2022 IEEE International Conference on Robotics and Automation (ICRA)
May 23-27, 2022. Philadelphia, PA, USA

Flow Supervised Neural Radiance Fields for Static-Dynamic Decomposition

Quei-An Chen and Akihiro Tsukada
DENSO Corporation

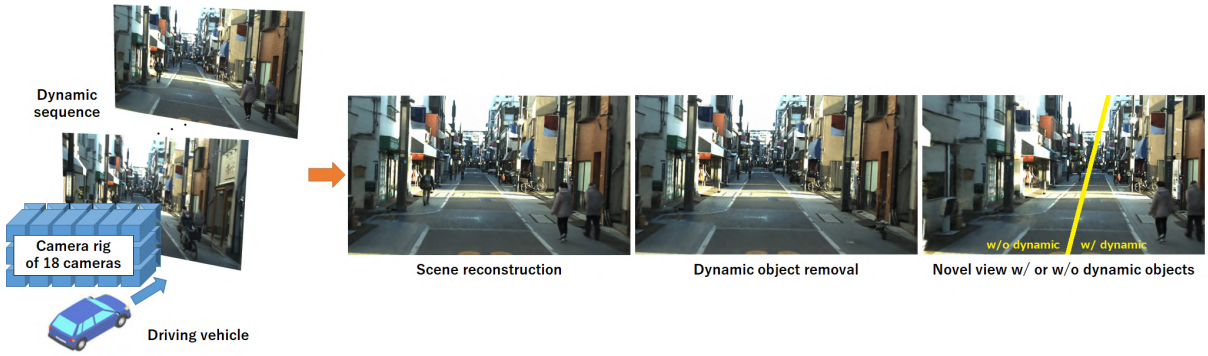


Fig. 1: From a multi-view dynamic sequence taken from a driving vehicle, we are able to 1. reconstruct the dynamic scene, 2. remove dynamic objects and 3. render consistent novel views with or without dynamic objects.

***Abstract*— We present an approach to synthesize novel views from dynamics scenes captured by multi-view videos of cameras mounted on a driving vehicle. We unify existing methods and propose a new training loss to explicitly disentangle the static background from the dynamic foreground objects using scene flow's magnitude, learnt only from proxy 2D optical flow supervision. We obtain high quality static and dynamic contents separately, which allow us to combine them freely for novel view and time syntheses. We establish a dataset consisting of 5 dynamic scenes with varying difficulties on which we conduct experiments, and show that our method is able to handle challenging scenarios in real-world traffics and create high quality novel view and time syntheses.**

I. Introduction

Novel view synthesis, built on scene reconstruction, is a challenging research topic that has a wide variety of applications such as bullet time effects, video stabilization, mixed reality, and so on. In the domain of autonomous driving, it can serve as a tool to generate novel views used in data augmentation, therefore boost the performance of other sensing algorithms such as object detection or semantic segmentation. However, most of the current approaches make the assumption that the scene is static, hence are not directly applicable on video sequences taken from a driving vehicle, where not only the scene - containing multiple active road users - is moving, but also the camera mounted on the egocar is moving.

Since the pioneer work of [1], the research community starts to follow their idea to parametrize the scene using deep neural networks (MLPs) and render the images using volume rendering techniques. To extend this framework to handle scene dynamicness, several follow-up methods have been proposed recently. [2] [3] [4] model the dynamicness by learning displacement fields of each 3D points across time, but the experiments are only done with human faces or small scale objects. Other approaches are experimented on large real world scenes. For example, NeRF-W [5] attempts to ignore the dynamic objects and learn the static background that is present in all images. NSFF [6] and DyNeRF [7] learn the same scene with continuous time, therefore are able to reconstruct the dynamic objects correctly by exploiting either the adjacent views or multi-views of the same timestamp.

Parallel to the problem of reconstructing dynamic scenes, another essential question is whether it is possible to identify the dynamic region of the scene, and disentangle it from the background. This allows the user to freely combine static background and dynamic contents from different timestamps, thus enables a wider possibility of novel view syntheses. To our surprise, only a few works have addressed this problem. NeRF-W and NSFF use two networks to separately estimate the static and dynamic contents, and use different compositing strategies to learn each component. [8] estimates rigidity of the scene, and removes the rigid part to reconstruct the background. [9] learns on KITTI [10], an autonomous vehicle dataset, and explicitly uses ground truth 3D bounding boxes to disentangle the background from the road users.

In this work, we propose to disentangle the dynamic road users from the static background by integrating the networks of prior works. More precisely, we design separate networks for static and dynamic parts as in NeRF-W and learn the scene flow following NSFF. To let the networks learn the decomposition, we use the learnt scene flow's magnitude and encourage the networks to reconstruct contents with small

978-1-7281-9680-0/22/$31.00 ©2022 IEEE

flow magnitude by the static network.

Existing self-driving datasets [11] [12] focus on surrounding perception, therefore, the cameras have small field of view overlap, and are not necessarily synchronized. To favor our need of dynamic scene reconstruction, we collect an multi-view dataset where the cameras are perfectly synchronized and all front-facing (large field of view overlap) to conduct initial experiments. We evaluate our approach on this dataset that exhibits a variety of dynamic contents. Our results show robust performance not only on reconstructing the scene, but also on the ability of correctly separating the foreground from the background.

To sum up, our key contributions consist of:

1) Being one of the first works that experiment on the possibility to apply NeRF-like methods on driving scenes with long egocar movement.
2) Unifying previous works to maximize the performance on dynamic scenes and designing an effective 2-stage training strategy with a dynamic regularization loss to separate static/dynamic regions with only proxy 2D optical flow supervision.

II. Related Work

A. Static scene reconstruction

Many traditional approaches first build a coarse geometry such as point cloud or mesh, then render this representation from novel view [13] [14]. Another effective representation is multiplane images (MPIs) [15] [16] [17] [18], which has shown promising results on real world applications. These methods are generally experimented on small scale objects or with small camera motions.

Recently, deep learning based neural rendering emerges as another promising direction. [19] [20] [21] use voxels to represent the scene, and render them using ray-tracing techniques. [22] [23] [24] share the same voxelized representation, but use CNNs to perform the final rendering. NeRF [1] on the other hand, does not voxelize the scene, but represent the radiance field as a continuous function conditioned on position and viewing direction. While the above methods have shown impressive results, they cannot handle temporal scene changes due to the network design.

B. Dynamic scene reconstruction

To handle the temporal change of the scene, novel network architecture extensions of [1] have been proposed. [5] simply puts less weight on the dynamic object reconstruction, while [7] uses multi-view images to reconstruct dynamic region. [6] even eliminates the need of multi-view images by exploiting the dynamic object's scene flow that allows the network to learn from adjacent frames. Finally, [9] demonstrates that if accurate 3D bounding boxes of the dynamic objects are provided, the network is able to learn both the scene reconstruction and the representation of each individual object.

C. Static-dynamic decomposition

Nevertheless, among the above methods that succeed in large-scale real world dynamic scene reconstruction, to our knowledge, except [9] which is supervised by ground truth 3D bounding boxes of dynamic objects, none has been able to both disentangle the static region from the dynamic one and reconstruct them in detail successfully. NeRF-W succeeds in reconstructing the background, but fails on dynamic content reconstruction due to lack of consistent views. While NSFF proposes a blending scheme to blend static and dynamic parts, its network fails to fully disentangle them, resulting in a meaningless static/dynamic separation. On the other hand, DyNeRF simply learns everything as dynamic and hence is not able to separate static and dynamic parts in its nature.

III. Background

Before we describe our proposed method, we give a brief description of the core components that we adopt based on previous works NeRF-W and NSFF.

A. NeRF-W

NeRF-W represents the scene with static and transient (which we refer to as "dynamic") heads. The static head S has two components: the first S_1 is a simple extension of NeRF that takes as input positionally-encoded [1] $\mathbf{x}$ and outputs a volumetric density σ_s. The second S_2 aims at capture time- and view-depending colors, so takes as input positionally-encoded viewing direction $\mathbf{d}$ along with a latent appearance encoding $\mathbf{a_t}$ depending on the time, and outputs RGB color $\mathbf{c_s}(t)$:

$$\sigma_s = S_1(\mathbf{x}) \tag{1}$$

$$\mathbf{c_s}(t) = S_2(\mathbf{x}, \mathbf{d}, \mathbf{a_t}) \tag{2}$$

The dynamic head D shares similar structure as S_2, but instead of $\mathbf{d}$ and $\mathbf{a_t}$, takes input $\mathbf{z_t}$ called transient embedding that represents the dynamic content of the each time instance. The outputs is the dynamic content's color and volumetric density, along with a pixel color variance measure $\beta(t)$:

$$(\mathbf{c_d}(t), \sigma_d(t), \beta(t)) = D(\mathbf{x}, \mathbf{z_t}) \tag{3}$$

To render an image of time t, volume rendering equation is used to compose the static and dynamic heads on all pixels. Let $\mathbf{r} = \mathbf{o} + p\mathbf{d}$ be a camera ray originating from the camera center and passes by a pixel. The expected color $\hat{\mathbf{C}}_\mathbf{r}(t)$ of that pixel is given by:

$$\hat{\mathbf{C}}_\mathbf{r}(t) = \int_{p_n}^{p_f} T_p(t)(\sigma_s \mathbf{c_s}(t) + \sigma_d(t)\mathbf{c_d}(t))dp \tag{4}$$

where

$$T_p(t) = \exp(-\int_{p_n}^{p} (\sigma_s + \sigma_d(t))ds) \tag{5}$$

is the accumulated opacity along the ray up to plane p, and p_n, p_f are the near and far planes between which we render the ray. In practice, the volume rendering integral is approximated by quadrature rule as in [1].

The optimization is done via minimizing the difference between the ground truth color $\mathbf{C}_\mathbf{r}(t)$ and the predicted color

$\hat{\mathbf{C}}_\mathbf{r}(t)$, along with regularization on β_t and dynamic densities $\sigma_d(t)$. More specifically, a transient weight regularization is added to avoid everything being explained as transient:

$$\mathcal{L}_\tau = \sum_{k=1}^{K} \sigma_d^k(t) \tag{6}$$

where the superscript k corresponds to each sample point on the ray.

B. NSFF

The main contribution of NSFF is to learn the forward and backward scene flows so that images from adjacent frames can be leveraged to reconstruct the scene. Its core dynamic model is defined as:

$$(\mathbf{c_d}(t), \sigma_d(t), F_t, W_t) = D_\phi(\mathbf{x}, t) \tag{7}$$

where t is positionally-encoded time, $F_t = (\mathbf{f}_{t\to t+1}, \mathbf{f}_{t\to t-1})$ are the forward and backward 3D scene flows and $W_t = (w_{t\to t+1}, w_{t\to t-1})$ are disocclusion weights. In addition to traditional NeRF's loss of time t, it optimizes the flow and adjacent frames' RGBσ jointly by computing warped images:

$$\hat{\mathbf{C}}_\mathbf{r}(j\to t) = \int_{p_n}^{p_f} T_p(j)\sigma_d(t\to j)\mathbf{c_d}(t\to j)dp \tag{8}$$

where $\mathbf{c_d}(t\to j)$ and $\sigma_d(t\to j)$ are the outputs by querying $D_\phi(\mathbf{x}+\mathbf{f}_{t\to j}, \mathbf{z_j})$, and $j \in \mathcal{N}(t) = \{t-1, t+1\}$ are adjacent frames.

The difference between the warped images and the ground truth images at each timestamps are added to the loss. Another core loss to correctly guide the scene flow is the geo consistency loss that minimizes the projected 3D scene flow and the proxy 2D optical flow precomputed using observation images:

$$\mathcal{L}_{geo} = \sum_{\mathbf{r}} \sum_{j\in\mathcal{N}(t)} \|\mathbf{p_r}(t\to j) - \hat{\mathbf{p}}_\mathbf{r}(t\to j)\|_1 \tag{9}$$

where $\mathbf{p_r}(t\to j)$ is the proxy 2D optical flow for ray $\mathbf{r}$ and $\hat{\mathbf{p}}_\mathbf{r}(t\to j)$ is the image projection of $\hat{\mathbf{X}}_\mathbf{r}(t) + \hat{\mathbf{F}}_\mathbf{r}(t\to j)$, the ray termination point and the expected scene flow computed using volume rendering equation. Along with other regularization terms, NSFF is able to learn the scene color and the flows correctly.

IV. Proposed Method

We unify the works described in the previous section, make some modifications to adapt to our settings and to improve the performance.

A. Network structure

Our network architecture is visualized in Fig. 2, which integrates the core components of NeRF-W and NSFF. To simplify the network, we remove the appearance embedding input, the uncertainty output in NeRF-W and the disocclusion weight output in NSFF.

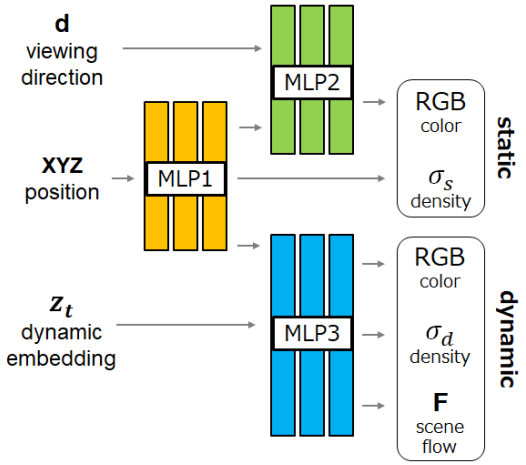


Fig. 2: Network structure adapted from NeRF-W and NSFF.

Same as [1], we train two copies of the same network (coarse and fine networks), except that the flow head $\mathbf{F}$ only exists in the fine network.

B. Optimization

To allow the network to disentangle static and dynamic parts, we want the static content to be learnt solely by the static network and the dynamic content to be learnt solely by the dynamic network, and when combined, the final rendering matches the actual observation.

Our approach is to learn the 3D scene flow by proxy 2D optical flow as in NSFF, and use the learnt flow's magnitude to identify the moving region then minimize the dynamic density σ_d in this region. Afterwards, since this region can no longer be explained by the dynamic network, in order to reconstruct the scene to match the observation, the model is forced to learn it via the static network, thus making the correct static-dynamic decomposition.

However, we do not directly train the flow from the beginning since the image warping (Eq. 8) costs three times the normal rendering process, resulting in intractable training time. With our multi-view setting, the dynamic objects can be considered "static" from all images of the same timestamp, and be reconstructed to some extent with traditional NeRF setting. Therefore, there is no need to exploit the flow from the beginning. Instead, we propose a two-stage training process, where at the first stage, the network except the flow output is trained from multi-view images, and at the second stage, we include the flow in training and use the flow's magnitude to train the network to separate static and dynamic regions.

1) Stage 1: The scene is considered "static" at each timestamp and is trained from the images solely. We initially followed the loss design as in NeRF-W, but found that the transient weight regularization (Eq. 6) makes the network fully ignore the dynamic objects (i.e. everything is learnt as static) in some scenes. Therefore, we decide to remove this regularization, and use conventional NeRF loss computed on the color only:

$$\mathcal{L}_{color} = \sum_{\mathbf{r},t} \|\mathbf{C}_\mathbf{r}(t) - \hat{\mathbf{C}}^c_\mathbf{r}(t)\|^2 + \|\mathbf{C}_\mathbf{r}(t) - \hat{\mathbf{C}}^f_\mathbf{r}(t)\|^2 \quad (10)$$

where the superscripts c and f correspond to coarse and fine network outputs.

After this stage, we observe that the scene reconstruction already reaches high quality in both static and dynamic regions. However, without explicit supervision, the model does not know the correct static-dynamic decomposition, and tends to output almost everything as dynamic. Thus, we enter the next stage to train the model with flow, and use its magnitude to minimize the dynamic weight on static regions.

2) Stage 2: After the scene (both static and dynamic) is reconstructed to some extent, we start supervising the network with proxy 2D optical flow to train 3D scene flow. More specifically, we adopt from NSFF the photometric loss,

$$\mathcal{L}_{pho} = \sum_{\mathbf{r},t} \sum_{j \in \mathcal{N}(t)} \|\mathbf{C}_\mathbf{r}(t) - \hat{\mathbf{C}}_\mathbf{r}(j \to t)\|^2 \quad (11)$$

the cycle loss,

$$\mathcal{L}_{cyc} = \sum_{\mathbf{x_t}} \sum_{j \in \mathcal{N}(t)} \|\mathbf{f}_{t \to j}(\mathbf{x_t}) + \mathbf{f}_{j \to t}(\mathbf{x_{t \to j}})\|_1 \quad (12)$$

the geo loss (Eq. 9) and the flow regularization losses.

$$\begin{aligned}\mathcal{L}_{reg} = &\sum_{\mathbf{x_t}} \|\mathbf{f}_{t \to t+1}(\mathbf{x_t}) + \mathbf{f}_{t \to t-1}(\mathbf{x_t})\|_1 \\ &+ \sum_{\mathbf{x_t}} \sum_{j \in \mathcal{N}(t)} \|\mathbf{f}_{t \to j}(\mathbf{x_t})\|_1\end{aligned} \quad (13)$$

In this stage, we add our proposed dynamic regularization loss that helps to separate static/dynamic regions. The core idea is to minimize the dynamic density σ_d if the scene flow's magnitude is lower than a preselected threshold F_{thr}.

First, each ray's scene flow magnitude is defined as the sum of the magnitudes of both the forward and backward scene flows. Note we convert the scene flow values from NDC (Normalized Device Coordinates) back to Euclidean space to avoid inconsistent values in different locations of the space.

$$\|\hat{\mathbf{F}}_\mathbf{r}\| = \sum_{j \in \mathcal{N}(t)} \|\hat{\mathbf{F}}_\mathbf{r}(t \to j)\|_1 \quad (14)$$

To eliminate the effect of bad scene flow values on large textureless regions such as the sky, we explicity zero the scene flow if the estimated depth $\hat{\mathbf{Z}}_\mathbf{r}$ (through volume rendering) is larger than 0.95, i.e.

$$\|\hat{\mathbf{F}}_\mathbf{r}\| := 0 \quad \text{if} \quad \hat{\mathbf{Z}}_\mathbf{r} > 0.95 \quad (15)$$

Then our dynamic regularization loss is formulated as:

$$\mathcal{L}_{dy} = \frac{1}{K} \sum_{\mathbf{r}, \|\hat{\mathbf{F}}_\mathbf{r}\| \le F_{thr}} \sum_{k=1}^{K} \sigma_d^k(t) \quad (16)$$

Theoretically, larger thresholds (i.e. eliminating more dynamic densities) lead to more similar performance as NeRF-W. In particular, when F_{thr} is large enough, $\mathcal{L}_{dy}$ equals $\mathcal{L}_\tau$ in NeRF-W, meaning that we minimize rays' dynamic densities no matter how much they move in the scene.

Finally, our loss in the second stage is:

$$\begin{aligned}\mathcal{L} = \mathcal{L}_{color} + \mathcal{L}_{pho} + \mathcal{L}_{cyc} + \lambda_{geo}\mathcal{L}_{geo} \\ + \lambda_{reg}(\mathcal{L}_{reg} + \mathcal{L}_{dy})\end{aligned} \quad (17)$$

V. Experiments

A. Data capture

We build a camera rig of 18 FLIR cameras arranged in a 6×3 grid mounted on a vehicle. Fig. 3 shows the layout of all the cameras and their corresponding ids. The cameras have 1920×1080 resolution with small lens distortion.

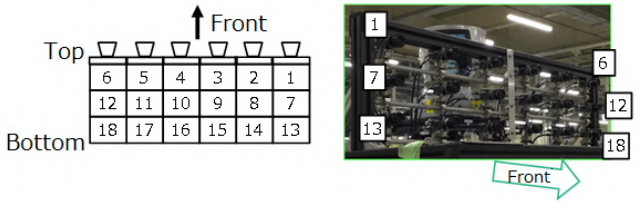


Fig. 3: The layout of our camera rig of 18 cameras.

B. Dataset

We manually inspect the images and establish a dataset of 5 dynamic scenes to test the limit of our method. The detailed description can be found in the supplementary video.

To identify reference dynamic regions, we use off-the-shelf semantic segmentation [25] trained on Cityscapes [26] to generate motion masks.

We use COLMAP [27] [14] to reconstruct the intrinsics (shared between cameras) and extrinsics (without camera rig constraint across time) using automatic reconstruction with the dynamic region masked out during feature extraction and matching. Finally, we use RAFT [28] to compute 2D optical flows between adjacent frames of the same cameras.

C. Quantitative evaluation

To make a fair comparison, we adopt the following training and evaluation protocol: the image size is set to 480×270 (1/4 scale). The training is performed on all cameras **except the 10th** and on all frames. The evaluation is done on camera 10, where the average metrics are computed between the generated images and the resized ground truth images.

1) Baselines and metrics: We compare our approach to the following baselines that can train on dynamic sequences:
NSFF: [6] with default settings without motion mask hard example mining. Since the official code only works for monocular sequences, we only use images from camera 9 for training.
NeRF+T: A straightforward extension of NeRF by adding the encoded time index as input. We adopt latent code

GT NSFF [6] NeRF+T NeRF-W [5] Ours-full

Fig. 4: Qualitative scene reconstruction results. Red bounding boxes are regions where the method clearly fails.

embedding with the same size N_τ as our approach instead of positional embedding for t.

NeRF-W: [5] without appearance encoding. The implementation is based on [29] since there is no official code.

We report the performance of each approach with three standard metrics: peak signal-to-noise ratio (PSNR), structural similarity index measure (SSIM) ([30]'s implementation with window size = 11) and perceptual similarity LPIPS [31] (using alexnet). The scores are computed on full images as well as on dynamic regions only.

2) Evaluation scores: The scene reconstruction scores are summarized in Table I. Ours-stage1 and Ours-full show similar performance, both reach better quality than other baselines. Further observations and discussions are described in the following qualitative evaluation section.

D. Qualitative evaluation

We visualize the scene reconstruction, the background content without dynamic objects as well as novel view and time syntheses.

Scene reconstruction. Fig. 4 shows 3 scene reconstruction results. NSFF trains only on monocular sequences, so the scene reconstruction is not as performant as other methods, but dynamic objects are quite well handled. Vanilla NeRF+T fails at capturing scene movement. NeRF-W achieves good quality when the scene dynamicness is large enough, but tends to ignore small moving regions and explain them as static (the second row), thus resulting in artifacts in dynamic regions. Our method achieves good scene reconstruction, although still missing details when the scene dynamicness is complicated.

Dynamic object removal. For methods that are able to separate static/dynamic, we visualize the background-only image by rendering only the static network, and output a dynamicness measure which estimates how much a pixel is likely to belong to a moving object. For NeRF-W and our methods, the dynamicness is calculated as the sum of dynamic weights $\{w_i^d\}$, and for NSFF, it is calculated as the average blending weight. The results are in Fig. 5.

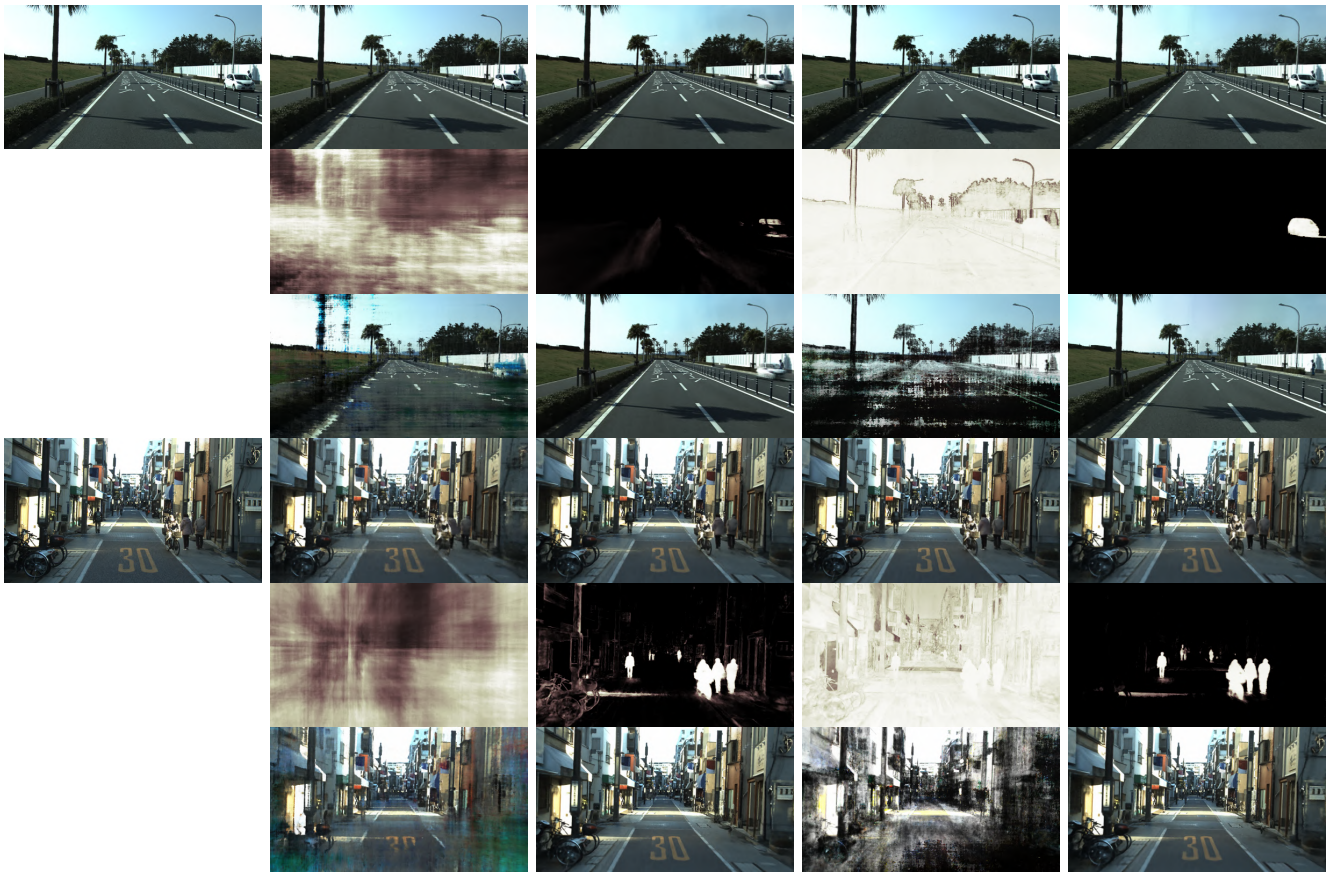
Methods	Dynamic Only			Full		
	PSNR(↑)	SSIM(↑)	LPIPS(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)
NSFF[1]	18.69	0.935	0.101	24.90	0.938	0.105
NeRF+T	19.99	0.938	0.099	28.25	0.950	0.090
NeRF-W	23.57	0.959	0.055	29.17	0.960	0.061
Ours-stage1	25.53	0.962	**0.047**	**30.50**	**0.964**	**0.052**
Ours-full	**26.28**	**0.963**	0.048	30.22	**0.964**	**0.052**

TABLE I: Quantitative results of **all scenes** averaged on our dataset.

NeRF-W succeeds in disentangling the dynamic objects if they occupy a large portion of the image, but fails if the dynamic region is small (the second row), in which case it tends to learn the region as static, resulting in low dynamicness and low image quality. Removing the transient weight regularization (Eq. 6) from NeRF-W is Ours-stage1, but the model fails entirely to decompose the scene, and learns nearly everything as dynamic. NSFF shows the same tendency with high dynamicness almost everywhere and bad background reconstruction. Our full method, however, disentangles well the dynamic region from the static one in all situations without reducing image quality, and the dynamicness obtained is cleaner than that from NeRF-W, including non-trivial decompositions such as object shadows.

Novel view and time syntheses. In addition to traditional novel camera view and dynamic object removal, we are able to synthesize more sophisticated bullet time effect images that mix different views and times, summarized in Fig. 6. We encourage the reader to see the supplementary material where we include more video results, including not only synthesized images but also visualizations of learnt geometry and scene flow.

[1]NSFF does not converge on the fourth scene possibly due to its long length, hence we train two models for the first half and the second half separately.



GT NSFF [6] NeRF-W [5] Ours-stage1 Ours-full

Fig. 5: Qualitative dynamic object removal results. From top to bottom for each scene: scene reconstruction, dynamicness estimation (black=static and white=dynamic) and background-only reconstruction.

Background NV FVCT FTCV FVCT FTCV

Fig. 6: Novel view and time syntheses. NV: novel camera angle. FVCT: fix-view-change-time (time advances from top to bottom) and FCVT: fix-time-change-view (camera advances from top to bottom).

VI. Conclusion

We presented an approach for novel view and time synthesis of highly dynamic scenes from multi-view videos recorded by a driving vehicle. We unified previous methods, and proposed a flow-based approach to successfully disentangle static and dynamic regions. In the future, we hope to extend the current framework to longer sequences and larger camera motions such as car turns, and to explore more efficient training strategies that focus more on the quality of the dynamic region.

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 *The European Conference on Computer Vision (ECCV)*, 2020.

[2] G. Gafni, J. Thies, M. Zollhöfer, and M. Nießner, "Dynamic neural radiance fields for monocular 4D facial avatar reconstruction," *https://arxiv.org/abs/2012.03065*, 2020.

[3] K. Park, U. Sinha, J. T. Barron, S. Bouaziz, D. Goldman, S. Seitz, and R. Martin-Brualla, "Deformable neural radiance fields," *https://arxiv.org/abs/2011.12948*, 2020.

[4] A. Pumarola, E. Corona, G. Pons-Moll, and F. Moreno-Noguer, "D-NeRF: Neural radiance fields for dynamic scenes," *https://arxiv.org/abs/2011.13961*, 2020.

[5] R. Martin-Brualla, N. Radwan, M. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth, "NeRF in the wild: Neural radiance fields for unconstrained photo collections," *https://arxiv.org/abs/2008.02268*, 2020.

[6] Z. Li, S. Niklaus, N. Snavely, and O. Wang, "Neural scene flow fields for space-time view synthesis of dynamic scenes," *https://arxiv.org/abs/2011.13084*, 2020.

[7] T. Li, M. Slavcheva, M. Zollhoefer, S. Green, C. Lassner, C. Kim, T. Schmidt, S. Lovegrove, M. Goesele, and Z. Lv, "Neural 3d video synthesis," 2021.

[8] E. Tretschk, A. Tewari, V. Golyanik, M. Zollhöfer, C. Lassner, and C. Theobalt, "Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a deforming scene from monocular video," *https://arxiv.org/abs/2012.12247*, 2020.

[9] J. Ost, F. Mannan, N. Thuerey, J. Knodt, and F. Heide, "Neural scene graphs for dynamic scenes," *https://arxiv.org/abs/2011.10379*, 2020.

[10] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research (IJRR)*, 2013.

[11] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *CVPR*, 2020.

[12] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 2446–2454.

[13] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multiview stereopsis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 8, pp. 1362–1376, Aug. 2010. [Online]. Available: https://doi.org/10.1109/tpami.2009.161

[14] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[15] T. Zhou, R. Tucker, J. Flynn, G. Fyffe, and N. Snavely, "Stereo magnification: learning view synthesis using multiplane images," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 65:1–65:12, 2018. [Online]. Available: https://doi.org/10.1145/3197517.3201323

[16] R. Tucker and N. Snavely, "Single-view view synthesis with multiplane images," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[17] P. P. Srinivasan, R. Tucker, J. T. Barron, R. Ramamoorthi, R. Ng, and N. Snavely, "Pushing the boundaries of view extrapolation with multiplane images," *CVPR*, 2019.

[18] 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," *ACM Transactions on Graphics (TOG)*, 2019.

[19] V. Sitzmann, J. Thies, F. Heide, M. Nießner, G. Wetzstein, and M. Zollhöfer, "Deepvoxels: Learning persistent 3d feature embeddings," in *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2019.

[20] V. Sitzmann, M. Zollhöfer, and G. Wetzstein, "Scene representation networks: Continuous 3d-structure-aware neural scene representations," in *Advances in Neural Information Processing Systems*, 2019.

[21] S. Lombardi, T. Simon, J. Saragih, G. Schwartz, A. Lehrmann, and Y. Sheikh, "Neural volumes: Learning dynamic renderable volumes from images," *ACM Trans. Graph.*, vol. 38, no. 4, pp. 65:1–65:14, Jul. 2019.

[22] G. Riegler and V. Koltun, "Free view synthesis," 2020.

[23] ——, "Stable view synthesis," 2020.

[24] 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," *arXiv preprint arXiv:2102.13090*, 2021.

[25] A. Tao, K. Sapra, and B. Catanzaro, "Hierarchical multi-scale attention for semantic segmentation," 2020. [Online]. Available: https://github.com/NVIDIA/semantic-segmentation

[26] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[27] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm, "Pixelwise view selection for unstructured multi-view stereo," in *European Conference on Computer Vision (ECCV)*, 2016.

[28] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," 2020.

[29] Q.-A. Chen, "nerf-pl: a pytorch-lightning implementation of nerf," 2020. [Online]. Available: https://github.com/kwea123/nerf_pl/

[30] E. Riba, D. Mishkin, D. Ponsa, E. Rublee, and G. Bradski, "Kornia: an open source differentiable computer vision library for pytorch," in *Winter Conference on Applications of Computer Vision*, 2020. [Online]. Available: https://arxiv.org/pdf/1910.02190.pdf

[31] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *CVPR*, 2018.