model. The temporal consistency loss is the mean squared error of the two values, which is calculated by

$$\begin{aligned} I_{t+\delta} &= \text{Interpolation}(I_t, I_{t+1}, \delta), \\ \mathcal{L}_{t+\delta} &= \|\hat{I}_{t+\delta} - I_{t+\delta}\|^2. \end{aligned} \quad (6)$$

### C. Optimizing Camera Parameters

Due to vibration and assembly tolerance, the camera parameters would not stay constant across time in real-world cases. Following NeRF- [11], we jointly optimize the network model and the camera parameters. The extrinsic matrix locates in the  $SE(3)$  space and is not closed for addition operation. We optimize the parameters in  $se(3)$  space and transform them back to  $SE(3)$ , so that the parameters are valid during optimization. A vector in  $se(3)$  consists of a 3D translation vector  $\mathbf{n}$  and a 3D rotation vector  $\mathbf{r} = [r_x, r_y, r_z]$ . The transformation from a rotation vector to a rotation matrix is the *Rodrigues' Rotation Formula*, which is described by

$$\mathbf{R} = \mathbf{I} + \sin(\theta)\mathbf{K} + (1 - \cos(\theta))\mathbf{K}^2, \quad (7)$$

where  $\mathbf{I}$  is the identity matrix,  $\theta$  is the length of the rotation vector, and  $\mathbf{K}$  is the skew-symmetric matrix derived from  $\mathbf{r}$ .

## IV. EXPERIMENTS

### A. Experiment Setting

**Multiscopic Camera System** Pilot work [6], [7] uses a single camera mounted on a moving robotic arm to simulate a multi-scopic camera. Although this setting can guarantee identical intrinsic parameters for different views and proper extrinsic parameters, its application is limited to in-door scenarios. Based on the previous work, we built a portable 5-view real-world camera to enable casual and dynamic capturing. We manufactured an acrylic support to mount five SONY IMX 322 camera modules with almost the same configurations. We place one module on the *center* position and place other modules equally distant from the center module in the four directions: *top*, *bottom*, *left* and *right*. At present, we set the distance between adjacent modules 120mm. As we have reserved much space on the supporting board, the positions and number of modules could be adjusted according to practical needs.

**Synthetic Data Generator** Our customized real-world camera could capture five views at one time. However, we require more views to evaluate the performance of our NVS method quantitatively. The quality of real-world data suffers from device noise and improper exposure. Moreover, the depth information is usually difficult to estimate without dedicated sensors. To address the issue, we developed a simulated multi-view (no longer limited to 5 views) camera based on the well-known Habitat-sim simulator designed for embodied robotic tasks. The current works use the simulator mainly for

testing robots' performance in autonomous navigation and instruction following. Our work provides a more impressive and exciting application scenario for the simulator. To enable dynamic simulation, we keep the agent and the cameras still and place a chair inside the common viewing frustum of the cameras. The chair would move from the left side to the right side from the capturing perspective.

**Data Preparation** We collected a set of real-world and synthetic data from various scenes using the sensor and the simulator mentioned above. For synthetic data, we render 24 frames of data per scene. Each frame contains five images rendered using the 5-view layout for training. For real-world data, we captured 120 frames of 5-view data per scene. The image resolutions are  $1280 \times 720$  for synthetic data and  $640 \times 480$  for real data, respectively.

**Environment and Parameters** Our experiments are conducted on a server equipped with Intel Xeon 5218 CPU and Nvidia RTX 2080 Ti GPU. We implement our network model in Pytorch and the data reading interface in Python. We use the Adam optimizer to train the MLP network and the camera parameters for 1200 epochs. We fix the flow model and load the pretrained checkpoint <sup>2</sup> since we want to learn the temporal consistency embedded in the model. For any two consecutive frames, 4 intermediate frames are generated from the frame interpolation model for temporal consistency training. The learning rate is set at 0.001 initially and exponentially decays every 100 epochs unless reaching  $1e-5$ . We sample 6400 rays and 128 points along each ray during the training to render pixel colors for each iteration.

### B. Results

We conduct extensive experiments to demonstrate the ability of our method to achieve photo-realistic novel view and time synthesis. More importantly, we compare our approach to state-of-the-art flow- and NeRF-based dynamic novel view synthesis methods [9], [10] to show that our proposed approach achieves comparable performance.

**Illustration of Novel View and Time Synthesis** We present rendered images at unseen viewpoints and specified timesteps in Fig. 4. We synthesize four images for the fixed time and camera pose cases, respectively. The results illustrate our method's generalization ability in space and time domains. It proves that most fine-grained appearance details are well embedded in our space-time model. Moreover, we leverage the predicted density to render depth maps to show that our model can capture adequate geometry information. More results to visualize depth quality in complex scenes are listed in Fig. 5. Since we have no ground-truth depth supervising signal during the training process, our method could serve as an auxiliary tool for self-supervised depth estimation or multi-view stereo matching.

**Qualitative and Quantitative Comparison** We carefully select top-performing methods NSFF [10] and NeRFFlow [9] as the baseline to evaluate our model. The ideas of

<sup>2</sup>The implementation of the flow-based frame interpolation model is available at: <https://github.com/avinashpaliwal/Super-SloMo>

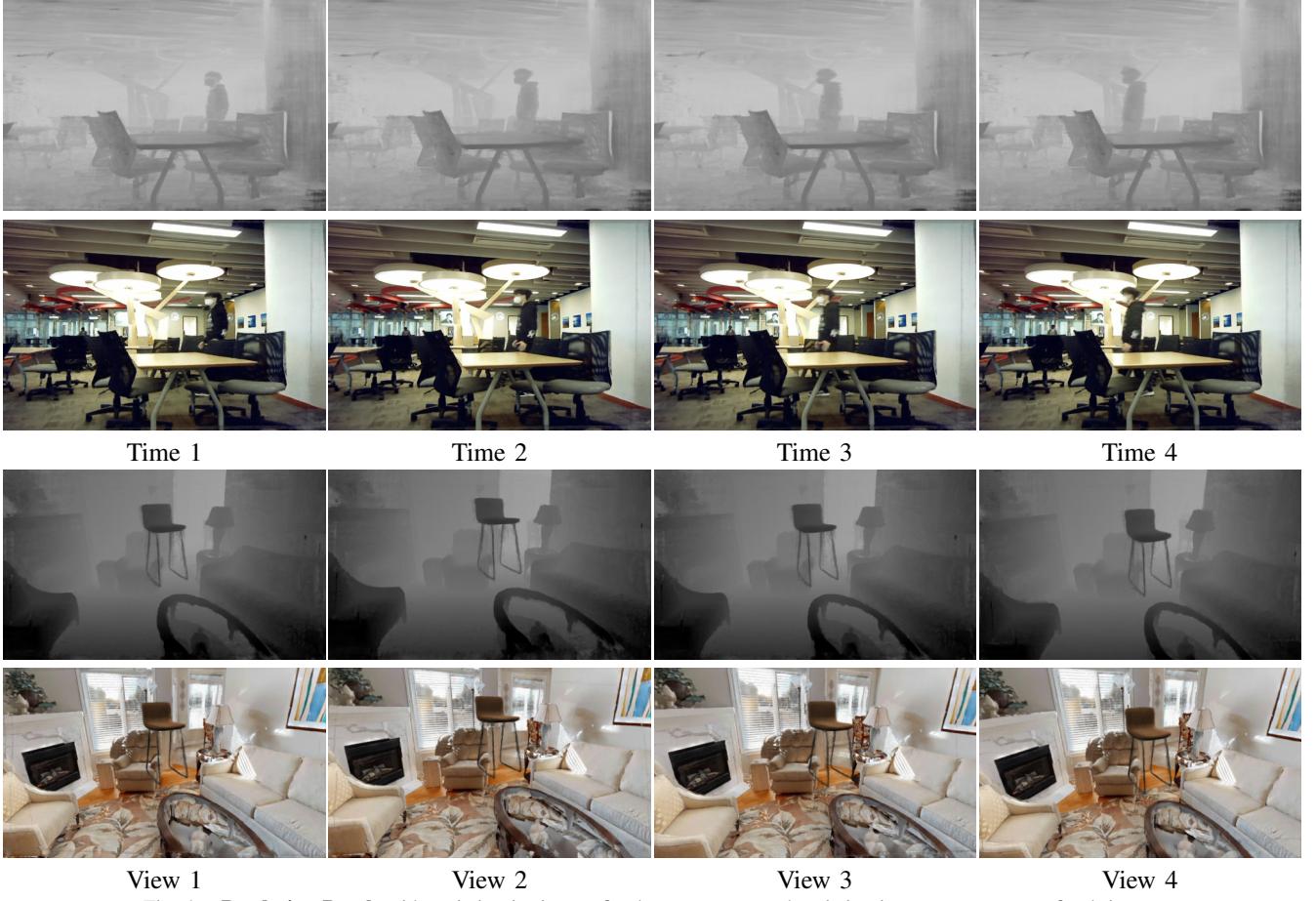


Fig. 4. **Rendering Result** with variation in time at fixed camera pose and variation in camera pose at a fixed time.



Fig. 5. **More results of depth maps rendered using our method from complex scenes.** We have no access to any ground-truth depth map, and thus, the estimation is completely performed in a self-supervised fashion.

their methods are similar to ours in that both represent the dynamic scene as a function of space-time coordinates. Nevertheless, all the methods propose a different mechanism to enforce coherent temporal learning. We train their models from scratch on our data, following the instructions from the officially published repositories. All parameters are set as default. For quantitative evaluation, we report two error metrics describing the similarity between the rendered images and the ground-truth images: structural similarity index measure (SSIM) and peak signal-to-noise ratio (PSNR). The metric statistics are shown in Table III, IV, which demonstrates that our method outperforms the baseline methods in all metrics.

TABLE I  
QUANTITATIVE EVALUATION ON SYNTHETIC DATA

Method	PSNR $\uparrow$	SSIM $\uparrow$
NeRFflow [9]	27.32	0.75
NSFF [10]	27.31	0.73
Ours	<b>33.06</b>	<b>0.88</b>

TABLE II  
QUANTITATIVE EVALUATION ON REAL DATA

Method	PSNR $\uparrow$	SSIM $\uparrow$
NeRFflow [9]	27.61	0.66
NSFF [10]	26.39	0.64
Ours	<b>29.68</b>	<b>0.75</b>

We perform qualitative analysis on both synthetic and real datasets to find where our method takes effect. Fig. 6 and Fig. 7 illustrate the rendering results. The synthetic results show that all the methods can reconstruct the static environment background, which demonstrates the powerful capability of NeRF model on static novel view synthesis. However, NSFF is bad at rendering small or thin foreground objects, e.g., the support of the chair. NeRFflow preserves the global content better while bringing more noisy points. Our rendered images avoid such artifacts and become more photo-realistic. For real-world cases, we compare performance in typical indoor and outdoor scenes. NeRFflow and NSFF show similar results as those of synthetic cases. It



Fig. 6. **Qualitative comparisons on synthetic data.** The dynamic scenes are generated following the same motion pattern: a chair moves from the left to the right. The reconstructed images are rendered from a randomized pose at a randomized time step. We compare our method with NeRFFlow [9] and NSFF [10]. All methods are able to render high-quality static background scenes, while our method could reconstruct the moving chair better with less blurry artifacts and reconstruction failures.

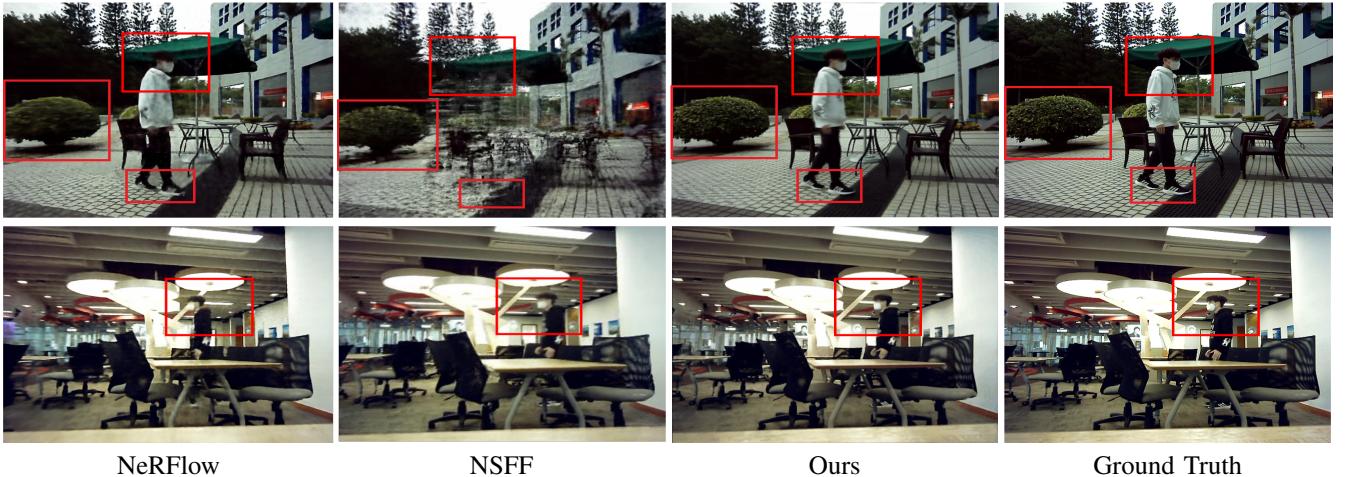


Fig. 7. **Qualitative comparisons on real data.** To capture the data, we fix the multi-sensor camera and make a person move in front of the background scenes. We render the novel images following the same settings of synthetic scenarios. The grass area and the human head area are marked by red rectangles to show that our method can render finer details.

TABLE III  
ABLATION STUDY ON SYNTHETIC DATA

Method	PSNR $\uparrow$	SSIM $\uparrow$
NeRF (without time)	24.13	0.78
Ours without optimization	29.25	0.74
Ours without $\mathcal{L}_t$	32.43	0.70
<b>Ours</b>	<b>33.06</b>	<b>0.88</b>

should be noted that both baseline methods have failure cases. NSFF fails at outdoor scenes and generates messy and chaotic foreground objects, while NeRFFlow fails at scene boundary. We attribute the artifacts to the improper estimation of camera poses and motion fields during training, highlighting the necessity of our two proposed refining modules.

**Ablation Study** We evaluate the impact of the proposed modules individually in the task of dynamic novel view synthesis by removing (1) latent time representation (NeRF (without time)); (2) optimizing camera parameters (without optim); (3) the temporal consistency loss (without  $\mathcal{L}_t$ ). The results, as given in Table III and IV, reveal the relative

TABLE IV  
ABLATION STUDY ON REAL DATA

Method	PSNR $\uparrow$	SSIM $\uparrow$
NeRF (without time)	21.51	0.63
Ours without optimization	28.58	0.74
Ours without $\mathcal{L}_t$	26.17	0.59
<b>Ours</b>	<b>29.68</b>	<b>0.75</b>

significance of each component, with the entire system doing the best. We observed that each component of our approach contributed to a different improvement in the outcome. We found that without the reinforcement of temporal consistency, image deterioration becomes considerable, resulting in a large drop in SSIM value and a slight decrease in perceptual similarity. The decline in SSIM value and perceptual similarity can be attributed to the model's failure to disentangle space-temporal information without the aid of flow model. On the other hand, we discovered that time encoding is essential in obtaining a decent PSNR result since the model might fail to query frames accurately. Without appropriate latent time representation, NeRF model tends to ignore dynamic part of the scene and focus on optimizing

the static part only. This causes additional noise and inconsistency in the dynamic scene, resulting in a low PSNR. Furthermore, because the fraction of dynamic objects is tiny in comparison to the whole picture, the SSIM value drops little owing to the degradation of the dynamic component.

## V. CONCLUSION

In this work, we build a portable multiscopic camera sensor for novel view and time synthesis using an end-to-end NeRF-based model that takes synchronized input views from the multiscopic camera. To evaluate our model, we built a synthetic dataset as well as a real-world dataset with dynamic scenes rendered by a multiscopic camera. Our NeRF-based model learns a mapping from a space and time representation to radiance and density for rendering a dynamic scene. We conduct an extensive evaluation on both synthetic and real-world datasets and the results show that our model outperforms alternative solutions by a large margin. With our customized NeRF-based model, we hope our new multiscopic camera designed for novel view and time synthesis can be a practical module in robot applications.

## REFERENCES

- [1] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, “Nerf: Representing scenes as neural radiance fields for view synthesis,” in *ECCV*, 2020.
- [2] K. Chitta, A. Prakash, and A. Geiger, “Neat: Neural attention fields for end-to-end autonomous driving,” in *ICCV*, 2021.
- [3] M. Tancik, V. Casser, X. Yan, S. Pradhan, B. Mildenhall, P. P. Srinivasan, J. T. Barron, and H. Kretzschmar, “Block-nerf: Scalable large scene neural view synthesis,” *arXiv:2202.05263*, 2022.
- [4] K. Park, U. Sinha, P. Hedman, J. T. Barron, S. Bouaziz, D. B. Goldman, R. Martin-Brualla, and S. M. Seitz, “Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields,” *arXiv:2106.13228*, 2021.
- [5] K. Park, U. Sinha, J. T. Barron, S. Bouaziz, D. B. Goldman, S. M. Seitz, and R. Martin-Brualla, “Nerfies: Deformable neural radiance fields,” in *ICCV*, 2021.
- [6] W. Yuan, R. Fan, M. Y. Wang, and Q. Chen, “Mfusenet: Robust depth estimation with learned multiscopic fusion,” *IEEE RA-L*, 2020.
- [7] W. Yuan, Y. Zhang, B. Wu, S. Zhu, P. Tan, M. Y. Wang, and Q. Chen, “Stereo matching by self-supervision of multiscopic vision,” in *IROS*, 2021.
- [8] A. Pumarola, E. Corona, G. Pons-Moll, and F. Moreno-Noguer, “D-nerf: Neural radiance fields for dynamic scenes,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 318–10 327.
- [9] Y. Du, Y. Zhang, H.-X. Yu, J. B. Tenenbaum, and J. Wu, “Neural radiance flow for 4d view synthesis and video processing,” in *ICCV*, 2020.
- [10] Z. Li, S. Niklaus, N. Snavely, and O. Wang, “Neural scene flow fields for space-time view synthesis of dynamic scenes,” in *CVPR*, 2021.
- [11] Z. Wang, S. Wu, W. Xie, M. Chen, and V. A. Prisacariu, “Nerf-: Neural radiance fields without known camera parameters,” *arXiv:2102.07064*, 2021.
- [12] H. Jiang, D. Sun, V. Jampani, M.-H. Yang, E. Learned-Miller, and J. Kautz, “Super slomo: High quality estimation of multiple intermediate frames for video interpolation,” in *CVPR*, 2018.
- [13] S. M. Seitz and C. R. Dyer, “Photorealistic scene reconstruction by voxel coloring,” *IJCV*, 1999.
- [14] T. Zhou, R. Tucker, J. Flynn, G. Fyffe, and N. Snavely, “Stereo magnification: Learning view synthesis using multiplane images,” in *SIGGRAPH*, 2018.
- [15] B. Mildenhall, P. P. Srinivasan, R. Ortiz-Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar, “Local light field fusion: Practical view synthesis with prescriptive sampling guidelines,” *TOG*, 2019.
- [16] P. P. Srinivasan, R. Tucker, J. T. Barron, R. Ramamoorthi, R. Ng, and N. Snavely, “Pushing the boundaries of view extrapolation with multiplane images,” in *CVPR*, 2019.
- [17] E. Penner and L. Zhang, “Soft 3d reconstruction for view synthesis,” in *SIGGRAPH Asia*, 2017.
- [18] A. Kar, C. Häne, and J. Malik, “Learning a multi-view stereo machine,” in *NeurIPS*, 2017.
- [19] T. Porter and T. Duff, “Compositing digital images,” in *SIGGRAPH*, 1984.
- [20] H. Ouyang, B. Zhang, P. Zhang, H. Yang, J. Yang, D. Chen, Q. Chen, and F. Wen, “Real-time neural character rendering with pose-guided multiplane images,” *arXiv preprint arXiv:2204.11820*, 2022.
- [21] M. Niemeyer, L. Mescheder, M. Oechsle, and A. Geiger, “Occupancy flow: 4d reconstruction by learning particle dynamics,” in *ICCV*, 2019.
- [22] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, “Occupancy networks: Learning 3d reconstruction in function space,” in *CVPR*, 2019.
- [23] Q. Wang, Z. Wang, K. Genova, P. Srinivasan, H. Zhou, J. T. Barron, R. Martin-Brualla, N. Snavely, and T. Funkhouser, “Ibrnet: Learning multi-view image-based rendering,” in *CVPR*, 2021.
- [24] K. Park, U. Sinha, J. T. Barron, S. Bouaziz, D. B. Goldman, S. M. Seitz, and R. Martin-Brualla, “Deformable neural radiance fields,” *arXiv preprint arXiv:2011.12948*, 2020.
- [25] R. Martin-Brualla, N. Radwan, M. S. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth, “Nerf in the wild: Neural radiance fields for unconstrained photo collections,” in *CVPR*, 2021.
- [26] C. Reiser, S. Peng, Y. Liao, and A. Geiger, “Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps,” in *ICCV*, 2021.
- [27] J. Chen, Y. Zhang, D. Kang, X. Zhe, L. Bao, X. Jia, and H. Lu, “Animatable neural radiance fields from monocular rgb videos,” *arXiv:2106.13629*, 2021.
- [28] S. Peng, Y. Zhang, Y. Xu, Q. Wang, Q. Shuai, H. Bao, and X. Zhou, “Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans,” in *CVPR*, 2021.
- [29] W. Xian, J.-B. Huang, J. Kopf, and C. Kim, “Space-time neural irradiance fields for free-viewpoint video,” in *CVPR*, 2021, pp. 9421–9431.
- [30] B. Attal, E. Laidlaw, A. Gokaslan, C. Kim, C. Richardt, J. Tompkin, and M. O’Toole, “Törf: Time-of-flight radiance fields for dynamic scene view synthesis,” *NeurIPS*, 2021.
- [31] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge University Press, 2003.
- [32] B. Wilburn, N. Joshi, V. Vaish, E.-V. Talvala, E. Antunez, A. Barth, A. Adams, M. Horowitz, and M. Levoy, “High performance imaging using large camera arrays,” in *SIGGRAPH*, 2005.
- [33] M. Broxton, J. Flynn, R. Overbeck, D. Erickson, P. Hedman, M. Duval, J. Dourgarian, J. Busch, M. Whalen, and P. Debevec, “Immersive light field video with a layered mesh representation,” in *SIGGRAPH*. ACM New York, NY, USA, 2020.
- [34] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niebner, M. Savva, S. Song, A. Zeng, and Y. Zhang, “Matterport3d: Learning from rgbd data in indoor environments,” in *3DV*. IEEE, 2017.
- [35] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese, “3d semantic parsing of large-scale indoor spaces,” in *CVPR*, 2016.
- [36] X. Puig, K. Ra, M. Boben, J. Li, T. Wang, S. Fidler, and A. Torralba, “Virtualhome: Simulating household activities via programs,” in *CVPR*, 2019.
- [37] Y. Wu, Y. Wu, G. Gkioxari, and Y. Tian, “Building generalizable agents with a realistic and rich 3d environment,” *arXiv:1801.02209*, 2018.
- [38] F. Xia, A. R. Zamir, Z. He, A. Sax, J. Malik, and S. Savarese, “Gibson env: Real-world perception for embodied agents,” in *CVPR*, 2018.
- [39] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, *et al.*, “Habitat: A platform for embodied ai research,” in *ICCV*, 2019.
- [40] A. Szot, A. Clegg, E. Undersander, E. Wijmans, Y. Zhao, J. Turner, N. Maestre, M. Mukadam, D. Chaplot, O. Maksymets, A. Gokaslan, V. Vondrus, S. Dharur, F. Meier, W. Galuba, A. Chang, Z. Kira, V. Koltun, J. Malik, M. Savva, and D. Batra, “Habitat 2.0: Training home assistants to rearrange their habitat,” *arXiv:2106.14405*, 2021.
- [41] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, “Flownet 2.0: Evolution of optical flow estimation with deep networks,” in *CVPR*, 2017.
- [42] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *MICCAI*, 2015.