EUROGRAPHICS 2021 / N. Mitra and I. Viola
(Guest Editors)

Volume 40 (2021), Number 2

Few-camera Dynamic Scene Variational Novel-view Synthesis

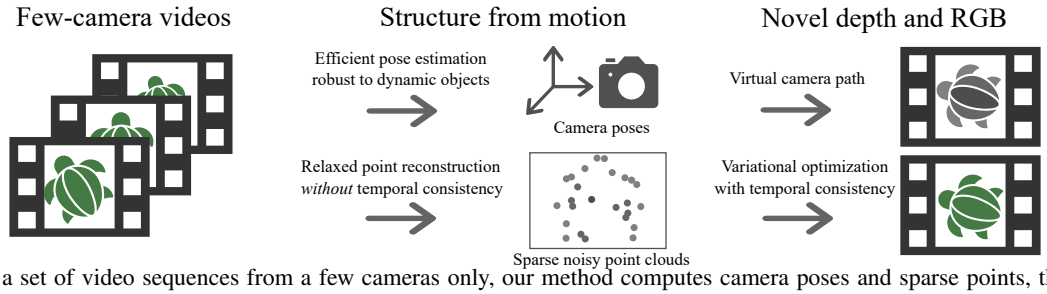

Figure 1: Given a set of video sequences from a few cameras only, our method computes camera poses and sparse points, then optimizes those points into a novel video sequence following a user-defined camera path. Our space-time SfM relaxes temporal consistency for points on dynamic objects and instead robustly adds consistency during the novel view synthesis via our variational formulation.

Abstract

Structure from motion (SfM) enables us to reconstruct a scene via casual capture from cameras at different viewpoints, and novel view synthesis (NVS) allows us to render a captured scene from a new viewpoint. However, in the casual setting, dynamic scenes are hard: SfM can produce noisy and spatio-temporally sparse reconstructed point clouds, and so the resulting NVS often produces spatio-temporally inconsistent effects. We consider SfM and NVS parts together *to ease the challenge. First, for SfM, we recover stable camera poses, then we relax the requirement for temporally-consistent points across the scene and reconstruct only a sparse point cloud per timestep that is noisy in space-time. Second, for NVS, we present a variational diffusion formulation on depths and colors that lets us robustly cope with the noise by enforcing spatio-temporal consistency via per-pixel reprojection weights derived from the input views. Together, these parts let us generate novel views for dynamic scenes without requiring challenging spatio-temporally consistent reconstructions, which expands the kinds of scenes to which these techniques can be applied. We demonstrate our algorithm on real-world dynamic scenes against recent baseline approaches.*

CCS Concepts

• ***Computing methodologies*** → ***Image-based rendering; Image and video acquisition;***

1. Introduction

Novel-view synthesis (NVS) creates a new view of a scene by combining existing images captured from different viewpoints. Many works have made progress in NVS over the past two decades to tackle its two core problems: 1) how to build a proxy scene geometry to aid in rendering, e.g. structure from motion (SfM), and 2) how to synthesize an image via the reprojected proxy given the existing captured imagery. In this paper, we consider dynamic scenes captured by a low number of cameras as might be set up by a small crowd of people capturing an event. In the literature, this is a relatively rare and challenging setting, because both the cameras and the scene objects can move simultaneously and because the smaller number of cameras makes robustness harder to obtain. It makes camera pose estimation and depth estimation in SfM hard, which can lead to ghosting, bleeding, and flickering artifacts across views and time during NVS in both moving objects and the background. Thus, it is crucial to enforce spatio-temporal consistency in both the SfM and the NVS to reduce these artifacts.

We propose to address these challenges by relaxing the difficult problem of reconstructing dynamic objects in time via SfM, and using a complementary NVS approach to enforce temporal consistency instead. To do so, we first recover camera poses for all views without any explicit dynamic object segmentation. Then, we recover scene points on both static and dynamic objects by *relaxing* temporal consistency and performing per-timestamp SfM across views only. This is easier to solve, but leads to significantly noisy reconstructions temporally. Our method then simultaneously computes a depth map and the RGB image in the virtual camera's view using a diffusion-based formulation. This lets us enforce robust temporal consistency in the output depth to overcome the initial noisy reconstructions.

2. Related Work

Static scene IBR. IBR has initially attempted to render static scenes either from set of images or videos [ZC04]. This can be achieved either via warping input views using optical flow [CW93], using coarse geometric proxies [GGSC96] or via deep learning approaches [FBD*19]. In complex environments, IBR techniques often need some 3D proxy reconstruction, either as planar coarse geometric proxies [GGSC96, BBM*01, SGHS98, DTM96], or 3D meshes from multi-view stereo reconstructions [SSS06, HRDB16, CDSHD13]. Recently, Riegler and Koltun [RK20] proposed to synthesize a new view via neural textures atop a Delaunay reconstruction of sparse points obtained from a video of a static scene. To achieve smoother interpolations, Kopf et al. [KLS*13] render static scenes in the gradient domain. Kopf et al. [KCS14] use a 3D reconstruction with rendered appearance obtained via a spatio-temporal Poisson equation to render hyperlapse videos, or to render new views [HK18].

With the recent success of deep learning, this framework has been largely employed for static IBR. This includes plane sweep volumes [FNPS16] to interpolate between multiple images of static scenes, multi-plane images to interpolate between views [ZTF*18, MSOC*19, FBD*19], appearance flows to generate novel views from a single image of static isolated objects [ZTS*16], and light-field view interpolation [KWR16]. [HPP*18] use a geometric proxy and learn blending weights between view reprojections using a CNN. To improve the quality around depth discontinuities, Choi et al. [CGT*19] use a 3D uncertainty volume as a proxy and neural network-based patch refinement. Deep Learning methods can work from a single image [SCH19]. More recently, NeRF [MST*20] learns a volumetric function for each scene that is then queried for synthesizing new views.

Dynamic Scenes Video-based rendering. Fewer works have tackled IBR in dynamic scenes [dAPG18]. Many dynamic scene methods focus on sports [YS04, GAD*13]. Many methods use controlled capture settings [ZKU*04, PTS*19], and camera arrays with up to 100 tightly-packed cameras [WJV*05, TTFY10, CCS*15], and high number of light sources [GLD*19]. Penner and Zhang [PZ17] use a soft volumetric representation for short baseline IBR to enforce smooth reconstructions, which works best with camera arrays.

Casually captured videos have also been considered for video-based rendering. Ballan et al [BBPP10] allow for quick transitions between video sequences. [MKGH16, MVK*19] reconstruct isolated moving objects after segmenting them out from the initial video. Very recently, Luo et al. [LHS*20] introduced a consistency term by training a neural network to improve the estimated depth per point in a single video with no or moderate dynamic motion. Finally, concurrent to our work, Bansal et al. [BVS*20] use foreground and background extraction together with a self-supervised CNN based composition operator and Yoon et al. [YKG*20] use deep learning to extrapolate new views from a single monocular video camera.

3. Overview

Our algorithm takes as input a set of casually-captured synchronized videos. We also provide the focal lengths for a pair of cameras, while the remaining focal lengths are estimated automatically by our algorithm. Our method proceeds in two steps (Figure **??**):

1. **Camera pose estimation and 3D scene points.** We perform a three-step structure from motion reconstruction to provide both the set of camera poses and a set of sparse 3D points for each time step (Section 4).
2. **Novel depth and novel view rendering.** We densify the sparse points into a depth map and render a new virtual camera frames by optimizing a coarse-to-fine variational formulation while enforcing spatio-temporal consistency (Section 5).

4. Camera pose estimation and 3D scene points.

Let us consider a set of S synchronized video views of a dynamic scene, each composed of T frames. We call $I = \{I_{s,t} | s = 1,...,S; t = 1,...,T\}$ the set of all frames indexed by s (camera index) and t (time-stamp). At each frame, via SfM, we will recover the camera parameters $C_{s,t}$ consisting of the intrinsic matrix and extrinsic rotation and translation matrices, and a set of sparse 3D points for all timesteps. First, we will efficiently recover a set of camera poses for all frames. In contrast to [BBPP10,MKGH16,MVK*19], we compute these without an explicit dynamic object segmentation step. Then, we recover 3D points by solving a relaxed per-timestep SfM. We solve each SfM problem with an *a contrario* algorithm [MMM12].

Efficient camera poses. One approach for accurate SfM could be to solve a problem across all frames simultaneously, but this can be expensive and memory prohibitive. A second approach might consider solving only between consecutive time steps, but this is known to produce camera position drift [CVV04]. Instead, we take a coarse-to-fine approach.

We begin by computing SfM across every $\kappa = 20^{\text{th}}$ keyframes of each video. We detect and match SIFT keypoints within this subset and then simultaneously solve for all camera poses and 3D points. Then, we refine our estimate with a second SfM that only matches keypoints between successive frames of the same camera view, with previously-estimated camera poses held fixed. This considers every frame of every video, but we only match $I_{s,t}$ to $I_{s,t+1}$, and not to $I_{s+1,t}$ or $I_{s+1,t+1}$. To recover smooth camera paths per view, we add two additional constraints to the bundle adjustment:

$$w(t-t')\|C_{s,t} - C_{s,t'}\|^2 = 0, \quad t-3 \le t' \le t+3 \tag{1}$$

and

$$w(t-t')\|A_{s,t} - A_{s,t'}\|^2 = 0, \quad t-3 \le t' \le t+3, \tag{2}$$

where $w(t-t')$ is a Gaussian weight function, and $A_{s,t}$ is the angle-axis representation of the rotation matrix $R_{s,t}$. This second SfM reduces computation time over all-pairs matching while still reducing drift by constraining the frame-to-frame pose estimates by the keyframe pose estimates. For hyperparameters, smaller κ will increase processing time, while larger κ may make it more difficult to match fast camera motion. We found $\kappa = 20$ to be a good compromise in our test sequences.

3D scene points. To recover 3D points across the scene, we solve one final SfM problem that is independent per time step. We consider all video frames together, we set as fixed all existing recovered camera poses, and then we restrict 2D keypoint matching to frames with the same timestamp. This results in a set of sparse 3D points per time step. Our relaxed point reconstruction is the key to handling dynamic scenes: as 2D keypoints are not matched along the time direction, moving objects are correctly recovered in space even if their motion makes matching over time difficult. However, this knowingly produces temporal inconsistencies; we will recover from these errors in our rendering step, where it is easier to enforce temporal consistency.

Post processing. By reprojection of the points onto the camera planes, we attach a color value to each reconstructed point. The color will serve as a coarse initialization to the novel view rendering. Second, to improve robustness, we increase the density of our point matches using PatchMatch [BSFG09]. At this stage, our output point clouds are still very sparse, and a simple re-projection would leave the majority of our frames uncovered. Hence we diffuse these points in depth and RGB while enforcing spatio-temporal constraints.

5. Novel Depth and Novel View Rendering

Our SfM provides a set of camera poses and a colored 3D point cloud per time step. Next, we will render a novel RGB camera path.

Notation . We will often reproject the content of one frame into the domain of another frame: this reprojection is computed using the extrinsic and intrinsic camera parameters estimated by SfM as well as the depth values d associated to each pixel. We will denote $proj^{s,t}_{D_{t'}}(x)$ as the pixel location on frame $I_{s,t}$ projected from the pixel location x of frame $I_{t'}$ using the depth map $D_{t'}$. We will denote a warping from an image plane s,t to image plane s',t' (using depth D_t) with $proj,s',t'$ in superscript. For example $I^{proj,s,t-1}_{s,t}$ denotes the projection of the content of frame s,t to the domain of frame $s,t-1$ based on the camera parameters and depth map D_t. If the superscript is not followed by a frame and time index (i.e., $D^{proj}_{s,t}$), then the depth is projected into the virtual view currently being synthesized. Finally, we denote by $\hat{D}_t$ the sparse depth map obtained by projecting the sparse point cloud into frame t.

Algorithm progression. To compute the projection of a frame $I_{s,t}$ into virtual view I_t, we need both the estimated camera poses and the dense depth map D_t. We proceed in a two-step manner: we first estimate the depth map D_t, and then we estimate the color image I_t. For efficiency, we also proceed in a multiscale fashion: we solve for depth and RGB at a coarse resolution, and then use these to initialize a finer resolution (Fig. 1)—our lowest level is 1/64 of the original frame size. Finally, we also proceed in a streaming manner: we reproject the previous frame's depth and RGB (denoted as D^{proj}_{t-1} and I^{proj}_{t-1}) into the current virtual camera pose.

Depth diffusion. We begin by projecting the sparse point cloud into the novel view, creating a sparse depth map $\hat{D}^{proj}_t$ as an initialization. We densify this by minimizing the following energy:

$$\begin{aligned} w^{D_t}(x,t)\|\nabla D_t\|^2 + \lambda_{PC}\int_{x\in\Omega} w^{\hat{D}_t}(x,t)(D_t(x)-\hat{D}_t(x))^2\mathrm{d}x \\ +\lambda_T\int_{x\in\Omega} w_T(x,t)(D_t(x)-D^{proj}_{t-1}(x))^2\mathrm{d}x \end{aligned} \tag{3}$$

where w^D is a weight that considers the inter-view consistency of the pixels. It combines two parts: 1) A frame weight $w_F(s)$ that decreases when frame s is far from the novel view point, and 2) A projection weight w_P that gives more weights to pixels where all input frames agree on the color; $w^{D_t}_P(x,s,t)$ acts as a soft occlusion map for the input images $I_{s,t}$. More precisely:

$$w^D(x,t) = \sum_{s=1\cdots S} w_F(s) w^D_P(x,s,t) \tag{4}$$

with w_F and w^D_P defined by:

$$w_F(s) = \frac{1}{C}(\mathrm{trace}(R_t R^T_{s,t})-1)\frac{\min_{s'=1\cdots S}\|C_t - C_{s',t}\|^2}{\|C_t - C_{s,t}\|^2} \tag{5}$$

C is a normalization factor ensuring that $\sum_{s=1\ldots S} w_F(s) = 1$; and

$$w^D_P(x,s,t) = \frac{1}{S-1}\sum_{\substack{s'=1\ldots S\\ s'\neq s}} \exp - \frac{\|I_{s,t}(proj^{s,t}_D(x)) - I_{s',t}(proj^{s',t}_D(x))\|^2}{2\sigma^2_D}. \tag{6}$$

Temporal consistency is enforced in the third term of Equation 3. It constrains the depth to remain similar to the warped depth at time $t-1$ for frames that are consistent. This consistency is given by a weight w_T that combines the frame weight $w_F(s)$ defined above and a factor that decreases when the warping of I_{t-1} is too different from that of $I_{s,t}$.

$$w_T(x,t) = \sum_{s=1\ldots S} w_F(s)\exp - \frac{\|I^{proj}_{t-1}(x) - I_{s,t}(proj^{s,t}_{D_{t-1}}(x))\|^2}{2\sigma^2_D} \tag{7}$$

There are 3 parameters in this diffusion process: σ_D controls the soft occlusion tolerance, and we set $\sigma_D = 0.075$ in all our experiments; the sparse point cloud attachment weight λ_{PC}, which we set to $\lambda_{PC} = 0.25$–2; and the temporal consistency term set in the range $\lambda_T = 0.25$–1.

Color diffusion. Given the depth map, we initialize the RGB image to a projection of the color in the input point cloud. Then, we densify this by minimizing the following diffusion energy:

$$\begin{aligned} &\|\nabla I_t\|^2 + \lambda_T\int_{x\in\Omega} w_T(x,t)(I_t(x) - I^{proj}_{t-1}(x))^2 dx \\ &+\sum_{s=1}^{S}\int_{x\in\Omega}\lambda_P w_F(s) w^{D_t}_P(x,s,t)(I_t(x) - I_{s,t}(proj^{s,t}_{D_t}(x)))^2 dx \\ &+\sum_{s=1}^{S}\int_{x\in\Omega}\lambda_G w_F(s)\|\nabla I_t(x) - \nabla I_{s,t}(proj^{s,t}_{D_t}(x))\|^2) dx \end{aligned} \tag{8}$$

The last two integrals in Equation 8 respectively constrain the

Ours DB [HPP*18] EVS [CGT*19] LLFF [MSOC*19] Monocam [YKG*20]

Figure 2: Comparison with other state-of-the-art IBR methods on the Jumping sequence.

Figure 3: Color and depth results for Cat&dog and Elephant-wiggle scenes.

RGB intensities and gradients of I_t to be close to the intensities and gradients of $I_{s,t}$. Over the intensity term, adding the gradient term helps to recover a sharp image. The color diffusion relies on three weights: σ_D and λ_T serve the same function as in the depth map diffusion, and λ_G provides control over the gradient equality constraint. We set λ_G in the range 0.5–2.5.

6. Experiments and Results

6.1. Dataset Sequences

We exploit existing datasets used in the context of novel view synthesis, all are taken from [YKG*20], as well as sequences (100 frames, 1920×1080) we captured:

- Jumping: A group of four people jump (12 cameras).
- Skating: A person rides a skateboard (12 cameras).
- Playground: A person flies a dinosaur balloon (12 cameras).
- Umbrella: A person opens and rotates an umbrella (12 cameras).
- Cat and dog: Two pet animatronics (5 cameras).
- Elephant wiggle: A puppet hanging by a wire (5 cameras).
- Drone: A drone hanging by a wire (5 cameras).

Figure 3 shows rendered frames from a new view and corresponding depth maps for the cat and dog sequence and the elephant wiggle sequence. While some artifacts remain in the depth video, the generation of the final novel view RGB rendered sequence is robust to these and has fewer artifacts.

In comparison with state-of-the-art IBR methods (Fig. 2), our method produces fewer artifacts than EVS [CGT*19], and produces sharper and higher-contrast images than DeepBlending (DB) [HPP*18]. Compared to LLFF [MSOC*19], our method exhibits fewer ghosting artifacts, and Monocam [YKG*20] produces temporally inconsistent background. Please see the accompanying video for full results and additional experiments.

6.2. Computational Resources

We implemented our system in C++ on a Intel(R) Xeon(R) CPU ES-2630 v3 @2.4GHz computer. We used the OpenMVG and OpenMVS libraries for the SfM. Our code uses OpenMP and run on 32 cores; the rendering algorithm loads up to 2GBs of data per frame. It took 2.6 hours to process the elephant-wiggle sequence (5 cameras, 100 frames per camera) split into: camera and sparse depth estimation (2 hours), and rendering (20s per frame, 33 minutes for the whole video).

7. Limitations

Our method has several limitations. First, our method requires that the video sequences should have enough texture on the objects and in the background such that enough SIFT keypoints can be detected and matched. Another limitation lies in the amount of motion in the frame: conceptually, if SIFT keypoints are only detected on moving objects, then camera pose estimation will fail. In practice, we did not find this to be a problem. Furthermore, if the baseline is too wide, then not enough points will be obtained on moving objects and the depth propagation will fail. Finally, our optimization has parameters that can be tuned for each sequences; we provide reasonable initial values (Sec. 5), but tweaking can improve quality.

Finally, our current implementation is unoptimized C++ running on a CPU. If we consider SfM as an offline task to be performed once per scene, then the view rendering part currently takes 20 seconds per frame. Given the fixed grid, GPU-based diffusion optimizers are possible, which would produce a much more application-friendly render time.

8. Conclusion

We introduce a few-camera view synthesis method which can handle dynamic scenes. It is based around the key insight that reconstructing temporally-consistent 3D points on dynamic objects is hard, yet a SfM reconstruction method need not be temporally consistent if temporal consistency can be enforced in the rendering algorithm. While our setting has some restrictions, we show competitive results against existing baselines for video-based rendering without using any learning-based approaches. As future work, we aim to relax constraints in camera motions with asynchronous videos and partial temporal overlap.

References

[BBM*01] BUEHLER C., BOSSE M., MCMILLAN L., GORTLER S., COHEN M.: Unstructured lumigraph rendering. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (2001), ACM, pp. 425–432. 2

[BBPP10] BALLAN L., BROSTOW G. J., PUWEIN J., POLLEFEYS M.: Unstructured video-based rendering: Interactive exploration of casually captured videos. *ACM Transactions on Graphics (TOG) 29*, 4 (2010), 87. 2

[BSFG09] BARNES C., SHECHTMAN E., FINKELSTEIN A., GOLDMAN D. B.: Patchmatch: A randomized correspondence algorithm for structural image editing. In *ACM SIGGRAPH 2009 Papers* (New York, NY, USA, 2009), SIGGRAPH '09, Association for Computing Machinery. 3

[BVS*20] BANSAL A., VO M., SHEIKH Y., RAMANAN D., NARASIMHAN S.: 4d visualization of dynamic events from unconstrained multi-view videos. 2

[CCS*15] COLLET A., CHUANG M., SWEENEY P., GILLETT D., EVSEEV D., CALABRESE D., HOPPE H., KIRK A., SULLIVAN S.: High-quality streamable free-viewpoint video. *ACM Trans. Graph. 34*, 4 (2015). 2

[CDSHD13] CHAURASIA G., DUCHENE S., SORKINE-HORNUNG O., DRETTAKIS G.: Depth synthesis and local warps for plausible image-based navigation. *ACM Transactions on Graphics (TOG) 32*, 3 (2013), 30. 2

[CGT*19] CHOI I., GALLO O., TROCCOLI A., KIM M. H., KAUTZ J.: Extreme view synthesis. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)* (2019), pp. 7780–7789. 2, 4

[CVV04] CORNELIS K., VERBIEST F., VAN GOOL L.: Drift detection and removal for sequential structure from motion algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence 26*, 10 (Oct 2004), 1249–1259. 2

[CW93] CHEN S. E., WILLIAMS L.: View interpolation for image synthesis. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques* (1993), ACM, pp. 279–288. 2

[dAPG18] DOS ANJOS R. K., PEREIRA J., GASPAR J.: A navigation paradigm driven classification for video-based rendering techniques. *Computers & Graphics 77* (2018), 205–216. 2

[DTM96] DEBEVEC P. E., TAYLOR C. J., MALIK J.: 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* (1996), pp. 11–20. 2

[FBD*19] FLYNN J., BROXTON M., DEBEVEC P., DUVALL M., FYFFE G., OVERBECK R., SNAVELY N., TUCKER R.: Deepview: View synthesis with learned gradient descent. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2019), pp. 2367–2376. 2

[FNPS16] FLYNN J., NEULANDER I., PHILBIN J., SNAVELY N.: Deepstereo: Learning to predict new views from the world's imagery. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 5515–5524. 2

[GAD*13] GOORTS P., ANCUTI C., DUMONT M., ROGMANS S., BEKAERT P.: Real-time video-based view interpolation of soccer events using depth-selective plane sweeping. 2

[GGSC96] GORTLER S. J., GRZESZCZUK R., SZELISKI R., COHEN M. F.: The lumigraph. In *Siggraph* (1996), vol. 96, pp. 43–54. 2

[GLD*19] GUO K., LINCOLN P., DAVIDSON P., BUSCH J., YU X., WHALEN M., HARVEY G., ORTS-ESCOLANO S., PANDEY R., DOURGARIAN J., TANG D., TKACH A., KOWDLE A., COOPER E., DOU M., FANELLO S., FYFFE G., RHEMANN C., TAYLOR J., DEBEVEC P., IZADI S.: The relightables: Volumetric performance capture of humans with realistic relighting. *ACM Trans. Graph. 38*, 6 (2019). 2

[HK18] HOLYNSKI A., KOPF J.: Fast depth densification for occlusion-aware augmented reality. In *SIGGRAPH Asia 2018 Technical Papers* (2018), ACM, p. 194. 2

[HPP*18] HEDMAN P., PHILIP J., PRICE T., FRAHM J.-M., DRETTAKIS G., BROSTOW G.: Deep blending for free-viewpoint image-based rendering. *ACM Trans. Graph. 37*, 6 (Dec. 2018). 2, 4

[HRDB16] HEDMAN P., RITSCHEL T., DRETTAKIS G., BROSTOW G.: Scalable inside-out image-based rendering. *ACM Transactions on Graphics (TOG) 35*, 6 (2016), 231. 2

[KCS14] KOPF J., COHEN M. F., SZELISKI R.: First-person hyper-lapse videos. *ACM Transactions on Graphics (TOG) 33*, 4 (2014), 78. 2

[KLS*13] KOPF J., LANGGUTH F., SCHARSTEIN D., SZELISKI R., GOESELE M.: Image-based rendering in the gradient domain. *ACM Transactions on Graphics (TOG) 32*, 6 (2013), 199. 2

[KWR16] KALANTARI N. K., WANG T.-C., RAMAMOORTHI R.: Learning-based view synthesis for light field cameras. *ACM Transactions on Graphics (TOG) 35*, 6 (2016), 193. 2

[LHS*20] LUO X., HUANG J., SZELISKI R., MATZEN K., KOPF J.: Consistent video depth estimation. 2

[MKGH16] MUSTAFA A., KIM H., GUILLEMAUT J., HILTON A.: Temporally coherent 4d reconstruction of complex dynamic scenes. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2016), pp. 4660–4669. 2

[MMM12] MOULON P., MONASSE P., MARLET R.: Adaptive structure from motion with a contrario model estimation. In *Proceedings of the Asian Computer Vision Conference (ACCV 2012)* (2012), Springer Berlin Heidelberg, pp. 257–270. 2

[MSOC*19] MILDENHALL B., SRINIVASAN P. P., ORTIZ-CAYON R., KALANTARI N. K., RAMAMOORTHI R., NG R., KAR A.: Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Trans. Graph. 38*, 4 (July 2019). 2, 4

[MST*20] MILDENHALL B., SRINIVASAN P. P., TANCIK M., BARRON J. T., RAMAMOORTHI R., NG R.: Nerf: Representing scenes as neural radiance fields for view synthesis, 2020. 2

[MVK*19] MUSTAFA A., VOLINO M., KIM H., GUILLEMAUT J., HILTON A.: Temporally coherent general dynamic scene reconstruction. *CoRR abs/1907.08195* (2019). 2

[PTS*19] POZO A. P., TOKSVIG M., SCHRAGER T. F., HSU J., MATHUR U., SORKINE-HORNUNG A., SZELISKI R., CABRAL B.: An integrated 6dof video camera and system design. *ACM Trans. Graph. 38*, 6 (2019). 2

[PZ17] PENNER E., ZHANG L.: Soft 3d reconstruction for view synthesis. *ACM Transactions on Graphics (TOG) 36*, 6 (2017), 235. 2

[RK20] RIEGLER G., KOLTUN V.: Free view synthesis. In *European Conference on Computer Vision* (2020). 2

[SCH19] SONG J., CHEN X., HILLIGES O.: Monocular neural image based rendering with continuous view control. In *ICCV 2019* (2019), pp. 4089–4099. 2

[SGHS98] SHADE J., GORTLER S., HE L.-W., SZELISKI R.: Layered depth images. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques* (1998), pp. 231–242. 2

[SSS06] SNAVELY N., SEITZ S. M., SZELISKI R.: Photo tourism: exploring photo collections in 3d. In *ACM transactions on graphics (TOG)* (2006), vol. 25, ACM, pp. 835–846. 2

[TTFY10] TANIMOTO M., TEHRANI M. P., FUJII T., YENDO T.: Free-viewpoint tv. *IEEE Signal Processing Magazine 28*, 1 (2010), 67–76. 2

[WJV*05] WILBURN B., JOSHI N., VAISH V., TALVALA E.-V., ANTUNEZ E., BARTH A., ADAMS A., HOROWITZ M., LEVOY M.: High performance imaging using large camera arrays. *ACM Trans. Graph. 24*, 3 (2005). 2

[YKG*20] YOON J. S., KIM K., GALLO O., PARK H. S., KAUTZ J.: Novel view synthesis of dynamic scenes with globally coherent depths from a monocular camera. 2, 4

[YS04] YAGUCHI S., SAITO H.: Arbitrary viewpoint video synthesis from multiple uncalibrated cameras. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 34*, 1 (2004), 430–439. 2

[ZC04] ZHANG C., CHEN T.: A survey on image-based rendering—representation, sampling and compression. *Signal Processing: Image Communication 19*, 1 (2004), 1–28. 2

[ZKU*04] ZITNICK C. L., KANG S. B., UYTTENDAELE M., WINDER S., SZELISKI R.: High-quality video view interpolation using a layered representation. *ACM Trans. Graph. 23*, 3 (Aug. 2004), 600–608. 2

[ZTF*18] ZHOU T., TUCKER R., FLYNN J., FYFFE G., SNAVELY N.: Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817* (2018). 2

[ZTS*16] ZHOU T., TULSIANI S., SUN W., MALIK J., EFROS A. A.: View synthesis by appearance flow. In *European conference on computer vision* (2016), Springer, pp. 286–301. 2