Hybrid Neural Fusion for Full-frame Video Stabilization

Yu-Lun Liu[1]   Wei-Sheng Lai[2]   Ming-Hsuan Yang[2,4]   Yung-Yu Chuang[1]   Jia-Bin Huang[3]
[1]National Taiwan University   [2]Google   [3]Virginia Tech   [4]University of California at Merced
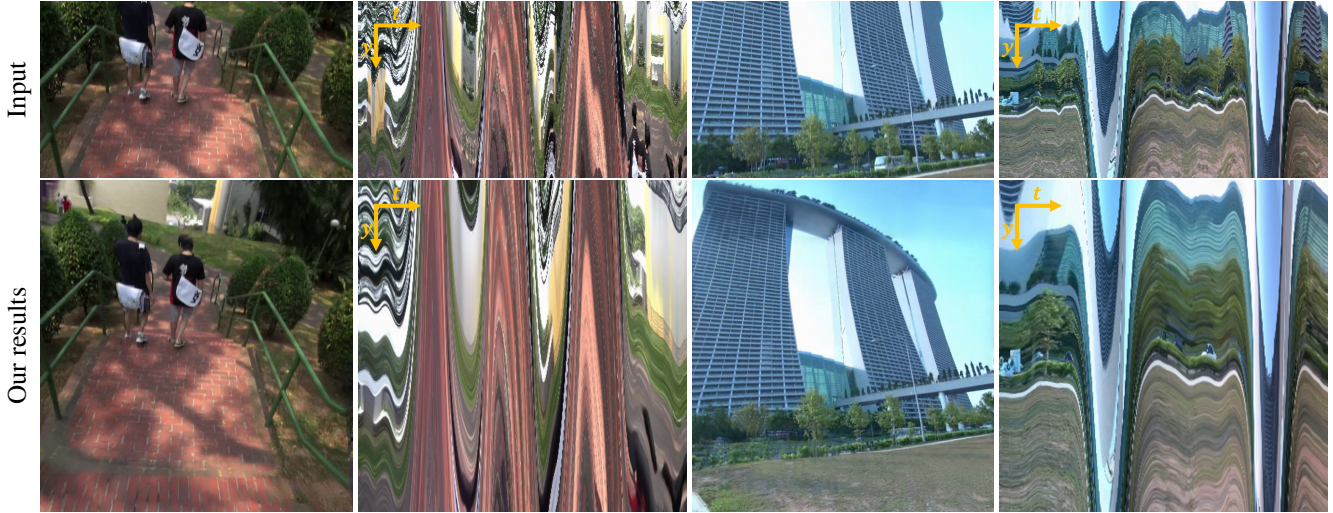https://alex04072000.github.io/FuSta/

Figure 1: **Full-frame video stabilization of challenging videos.** Our method takes a shaky input video (*top*) and produces a stabilized and distortion-free video (*bottom*), as indicated by less fluctuation in the $y-t$ epipolar plane image. Furthermore, by robustly fusing multiple neighboring frames, our results do not suffer from aggressive cropping of frame borders in the stabilized video and can even expand the field of view of the original video. Our approach significantly outperforms representative state-of-the-art video stabilization algorithms on these challenging scenarios (see Figure 2).

Abstract

Existing video stabilization methods often generate visible distortion or require aggressive cropping of frame boundaries, resulting in smaller field of views. In this work, we present a frame synthesis algorithm to achieve full-frame video stabilization. We first estimate dense warp fields from neighboring frames and then synthesize the stabilized frame by fusing the warped contents. Our core technical novelty lies in the learning-based hybrid-space fusion that alleviates artifacts caused by optical flow inaccuracy and fast-moving objects. We validate the effectiveness of our method on the NUS, selfie, and DeepStab video datasets. Extensive experiment results demonstrate the merits of our approach over prior video stabilization methods.

1. Introduction

Video stabilization has become increasingly important with the rapid growth of video content on the Internet platforms, such as YouTube, Vimeo, and Instagram. Casually captured cellphone videos without a professional video stabilizer are often shaky and unpleasant to watch. These videos pose significant challenges for video stabilization algorithms. For example, videos are often noisy due to small image sensors, particularly in low-light environments. Handheld captured videos may contain large camera shake/jitter, resulting in severe motion blur and wobble artifacts from a rolling shutter camera.

Existing video stabilization methods usually consist of three main components: 1) motion estimation, 2) motion smoothing and 3) stable frame generation. First, the motion estimation step involves estimating motion through 2D feature detection/tracking [24, 29, 14, 55], dense flow [61, 62], or recovering camera motion and scene structures [28, 64, 5, 46, 30]. Second, the motion smoothing step then removes the high-frequency jittering in the estimated motion and predicts the spatial transformations to stabilize each frame in the form of homography [34], mixture of homography [31, 16], or per-pixel warp fields [32, 61, 62]. Third,
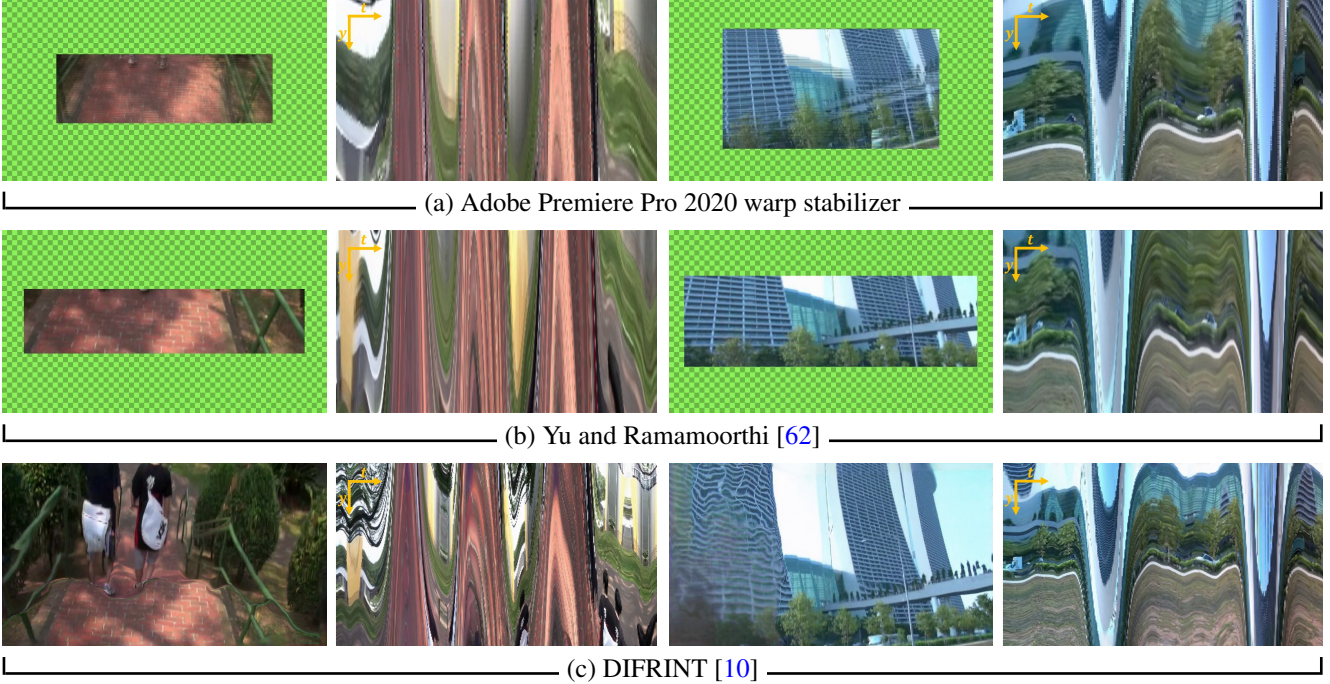
Figure 2: **Limitations of current state-of-the-art video stabilization techniques.** (a) Current commercial video stabilization software (Adobe Premiere Pro 2020) fails to generate smooth videos in challenging scenarios of rapid camera shakes. (b) Yu and Ramamoorthi's method [62] produces a temporally smooth video. However, the warped (stabilized) video contains many missing pixels at frame borders and inevitably requires applying aggressive cropping (green checkerboard areas) to generate a rectangular video. (c) The DIFRINT method [10] achieves full-frame video stabilization by iteratively applying frame interpolation to generate in-between, stabilized frames. However, interpolating between frames with large camera motion and moving occlusion is challenging. Their results are thus prone to severe artifacts.

the stable frame generation step uses the predicted spatial transform to synthesize the stabilized video. The stabilized frames, however, often contain large missing regions at frame borders, particularly when videos with large camera motion. This forces existing methods to apply aggressive cropping for maintaining a rectangular frame and therefore leads to a significantly *zoomed-in* video with resolution loss (Figure 2(a) and (b)).

Full-frame video stabilization methods aim to address the above-discussed limitation and produce stabilized video with the same field of view (FoV). One approach for full-frame video stabilization is to first compute the stabilized video (with missing pixels at the frame borders) and then apply flow-based video completion methods [34, 19, 12] to fill in missing contents. Such two-stage methods may suffer from the inaccuracies in flow estimation and inpainting (e.g., in poorly textured regions, fluid motion, and motion blur). A recent learning-based method, DIFRINT [10], instead uses iterative frame interpolation to stabilize the video while maintaining the original FoV. However, applying frame interpolation repeatedly leads to severe distortion and blur artifacts in challenging cases (Figure 2(c)).

In this paper, we present a new algorithm that takes a shaky video and the estimated smooth motion fields for stabilization as inputs and produces a full-frame stable video. The core idea of our method lies in fusing information from multiple neighboring frames in a robust manner. Instead of using color frames directly, we use a learned CNN representation to encode rich local appearance for each frame, fuse multiple aligned feature maps, and use a neural decoder network to render the final color frame. We first explore multiple design choices for fusing and blending multiple aligned frames (commonly used in image stitching and view synthesis applications). We then propose a hybrid fusion mechanism that leverages both feature-level and image-level fusion to alleviate the sensitivity to flow inaccuracy. We further improve the visual quality of the synthesized results by learning to predict spatially varying blending weights, removing blurry input frames for sharp video generation, and transferring high-frequency details residual to the re-rendered, stabilized frames. To minimize regions where contents are unknown for all neighboring frames, we propose a path adjustment method for balancing the goals of smoothing camera motion and maximizing frame cov-

erage. Our method generates stabilized video with significantly fewer artifacts and distortions while retaining (or even expanding) the original FoV (Figure 1). We evaluate the proposed algorithm with state-of-the-art methods and commercial video stabilization software (Adobe Premiere Pro 2020 warp stabilizer). Extensive experiments show that our method performs favorably against existing methods on three public benchmark datasets [31, 60, 54].

The main contributions of this work are:

- We apply a neural fusion technique in the context of full-frame video stabilization to alleviate the issues of sensitivity to flow inaccuracy.
- We present a hybrid fusion method for fusing information from multiple stabilized frames at both feature- and image-level. We systematically validate various design choices through detailed ablation studies.
- We demonstrate favorable performance against representative video stabilization techniques on three public datasets. We will release the source code to facilitate future research.

2. Related work

Motion estimation and smoothing. Most video stabilization methods focus on estimating motion between frames and smoothing the motion. For motion estimation, existing methods often estimate the motion in the 2D image space using sparse feature detection/tracking and dense optical flow. These methods differ in motion modeling, e.g., eigen-trajectories [29], epipolar geometry [14], warping grids [31], or dense flow fields [62]. For motion smoothing, prior methods use low-pass filtering [29], L1 optimization [17], and spatio-temporal optimization [55].

In contrast to estimating 2D motion, several methods recover the camera motion and proxy scene geometry by leveraging Structure from Motion (SfM) algorithms. These methods stabilize frames using 3D reconstruction and projection along with image-based rendering [21] or content-preserving warps [5, 28, 14]. However, SfM algorithms are less effective in handling complex videos with severe motion blur and highly dynamic scenes [23]. Specialized hardware such as depth cameras [30] or light field cameras [46] may be required for reliable pose estimation.

Deep learning-based approaches have recently been proposed to directly predict warping fields [59, 54] or optical flows [61, 62] for video stabilization. In particular, methods with dense warp fields [32, 61, 62] offer greater flexibility for compensating motion jittering and implicitly handling rolling shutter effects than parametric warp fields [59, 54].

Our work builds upon existing 2D motion estimation/smoothing techniques for stabilization and focuses on synthesizing *full-frame video* outputs. Specifically, we adopt the state-of-the-art flow-based stabilization method [62] and use the estimated per-frame warped fields as inputs to our method. Note that our method is *agnostic* to the motion smoothing techniques. Other approaches such as parametric warps can also be applied.

Image fusion and composition. With the estimated and smoothed motion, the final step of video stabilization is to render the stabilized frames. Most existing methods synthesize frames by directly warping each input frame to the stabilized location using smoothed warping grids [31, 29] or flow fields predicted by CNNs [61, 62]. However, such approaches inevitably synthesize images with missing regions around frame boundaries. To maintain a rectangle shape, existing methods often crop off the blank areas and generate output videos with a lower resolution and a smaller FOV than the input video. To address this issue, full-frame video stabilization methods aim to stabilize videos without cropping. These methods use neighboring frames to fill in the blank and produce full-frame results by 2D motion inpainting [34, 13, 12]. In contrast to existing motion inpainting methods that first generate stabilized frames then filling in missing pixels, our method leverage neural rendering to encode and fuse warped appearance features and learn to decode the fused feature map to the final color frames.

Several recent methods can generate full-frame stabilized videos without explicit motion estimation. For example, the method in [54] train a CNN with collected unstable-stable pairs to directly synthesize stable frames. However, direct synthesis of output frames without spatial transformations remains challenging. Recently, the DIFRINT method [10] generates full-frame stable videos by iteratively applying frame interpolation. This method couples motion smoothing and frame rendering together. However, the repeated frame interpolation often introduces visible distortion and severe artifacts (see Figure 2(c)).

View synthesis. View synthesis algorithms aim to render photorealistic images of novel viewpoints from a single image [36, 57, 56, 42, 22] or multiple posed images [26, 15, 7, 37, 18, 38, 39]. These methods mainly differ in the ways to map and fuse information, e.g., view interpolation [8, 11, 41], 3D proxy geometry and mapping [6], multi-plane images [63, 47], and CNN [18, 38, 39]. Our fusion network resembles the encoder-decoder network used for free view synthesis[38].

A recent line of research focuses on rendering novel views for *dynamic* scenes from a single video [58, 53, 27] based on neural volume rendering [33, 35]. These methods can be used for full-frame video stabilization by rendering the dynamic video from a smooth camera trajectory. While promising results have been shown, these methods require per-video training and precise camera pose estimates. In contrast, our frame synthesis method can be applied to a wider variety of videos without re-training and when accurate camera poses are difficult to obtain.

Neural rendering. A direct blending of multiple images in the image space may lead to glitching artifacts (visible seams). Some recent methods train neural scene representations to synthesize novel views, such as NeRF [35], scene representation networks [45], neural voxel grid [44, 33], 3D Neural Point-Based Graphics [2], and neural textures [52]. However, these methods often require time-consuming per-scene training and do not handle dynamic scenes. In contrast, our method does not require per-video finetuning.

3. Full-frame video stabilization

Let I_{k^s} denote the *source* frame in the real (unstabilized) camera space and I_{k^t} the *target* frame in the virtual (stabilized) camera space at a timestamp k. Given an input video with T frames $\{I_{k^s}\}_{k=1}^{T}$, our goal is to generate a video $\{I_{k^t}\}_{k=1}^{T}$ that is visually stable and maintains the same FOV as the input video without any cropping. Existing video stabilization methods often apply a certain amount of cropping to exclude any missing pixels due to frame warping, as shown in Figure 2. In contrast, we utilize the information from neighboring frames to render stabilized frames with completed contents or even expanding the FOV of the input video.

Video stabilization methods typically consist of three stages: 1) motion estimation, 2) motion smoothing, and 3) frame warping/rendering. Our method focuses on the third stage for rendering high-quality frames without any cropping. Our proposed algorithm is thus agnostic to particular motion estimation/smooth techniques. We assume that the warping field from the real camera space to the virtual camera space is available for each frame (e.g., from [62]).

Given an input video, we first encode image features for each frame, warp the neighboring frames to the virtual camera space at the specific target timestamp, and then fuse the features to render a stabilized frame. We describe the technical detail of each step in the following sections.

3.1. Pre-processing

Motion estimation and smoothing. Several motion estimation and smoothing methods have been developed [29, 31, 62]. In this work, we use the state-of-the-art method [62] to obtain a backward dense warping field $F_{k^t \rightarrow k^s}$ for each frame, where k^s indicates the source input and k^t denotes the target stabilized output. Note that one can also generate the warping fields from other parametric motion models such as the homography or affine transformation. These warping fields can be directly used to warp the input video. However, the stabilized video often contains irregular boundaries and a large portion of missing pixels. Therefore, the output video requires aggressive cropping and thus loses some content.

Optical flow estimation. To recover the missing pixels

caused by warping, we need to project the corresponding pixels from nearby frames to the target stabilized frame. For each key frame I_{k^s} at time k, we compute the optical flows $\{F_{n^s \rightarrow k^s}\}_{n \in \Omega_k}$ from neighboring frames to the key frame using RAFT [51], where n indicates a neighboring frame and Ω_k denotes the set of neighboring frames for the key frame I_{k^s}.

3.2. Warping and fusion

Warping. We warp the neighboring frames $\{I_{n^s}\}_{n \in \Omega_k}$ to align with the target frame I_{k^t} in the virtual camera space. Since we already have the warping field from the target frame to the keyframe $F_{k^t \rightarrow k^s}$ (estimated from [62]) and the estimated optical flow from the keyframe to neighboring frames $\{F_{k^s \rightarrow n^s}\}_{n \in \Omega_k}$, we can then compute the warping field from the target frame to neighboring frames $\{F_{k^t \rightarrow n^s}\}_{n \in \Omega_k}$ by *chaining* the flow vectors. We can thus warp a neighboring frame I_{n^s} to align with the target frame I_{k^t} using backward warping. Some pixels in the target frame are not visible in the neighboring frames due to occlusion/dis-occlusion or out-of-boundary. Therefore, we compute visibility mask $\{M_{n^s}\}_{n \in \Omega_k}$ for each neighboring frame to indicate whether a pixel is valid (labeled as 1) in the source frame or not. We use Sundaram *et al.*'s method [49] to identify occluded pixels (labeled as 0).

Fusion space. With the aligned frames, we explore several fusion strategies. First, we can directly blend the warped color frames in the *image space* to produce the output stabilized frame, as shown in Figure 3(a). This image-space fusion approach is a commonly used technique in image stitching [1, 50], video extrapolation [25], novel view synthesis [18], and HDR reconstruction [20]. However, image-space fusion is prone to generating ghosting artifacts due to misalignment, or glitch artifacts due to inconsistent labeling between neighbor pixels. Alternatively, one can also fuse the aligned frames in the *feature space*, e.g., [9, 40], as shown in Figure 3(b). Fusing in the high-dimensional feature spaces allows the model to be more robust to flow inaccuracy. However, rendering the fused feature map using a neural image-translation decoder often leads to blurry outputs.

To combine the best worlds of both image-space and feature-space fusions, we propose a *hybrid-space* fusion mechanism for video stabilization (Figure 3(c)). Similar to the feature-space fusion, we first extract high-dimensional features from each neighboring frame and warp the features using flow fields. We then learn a CNN to predicting the blending weights that best fuse the features. We concatenate the *fused feature map* and the *warped feature for each neighboring frame* to form the input for our image generator. The image generator learns to predict a target frame and a confidence map for each neighboring frame. Finally, we adopt an image-space fusion to merge all the predicted

(a) Image (late fusion) (b) Feature (early fusion) (c) Hybrid fusion (proposed)
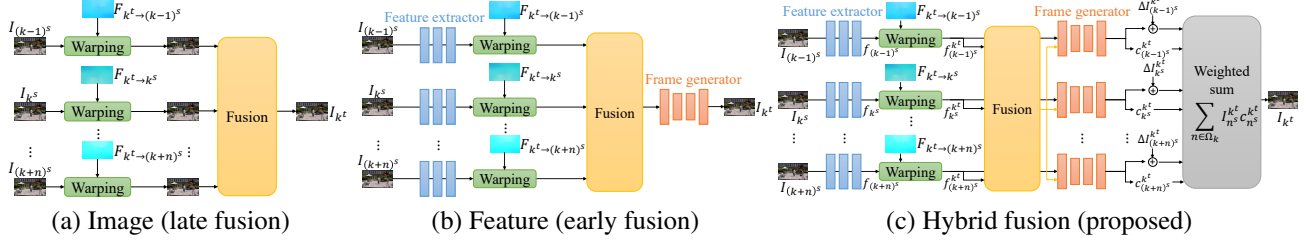
Figure 3: **Design choices for fusing multiple frames.** To synthesize a full-frame stabilized video, we need to *align* and *fuse* the contents from multiple neighboring frames in the input shaky video. (a) Conventional panorama image stitching (or in general image-based rendering) methods often fuse the warped (stabilized) images in the *image level*. Fusing in image-level works well when the alignment is accurate, but may generate blending artifacts (e.g., visible seams) when flow estimates are not reliable. (b) One can also encode the images as abstract CNN features, perform the fusion in the feature-space, and learn a decoder to convert the fused feature to output frames. Such approaches are more robust to flow inaccuracy but often produce overly blurred images. (c) Our proposed hybrid fusion combines the advantages of both strategies. We first extract abstract image features (Eq. (1)). We then fuse the warped features from multiple frames. For each source frame, we take the fused feature map together with the individual warped features and decode it to the output frames and the associated confidence maps. Finally, we produce the final output frame by using the weighted average of the generated images as in Eq. (4).

target frames according to the predicted weights to generate the final stabilized frame.

The core difference between our hybrid-space fusion and feature-space fusion lies in the input to the image generator. The image generator in Figure 3(b) takes *only* the fused feature as input to predict the output frame. The fused feature map already contains mixed information from multiple frames. The image generator may thus have difficulty in synthesizing sharp image contents. In contrast, our image generator in Figure 3(c) takes the fused feature map as guidance to reconstruct the target frame from the warped feature. We empirically find that this improves the sharpness of the output frame while avoiding ghosting and glitching artifacts, as shown in the supplementary material.

Fusion function. We explore a *learning-based* fusion method using deep CNNs. Specifically, we train a CNN to predicts a blending weight for each neighboring frame using the encoded features, visibility masks, and the flow error (Figure 4):

$$f_{\mathrm{CNN}}^{k^t} = \sum_{n \in \Omega_k} f_{n^s}^{k^t} \underbrace{\sigma(G_\theta(f_{n^s}^{k^t}, M_{n^s}^{k^t}, f_{k^s}^{k^t}, M_{k^s}^{k^t}, e_{n^s}^{k^t}))}_{\omega_{n^s}^{k^t}}, \quad (1)$$

where G_θ is the CNN, $\sigma(\cdot)$ is a softmax activation, $f_{n^s}^{k^t}$ and $M_{n^s}^{k^t}$ are the encoded feature map and warping mask of frame n, respectively. The superscript k^t indicates that the encoded feature and warping mask are warped to the target stable frame k. The forward-backward flow consistency error e_{n^s} is calculated by:

$$e_{n^s}(\mathbf{p}) = \|F_{k^s \to n^s}(\mathbf{p}) + F_{n^s \to k^s}(\mathbf{p} + F_{k^s \to n^s})\|_2, \quad (2)$$

where $\mathbf{p}$ denotes the pixel coordinate in $F_{k^s \to n^s}$. The error $e_{n^s}^{k^t}$ in (1) is calculated by warping the flow consistency er-

ror e_{n^s} to the target frame k. All the flow consistency errors are calculated from the input unstabilized frames.

After fusing the feature, we concatenate the fused feature with the warped feature and warping mask of each frame as the input to the image generator. The image generator then predicts the output color frame and confidence map for each frame:

$$\{I_{n^s}^{k^t}, C_{n^s}^{k^t}\} = G_\phi(f_{n^s}^{k^t}, M_{n^s}^{k^t}, f_{\mathrm{CNN}}^{k^t}), \quad (3)$$

where G_ϕ denotes the image generator, $I_{n^s}^{k^t}$ and $C_{n^s}^{k^t}$ represent the predicted frame and confidence map of frame n in the virtual camera space at time k, respectively. Finally, the output stabilized frame I_{k^t} is generated by a weighted sum using these predicted frames and confidence maps:

$$I_{k^t} = \sum_{n \in \Omega_k} I_{n^s}^{k^t} C_{n^s}^{k^t}. \quad (4)$$

3.3. Path adjustment

When stabilizing a long video with large motion, some pixels around the frame boundary in the target frames may not be visible in *any* of the neighboring frames. For such cases, the network has to "hallucinate" new contents, often resulting in blurry predictions and unwanted visual artifacts.

To mitigate this problem, we propose to adjust the flow fields by global translations in each frame to increase the coverage of valid regions in the entire video. More specifically, for each target frame k, we aim to find a global translation $\mathbf{x}_{k^t}$ to adjust the flow fields of its neighbor frames as $F_{k^t \to n^s} + \mathbf{x}_{k^t}$, where $n \in \Omega_k$. Let the valid pixel mask $M_{\mathrm{valid}}^{k^t}(\mathbf{x}_{k^t})$ denote the *union* of the warping masks after the adjustment. We find the translations by optimizing the fol-
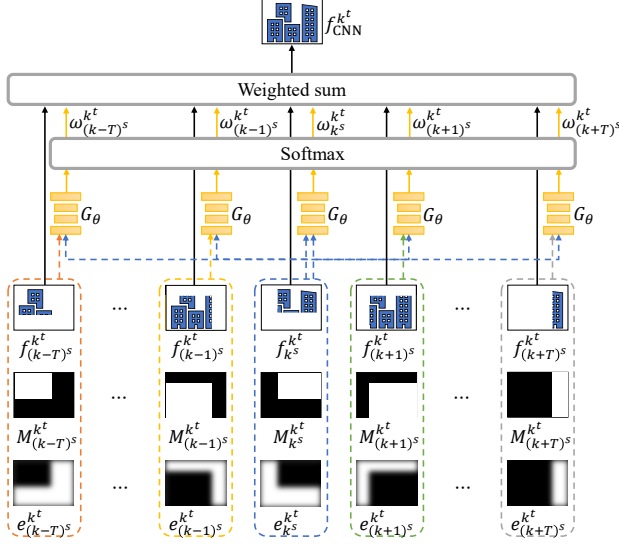
Figure 4: **Learning-based fusion.** Given the warped features $\{f_{n^s}^{k^t}\}_{n\in\Omega_k}$ (or warped images for image-space fusion), warping masks $\{M_{n^s}^{k^t}\}_{n\in\Omega_k}$, and the flow error maps $\{e_{n^s}^{k^t}\}_{n\in\Omega_k}$, we first concatenate feature, warping mask, and flow erro maps for each frame (shown as dotted blocks). We then use a CNN to predict the blending weights $\omega_{n^s}^{k^t}$ for each neighbor frame. Using the predicted weights, we compute the fused feature by weighted averaging the individual warped features $\{f_{n^s}^{k^t}\}_{n\in\Omega_k}$.

lowing energy function:

$$\arg\min_{\mathbf{x}_{k^t}} \sum_{k^t} (1 - M_{\text{valid}}^{k^t}(\mathbf{x}_{k^t})) + \lambda_{\text{s}} \sum_{k^t, q^t} \|\mathbf{x}_{k^t} - \mathbf{x}_{q^t}\|_2^2, \quad (5)$$

where $q \in \{k \pm 1\}$ indicates the neighbor frame of frame k. Here, the first term is a data term that aims to maximize the *coverage* of the valid mask. The second term is a smoothness term that penalizes large translation adjustments between nearby frames. The weight λ_{s} is a hyper-parameter that balances between the data and smoothness terms. We set $\lambda_{\text{s}} = 100$ in all the experiments and solve (5) using the coarse-to-fine alpha-expansion approach [3].

3.4. Training details

Loss functions. Our loss functions include the L1 and VGG perceptual losses:

$$\mathcal{L} = \||I_{k^t} - \hat{I}_{k^t}\||_1 + \sum_l \lambda_l \||\psi_l(I_{k^t}) - \psi_l(\hat{I}_{k^t})\||_1, \quad (6)$$

where $\hat{I}_{k^t}$ denotes the ground-truth target frame and ψ_l is intermediate feature extracted from a pre-trained VGG-19 network [43].

Training data. Our model requires a pair of unstabilized and stabilized videos for training. However, it is difficult to obtain such a real video pair. Therefore, we apply random motion to a stable video sequence to synthesize the input shaky video. Specifically, we sample a short sequence of 7 frames from the training set of [48] and randomly crop the frames to generate the input *unstable* video. We then apply another random cropping on the center frame as the ground-truth of the target *stabilized* frame.

4. Experimental results

We start with validating various design choices of our approach (Section 4.1). Next, we present quantitative comparison against representative state-of-the-art video stabilization algorithms (Section 4.2) and visual results (Section 4.3). In the supplementary material, we present ablation studies on path adjustment, residual detail transfer, additional detailed experimental results (comparisons with video completion method [12] and learning-based blending [18]), and stabilized videos by the evaluated methods as well as the user study and discuss the limitations of our approach. We will make the source code and pre-trained model publicly available.

4.1. Ablation study

We analyze the contribution of each design choice, including the fusion function design and fusion mechanism. We generate 940 test videos using the same method as our training data generation, where each video contains seven input frames and one target frame (see the supplementary material for more details).

Fusion function. We explore the following differentiable functions for fusion: 1) *Mean fusion*, 2) *Gaussian-weighted fusion*, 3) *Argmax fusion*, 4) *Flow error-weighted fusion*, and 5) the proposed *CNN-based fusion function*. We include the definition of the above fusion functions in the supplementary material. We train the proposed model using image-space fusion, feature-space fusion, and hybrid-space fusion. For image-space fusion, we also include two conventional fusion methods: multi-band blending [4] and graph-cut [1].

Table 1 shows the quantitative results of different fusion methods and fusion spaces. None of the fusion methods dominant the results for image-space fusion, where the argmax and CNN-based fusions perform slightly better than other alternatives. For both feature-space and hybrid-space fusion, the proposed CNN-based fusion shows advantages over other approaches.

Fusion space. Next, we compare different fusion levels using our CNN-based fusion. The last row of Table 1 shows that the proposed hybrid-space fusion achieves the best results compared to image-space fusion and feature-space fu-

Table 1: **Quantitative evaluation of fusion functions and fusion spaces on the test set of [48].** We highlight **the best** and the second best in each column.

	(a) Image-space fusion			(b) Feature-space fusion			(c) Hybrid-space fusion		
	LPIPS ↓	SSIM ↑	PSNR ↑	LPIPS ↓	SSIM ↑	PSNR ↑	LPIPS ↓	SSIM ↑	PSNR ↑
Multi-band blending	0.105	0.926	26.150	-	-	-	-	-	-
Graph-cut	0.105	**0.928**	26.190	-	-	-	-	-	-
Mean	0.123	0.899	24.618	0.108	0.878	26.028	0.099	0.898	27.013
Gaussian	**0.099**	0.920	25.310	0.095	0.874	26.344	0.097	0.899	27.080
Argmax	0.105	**0.928**	26.200	0.093	0.892	26.891	0.087	0.906	27.519
Flow error-weighted	0.114	0.901	25.363	0.096	0.885	26.176	0.095	0.911	27.371
CNN-based (Ours)	0.101	0.895	**26.692**	**0.092**	**0.902**	**27.187**	**0.073**	**0.914**	**27.868**

sion. The synthesized frame from the image-space fusion looks sharp but often contains visible glitching artifacts due to the discontinuity of different frames and inaccurate motion estimation. The results of the feature-space fusion are smooth but overly blurred. Finally, our hybrid-space fusion takes advantage of both above methods, generating a sharp and artifact-free frame. Please refer to the supplementary material and videos for more visual comparisons.

4.2. Quantitative evaluation

We evaluate the proposed method with state-of-the-art video stabilization algorithms, including L1Stabilizer[17], Bundle[31], StabNet[54], DIFRINT[10], and Yu and Ramamoorthi[62], and the warp stabilizer in Adobe Premiere Pro CC 2020. We obtain the results of the compared methods from the videos released by the authors or generated from the publicly available official implementation with default parameters or pre-trained models.

Datasets. We evaluate the above methods on the NUS dataset [31], the selfie dataset [60], and the DeepStab dataset [54]. The NUS dataset consists of 144 video sequences, which are classified into six categories: 1) Simple, 2) Quick rotation, 3) Zooming, 4) Parallax, 5) Crowd, and 6) Running. The selfie dataset includes 33 video clips with frontal faces and severe jittering. The DeepStab dataset includes 61 video clips with forward, pan, spin, and complex movements.

Metrics. We use the following metrics to evaluate the performance of video stabilization, which are widely used in prior work [31, 54, 10, 61]: **1)** *Cropping ratio*: measures the remaining frame area after cropping off the undefined pixels due to motion compensation. **2)** *Distortion value*: measures the anisotropic scaling of the homography between the input and output frames. **3)** *Stability score*: measures the stability and smoothness of the stabilized video. **4)** *Accumulated optical flow*: accumulates the optical flow over the entire stabilized video. Please refer to the supplementary material for a more detailed definition of each metric. Note that these metrics are used to evaluate the performance of video

stabilization, while in Section 4.1 we use PSNR, SSIM, and LPIPS to evaluate the quality of synthesized frames.

Results on the NUS dataset. We show the average scores on the NUS dataset [31] on the left side of Table 2. Both DIFRINT[10] and our method are full-frame methods and thus have an average cropping ratio of 1. Note that the distortion metric measures the global distortion by fitting a homography between the input and stabilized frames. Therefore, it is not suitable to measure local distortion. Although DIFRINT [10] obtains the highest distortion score, its results contain visible local or color distortion due to iterative frame interpolation, as shown in Figure 5 and the supplementary material. Adobe Premiere Pro 2020 warp stabilizer obtains the highest stability score at the cost of a low cropping ratio (0.74). Our results achieve the lowest accumulated optical flow, the second-best distortion and stability scores, and an average cropping ratio of 1 (no cropping for all the videos), demonstrating our advantages over the state-of-the-art approaches in the NUS dataset.

Results on the Selfie dataset. The middle of Table 2 shows the average scores on the selfie dataset [60]. Note that the videos in the dataset often contain large camera smooth and are more challenging to stabilize. Similar to the NUS dataset, our method achieves the best cropping ratio and accumulated flow. Our distortion and stability scores are comparable to Yu and Ramamoorthi's method [60], which is specially designed to stabilize selfie videos.

Results on the DeepStab dataset. We show the average scores on the DeepStab dataset [54] on the right side of Table 2. Our results achieve the second-best distortion, stability score, and accumulated optical flow without cropping.

Integrating with different flow smoothing methods. The proposed method can be easily integrated with existing flow smoothing methods to generate a full-frame stabilized video. Table 3 shows that the proposed method not only obtains full-frame results but also improves all metrics when combining with [31] or [62].

Table 2: **Quantitative evaluations with the state-of-the-art methods on the NUS dataset [31], the selfie dataset [60], and the DeepStab dataset [54].** We evaluate the following metrics: *Cropping Ratio* (C), *Distortion Value* (D), *Stability Score* (S), and *Accumulated Optical Flow* (A). **Red** text indicates the best and blue text indicates the second-best performing method.

	NUS dataset [31]				Selfie dataset [60]				DeepStab dataset [54]			
	C ↑	D ↑	S ↑	A ↓	C ↑	D ↑	S ↑	A ↓	C ↑	D ↑	S ↑	A ↓
Bundle [31]	0.84	0.93	0.84	0.78	0.68	0.76	0.82	0.84	0.76	0.91	**0.84**	**0.56**
L1Stabilizer [17]	0.74	0.92	0.82	0.88	0.75	**0.92**	0.84	0.84	0.74	0.91	0.82	0.70
StabNet [54]	0.68	0.82	0.81	1.02	0.69	0.79	**0.86**	0.83	0.64	0.86	0.83	0.80
DIFRINT [10]	**1.00**	**0.97**	0.82	0.87	**1.00**	0.85	0.81	0.72	**1.00**	**0.96**	0.83	0.78
Yu and Ramamoorthi [62]	0.86	0.91	0.83	0.88	0.77	0.83	0.82	0.77	0.83	0.92	0.83	0.72
Yu and Ramamoorthi [60]	-	-	-	-	0.85	0.89	0.84	0.76	-	-	-	-
Adobe Premiere Pro 2020 warp stabilizer	0.74	0.83	**0.86**	0.84	0.70	0.80	0.85	0.79	0.75	0.87	0.83	0.78
Ours	**1.00**	0.96	0.84	**0.77**	**1.00**	0.86	0.84	**0.64**	**1.00**	0.95	0.83	0.64

Figure 5: **Visual comparison to state-of-the-art methods.** Our proposed fusion approach does not suffer from aggressive cropping of frame borders and renders stabilized frames with significantly fewer artifacts than DIFRINT [10]. We refer the readers to our supplementary material for extensive video comparisons with prior representative methods.

Table 3: **Using different flow smoothing methods in our framework.** Our method can improve all four metrics on the Selfie dataset [60] when integrating with different flow smoothing methods.

	C ↑	D ↑	S ↑	A ↓
Bundle [31]	0.68	0.76	0.82	0.84
Ours + Bundle [31]	**1.00**	**0.78**	**0.83**	**0.70**
Yu and Ramamoorthi [62]	0.77	0.83	0.82	0.77
Ours + Yu and Ramamoorthi [62]	**1.00**	**0.86**	**0.84**	**0.64**

sive cropping is required. The DIFRINT method [10] generates full-frame results. However, their results often contain visible artifacts due to frame interpolation. In contrast, our method generates full-frame stabilized videos with fewer visual artifacts. Please refer to our supplementary material for more video comparisons.

5. Conclusions

We have presented a novel method for full-frame video stabilization. Our core idea is to develop a learning-based fusion approach to aggregate warped contents from multiple neighboring frames in a robust manner. We explore several design choices, including early/late fusion, heuristic/learned fusion weights, and residual detail transfer, and provide a systematic ablation study to validate the contributions of each individual component. Experiment results on three public benchmark datasets demonstrate that our method compares favorably against state-of-the-art video stabilization algorithms.

4.3. Visual comparison

We show the stabilized frame of our method and state-of-the-art approaches from the Selfie dataset in Figure 5. Most of the methods [17, 54, 31, 62] suffer from a large amount of cropping, as indicated by the green checkerboard regions. The aspect ratios of the output frames are also changed. To keep the same aspect ratio as the input video, more exces-

References

[1] Aseem Agarwala, Mira Dontcheva, Maneesh Agrawala, Steven Drucker, Alex Colburn, Brian Curless, David Salesin, and Michael Cohen. Interactive digital photomontage. In *ACM TOG*. 2004. 4, 6

[2] Kara-Ali Aliev, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. 2020. 4

[3] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *TPAMI*, 2004. 6

[4] Matthew Brown, David G Lowe, et al. Recognising panoramas. In *ICCV*, 2003. 6

[5] Chris Buehler, Michael Bosse, and Leonard McMillan. Non-metric image-based rendering for video stabilization. In *CVPR*, 2001. 1, 3

[6] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, 2001. 3

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

[8] Shenchang Eric Chen and Lance Williams. View interpolation for image synthesis. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, pages 279–288, 1993. 3

[9] Inchang Choi, Orazio Gallo, Alejandro Troccoli, Min H Kim, and Jan Kautz. Extreme view synthesis. In *ICCV*, 2019. 4

[10] Jinsoo Choi and In So Kweon. Deep iterative frame interpolation for full-frame video stabilization. *ACM TOG*, 2020. 2, 3, 7, 8

[11] Paul E Debevec, Camillo J Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pages 11–20, 1996. 3

[12] Chen Gao, Ayush Saraf, Jia-Bin Huang, and Johannes Kopf. Flow-edge guided video completion. In *ECCV*, 2020. 2, 3, 6

[13] Michael L Gleicher and Feng Liu. Re-cinematography: Improving the camerawork of casual video. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 2008. 3

[14] Amit Goldstein and Raanan Fattal. Video stabilization using epipolar geometry. *ACM TOG*, 2012. 1, 3

[15] Steven J Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F Cohen. The lumigraph. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pages 43–54, 1996. 3

[16] Matthias Grundmann, Vivek Kwatra, Daniel Castro, and Irfan Essa. Calibration-free rolling shutter removal. In *ICCP*, 2012. 1

[17] Matthias Grundmann, Vivek Kwatra, and Irfan Essa. Auto-directed video stabilization with robust l1 optimal camera paths. In *CVPR*, 2011. 3, 7, 8

[18] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM TOG*, 2018. 3, 4, 6

[19] Jia-Bin Huang, Sing Bing Kang, Narendra Ahuja, and Johannes Kopf. Temporally coherent completion of dynamic video. *ACM TOG*, 2016. 2

[20] Nima Khademi Kalantari and Ravi Ramamoorthi. Deep high dynamic range imaging of dynamic scenes. *ACM Trans. Graph.*, 36(4):144–1, 2017. 4

[21] Johannes Kopf, Michael F Cohen, and Richard Szeliski. First-person hyper-lapse videos. *ACM TOG*, 2014. 3

[22] Johannes Kopf, Kevin Matzen, Suhib Alsisan, Ocean Quigley, Francis Ge, Yangming Chong, Josh Patterson, Jan-Michael Frahm, Shu Wu, Matthew Yu, et al. One shot 3d photography. *ACM Transactions on Graphics (TOG)*, 39(4):76–1, 2020. 3

[23] Johannes Kopf, Xuejian Rong, and Jia-Bin Huang. Robust consistent video depth estimation. In *CVPR*, 2021. 3

[24] Ken-Yi Lee, Yung-Yu Chuang, Bing-Yu Chen, and Ming Ouhyoung. Video stabilization using robust feature trajectories. In *ICCV*, 2009. 1

[25] Sangwoo Lee, Jungjin Lee, Bumki Kim, Kyehyun Kim, and Junyong Noh. Video extrapolation using neighboring frames. *ACM TOG*, 2019. 4

[26] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the 23rd Annual conference on Computer Graphics and Interactive Techniques*, pages 31–42, 1996. 3

[27] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. *arXiv preprint arXiv:2011.13084*, 2020. 3

[28] Feng Liu, Michael Gleicher, Hailin Jin, and Aseem Agarwala. Content-preserving warps for 3d video stabilization. *ACM TOG*, 2009. 1, 3

[29] Feng Liu, Michael Gleicher, Jue Wang, Hailin Jin, and Aseem Agarwala. Subspace video stabilization. *ACM TOG*, 2011. 1, 3, 4

[30] Shuaicheng Liu, Yinting Wang, Lu Yuan, Jiajun Bu, Ping Tan, and Jian Sun. Video stabilization with a depth camera. In *CVPR*, 2012. 1, 3

[31] Shuaicheng Liu, Lu Yuan, Ping Tan, and Jian Sun. Bundled camera paths for video stabilization. *ACM TOG*, 2013. 1, 3, 4, 7, 8

[32] Shuaicheng Liu, Lu Yuan, Ping Tan, and Jian Sun. Steadyflow: Spatially smooth optical flow for video stabilization. In *CVPR*, 2014. 1, 3

[33] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM TOG*, 2019. 3, 4

[34] Yasuyuki Matsushita, Eyal Ofek, Xiaoou Tang, and Heung-Yeung Shum. Full-frame video stabilization. In *CVPR*, 2005. 1, 2, 3

[35] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 3, 4

[36] Simon Niklaus, Long Mai, Jimei Yang, and Feng Liu. 3d ken burns effect from a single image. *ACM TOG*, 2019. 3

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

[38] Gernot Riegler and Vladlen Koltun. Free view synthesis. In *ECCV*, pages 623–640, 2020. 3

[39] Gernot Riegler and Vladlen Koltun. Stable view synthesis. *arXiv preprint arXiv:2011.07233*, 2020. 3

[40] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *ICCV*, 2019. 4

[41] Steven M Seitz and Charles R Dyer. View morphing. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, 1996. 3

[42] Meng-Li Shih, Shih-Yang Su, Johannes Kopf, and Jia-Bin Huang. 3d photography using context-aware layered depth inpainting. In *CVPR*, 2020. 3

[43] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. 2015. 6

[44] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. Deepvoxels: Learning persistent 3d feature embeddings. In *CVPR*, 2019. 4

[45] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. 2019. 4

[46] Brandon M Smith, Li Zhang, Hailin Jin, and Aseem Agarwala. Light field video stabilization. In *ICCV*, 2009. 1, 3

[47] Pratul P Srinivasan, Richard Tucker, Jonathan T Barron, Ravi Ramamoorthi, Ren Ng, and Noah Snavely. Pushing the boundaries of view extrapolation with multiplane images. In *CVPR*, 2019. 3

[48] Shuochen Su, Mauricio Delbracio, Jue Wang, Guillermo Sapiro, Wolfgang Heidrich, and Oliver Wang. Deep video deblurring for hand-held cameras. In *CVPR*, 2017. 6, 7

[49] Narayanan Sundaram, Thomas Brox, and Kurt Keutzer. Dense point trajectories by gpu-accelerated large displacement optical flow. In *ECCV*, 2010. 4

[50] Richard Szeliski. Image alignment and stitching: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 2(1):1–104, 2006. 4

[51] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. 2020. 4

[52] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *ACM TOG*, 2019. 4

[53] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a deforming scene from monocular video. *arXiv preprint arXiv:2012.12247*, 2020. 3

[54] Miao Wang, Guo-Ye Yang, Jin-Kun Lin, Song-Hai Zhang, Ariel Shamir, Shao-Ping Lu, and Shi-Min Hu. Deep online video stabilization with multi-grid warping transformation learning. *IEEE Transactions on Image Processing*, 2018. 3, 7, 8

[55] Yu-Shuen Wang, Feng Liu, Pu-Sheng Hsu, and Tong-Yee Lee. Spatially and temporally optimized video stabilization. *IEEE Transactions on Visualization and Computer Graphics*, 2013. 1, 3

[56] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. Synsin: End-to-end view synthesis from a single image. In *CVPR*, 2020. 3

[57] Shangzhe Wu, Christian Rupprecht, and Andrea Vedaldi. Unsupervised learning of probably symmetric deformable 3d objects from images in the wild. In *CVPR*, 2020. 3

[58] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. *arXiv preprint arXiv:2011.12950*, 2020. 3

[59] Sen-Zhe Xu, Jun Hu, Miao Wang, Tai-Jiang Mu, and Shi-Min Hu. Deep video stabilization using adversarial networks. In *Computer Graphics Forum*, 2018. 3

[60] Jiyang Yu and Ravi Ramamoorthi. Selfie video stabilization. In *ECCV*, 2018. 3, 7, 8

[61] Jiyang Yu and Ravi Ramamoorthi. Robust video stabilization by optimization in cnn weight space. In *CVPR*, 2019. 1, 3, 7

[62] Jiyang Yu and Ravi Ramamoorthi. Learning video stabilization using optical flow. In *CVPR*, 2020. 1, 2, 3, 4, 7, 8

[63] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *ACM TOG*, 2018. 3

[64] Zihan Zhou, Hailin Jin, and Yi Ma. Plane-based content preserving warps for video stabilization. In *CVPR*, 2013. 1