# SensorFlow: Sensor and Image Fused Video Stabilization

Jiyang Yu Tianhao Zhang Fuhao Shi Lei He Chia-Kai Liang
Google

## Abstract

*We present SensorFlow, a novel image and sensor fusion framework for robust, high-quality video stabilization. We start with sensor-based pre-stabilization that smooths out large-scale camera motion. A new angular velocity domain optimization has been introduced to achieve frame rate invariance. We then feed the stabilized optical flows into an occlusion-aware 3D CNN that infers dense warp fields to remove residual translation and jitter. To further avoid distortion, we propose a novel masking scheme to determine the disoccluded and dynamic regions in optical flow and inpaint them with spatially smooth flow vectors. Our method is appealing as it shares both the dense warping field's flexibility to correct complex motions and the robustness of sensor data for arbitrarily challenging scenes. We have validated its effectiveness and demonstrated our solution outperforms state-of-the-art alternatives via extensive ablation studies and quantitative comparisons.*

## 1. Introduction

Video stabilization is crucial for creating smooth and professional-looking footage. It eliminates the shaky, distracting hand-held vibrations that often plague videos, especially when shot on smartphones or action cameras.

Despite decades of research in video stabilization and a plethora of approaches, achieving high-quality, robust video stabilization remains a significant challenge due to the complexities of real-world scenes and camera motions. Video stabilization often follows a ***track-smooth-and-warp*** process. The ***tracking*** stage uses techniques like feature tracking [10, 18, 19, 22] and optical flow [24, 37, 38] to model camera motion. However, even with mature algorithms like KLT [1] or deep learning-based methods like RAFT [33], the detected motion might not be directly suitable for stabilization. Large dynamic scenes with disocclusion often require additional handling, typically involving hand-crafted metrics [38] or rigidity regularization [18], which can introduce errors and reduce stability. Furthermore, the ***smooth-and-warp*** stage faces difficulties in achieving artifact-free results. Grid-based [18,20,22,35] and dense per-pixel warping [20, 24, 38] struggle to balance stability and visual distortion in the presence of non-rigid objects. While interpolation [7] and 3D synthesis [15, 27] show potential, their quality and robustness are inferior to specialized stabilization algorithms. The core challenge lies in disentangling the combined effects of camera motion, depth, rolling shutter, and potential errors in motion detection algorithms. Limitations in motion tracking and frame synthesis process make it impractical to model these factors simultaneously.

In contrast, most on-device camera stabilization [17] (often called electronic image stabilization, i.e. EIS) relies on sensors like gyroscopes and optical image stabilization (i.e. OIS). These signals are detached from video content and the image transformations are computed solely to compensate motion detected by sensors, making them able to handle large motions without image-based tracking. However, it is usually limited to rotational motion. Additionally, sensor are susceptible to calibration errors and environmental factors like temperature, potentially causing jitters and drift.

In this paper, we propose a novel video stabilization pipeline that draws the strengths of both sensor-based and vision-based solutions. On high-level, our pipeline includes two steps. First, we introduce sensor-based pre-stabilization which leverages sensor information to remove large-scale camera motion. A novel angular-velocity-based optimization is used to achieve frame rate invariance during path smoothing (Sec. 3.2). Second, we perform image-based refinement on top of the pre-stabilized video. We introduce a 3D convolutional neural network (Sec. 4) that infers dense warp fields to eliminate residual translational motion and jitter. To ensure robustness to disocclusion and dynamic object, we introduce a novel occlusion-aware loss into training, and further propose a novel masking scheme to determine dynamic regions and inpaint with spatially smooth flow vectors (Sec. 3.3). Our video stabilization pipeline has shown visually pleasing results and achieved state-of-the-art performance under various quantitative metrics compared to existing vision-based and sensor-vision hybrid video stabilization algorithms (Sec. 5). This work has made the following contributions:

- A novel 3D convolutional video stabilization network with occlusion-aware training (Sec. 4).
- A novel masking strategy for dynamic object and disocclusion handling (Sec. 3.3).
- A more effective quantitative metric for evaluating the stability of videos (Sec. 5.2).

## 2. Related Work

Stabilization solutions can be categorized into image-based, sensor-based and image-sensor fusion. While clas-

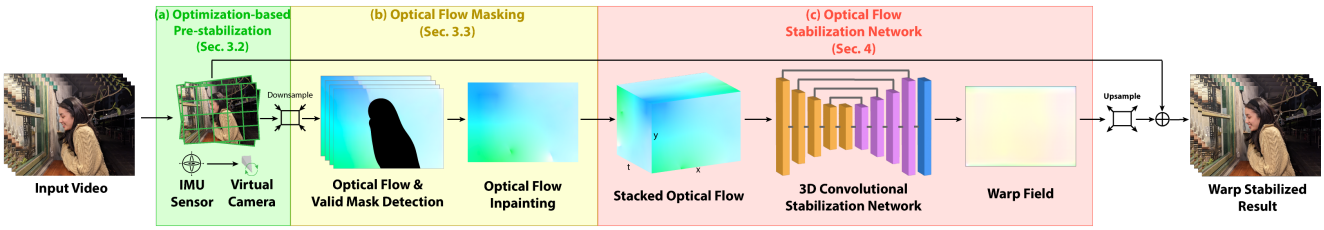


Figure 1. **Overview of our pipeline.**(a) Sensor-based pre-stabilization smooths rotation and minor translation motions based on sensor data. (b) The flow masking estimates the optical flow, detects dynamic and disocclusion regions, and inpaint with spatially smooth flow. (c) The stabilization network infers warp fields for each frame from flow fields stacked along time dimension.

sic methods typically follow the ***track-smooth-warp*** design [28], recent learning-based methods directly map unstabilized frames or optical flow to stabilized results [6, 25, 34, 37, 38, 41–43].

**Image-based methods.** Conventional image-based methods typically start with motion estimation, which can be done in 2D, 3D and hybrid manner. The 2D approaches represent and estimate camera motions as a series of 2D affine or perspective transformations [11, 22, 26] which are estimated via robust feature from salient features [20, 40] or optical flow [24], followed by outlier rejection. The 3D approaches model the camera poses and estimate a smooth virtual camera trajectory in the 3D space. The 6DoF camera poses can be obtained from projective 3D reconstruction [4], depth camera [21], structure from motion [18], light-field [32], and recently a joint pose and depth estimate network [13, 29]. 3D nerual representation and image rendering methods also show potential for stabilization [15, 16], though they are computationally heavy and have limited robustness to dynamic scene and motion variations. Some hybrid 2D-3D approaches exploit the subspace constraints [19] and epipolar geometry [9]. Selfie stabilization further stabilizes the face trajectory on top of the background [36, 39]. To smooth the camera motion, approaches like low-pass filtering on camera parameters [5, 26], $\mathcal{L}_1$ optimization [11], $\mathcal{L}_2$ optimization [20, 24] and joint optimization with bundled local camera paths [20, 22] have been developed. In contrast, recent learning methods takes unstabilized frames or the optical flow as input, and stabilize the frames by inferring a sparse mesh grid [34] or dense warping field [38, 39, 41–43]. Choi et al. [7] learn a frame interpolation model to iteratively interpolate the input video into a stable one without cropping.

**Sensor and fusion based methods.** Sensor-based stabilization uses IMU information to estimate 3D camera pose and stabilize the videos along a smoothed camera path [3, 12, 14, 17]. The advantages of using sensor information include low latency and high robustness, but it suffers from calibration errors and runtime drift. It also lacks the ability to model parallax and depth effect in the scene. Recently, sensor-image fusion solutions [30, 45] have been developed to address these limitations, including referring to 2D face trajectory or stabilizing residual 2D motion with feature matching. However, these methods relies on simple parametric 2D motion models which fail to model residual motion and parallax accurately. Shi et al. [31] proposes a DNN that directly infers smoothed camera path from IMU and optical flow. While optical flow is used, their camera model still stabilizes the frame globally, therefore cannot properly handle parallax.

Our work introduces a novel sensor and optical flow fusion framework with a robust masking strategy to tackle challenging dynamic scenes. Our work has important differences from existing works. First, fusion with the dense warp field enhances the robustness to sensor inaccuracy, enabling local parallax correction without introducing distortion. Second, we introduced a novel stabilization network with a occlusion-aware training that operates on both spatial and temporal dimension and stabilize residual motion more effectively. Finally, we developed a novel masking strategy and occlusion-aware training loss based on 2D/3D mixed geometric constraint, making our method exceptionally robust to dynamic object and disocclusions. We have demonstrated that our method outperforms state-of-the-art alternatives in both stability and robustness.

## 3. SensorFlow Pipeline

### 3.1. Overview and Notations

Fig. 1 illustrates an overview of the proposed pipeline which consists of two stages. The first stage is the sensor-based stabilization (Fig. 1(a)). In this stage, we extract camera rotation information [31] based on sensor data from gyroscope and OIS, and run an optimization-based algorithm to smooth large rotation motions and small translational motions. We will discuss more details in Sec.3.2.

To further smooth the residual translational motions, the second stage computes and processes optical flows (Fig. 1(b)) based on the sensor-stabilized frames, and uses a deep 3D convolutional network to predict per-pixel dense warp fields to stabilize the sequence in an iterative manner(Fig. 1(c)).

As we will discuss in Sec. 5, optical flow based video stabilization is vulnerable to disocclusion and dynamic objects. To avoid local artifacts in the stabilized video, the input flows are downsampled and the flow stabilization network is designed to operate at a low resolution ($320 \times 240$) so that the output dense warp field is spatially smoother.

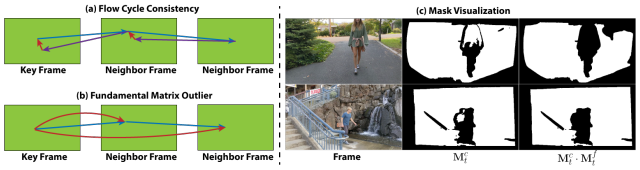


Figure 2. Demonstration of optical flow masking strategy. (a) Flow cycle consistency mask $\mathbf{M}_t^c$. (b) Fundamental matrix outlier mask $\mathbf{M}_t^f$. (c) Visualization of masks, black is the *invalid region*.

In addition, we introduce the optical flow masking algorithm in Sec. 3.3, which removes and inpaints optical flow regions that could potentially introduce distortion artifacts in the final warp field. We then discuss the structure and training details of the deep flow stabilization network in Sec. 4. To generate the final stabilized frame, the output warp field is upsampled to the original resolution and combined with output of the sensor-based stabilization to warp the input frame.

Now we define the general notations that will be used throughout this paper. For a video with $T$ frames, we denote the sequence of frames as $\{\mathbf{I}_0, ..., \mathbf{I}_t, ..., \mathbf{I}_T\}$. The output of the first stage is the downsampled sensor-stabilized frames $\{\mathbf{I}_{\downarrow t}\}$. We use RAFT [33] to compute the bi-directional optical flow fields between the consecutive frames, namely $\mathbf{F}_t$ is the optical flow from $\mathbf{I}_{\downarrow t}$ to $\mathbf{I}_{\downarrow t+1}$, and $\mathbf{B}_t$ is the optical flow from $\mathbf{I}_{\downarrow t+1}$ to $\mathbf{I}_{\downarrow t}$.

### 3.2. Optimization-based Pre-stabilization

In this section, we introduce the pre-stabilizion using sensor information. We integrate camera rotation from gyroscope samples and frame timestamps, and denote the camera parameters as $P = (R, O)$ similar to Shi et al. [31], where $R$ is the camera rotation and $O$ is the 2D principal offset.

Instead of constraining the rotation smoothness directly with quaternions, we propose to represent the virtual camera motion with the angular velocity. In this way, both the 0th and 1st order smoothness terms are now frame rate invariant and smoother camera paths are obtained. Specifically, we measure the 0th and 1st order smoothness of the virtual camera rotation by

$$L_{C^0(R)} = ||\Omega_v(t)||_1, \tag{1}$$
$$L_{C^1(R)} = ||\Omega_v(t) - \Omega_v(t - \Delta t)||_2, \tag{2}$$

where $\Omega_v(t) = (\omega_v^x(t), \omega_v^y(t), \omega_v^z(t))$ is the 3D virtual angular velocity. We use $\ell_1$ norm to encourage the virtual camera to stay still when possible. We integrate the virtual camera pose as $R_v(t) = \int_t^{t+\Delta t} \Omega_v(t)\, dt$, and the protrusion term $L_p$ follows the definition in Shi et al. [31].

Another key difference is that we do not use image information at this stage, so there is no need to extract OIS-free optical flow like Shi et al. [31]. Instead of fixing the virtual OIS offset $O_v$, we now allow it to move smoothly functioning as a virtual OIS. Two extra offset smoothness terms as

$$L_{C^0(O)} = ||O_v(t) - O_v(t - \Delta t)||_2, \tag{3}$$
$$L_{C^1(O)} = ||O_v(t) + O_v(t - 2\Delta t) - 2O_v(t - \Delta t)||_2. \tag{4}$$

The virtual camera pose is obtained by minimizing the following losses.

$$\begin{aligned} L = &\lambda_{C^0(R)} L_{C^0(R)} + \lambda_{C^1(R)} L_{C^1(R)} + \\ &\lambda_{C^0(O)} L_{C^0(O)} + \lambda_{C^1(O)} L_{C^1(O)} + \lambda_p L_p, \end{aligned} \tag{5}$$

where $\lambda_{C^0(R)}, \lambda_{C^1(R)}, \lambda_{C^0(O)}, \lambda_{C^1(O)}, \lambda_p$ are set to $0.002, 0.25, 50, 135$ and $0.2$ respectively in our experiments.

### 3.3. Optical Flow Masking Strategy

Optical flow masking is another critical component in our pipeline. The purpose of this process is to detect any optical flow region that could potentially introduce distortion artifact in the stabilized result. Such cases include disocclusion and dynamic objects, where the 2D optical flow could not well represent the actual 3D scene motion.

We name the regions that contain these erroneous flow vectors as *invalid regions*, and denote the per-frame invalid region mask as $\mathbf{M}_t$, where $\mathbf{M}_t(x, y) = 0$ if a pixel $(x, y)$ is invalid, and $\mathbf{M}_t(x, y) = 1$ if the pixel is valid. Our invalid region detection algorithm consists of two parts.

**Flow cycle consistency** is commonly used in detecting disocclusion. In our algorithm, to compute $\mathbf{M}_t$, we consider a group of its neighbor optical flows $[\mathbf{F}_{t-\omega}...\mathbf{F}_{t+\omega}]$ and $[\mathbf{B}_{t-\omega}...\mathbf{B}_{t+\omega}]$. We track a pixel from the frame $t$ in both forward and backward direction, and compute the cycle consistency mask in each step:

$$\begin{aligned} \mathbf{M}_t^c = &\prod_{\tau=t}^{t+\omega} \left( \left\| \widehat{\mathbf{F}}_\tau + \mathbf{B}_\tau[\widehat{\mathbf{F}}_\tau] \right\|_2 < threshold \right) \cdot \\ &\prod_{\tau=t}^{t-\omega} \left( \left\| \widehat{\mathbf{B}}_\tau + \mathbf{F}_\tau[\widehat{\mathbf{B}}_\tau] \right\|_2 < threshold \right), \end{aligned} \tag{6}$$

where $[\cdot]$ is the spatial sampling operator, $\widehat{\mathbf{F}}_\tau = \mathbf{F}_\tau[\mathbf{F}_{\tau-1}]$ and $\widehat{\mathbf{B}}_\tau = \mathbf{B}_\tau[\mathbf{B}_{\tau+1}]$ are the tracked optical flow. We use 1 as the threshold in our experiments. We illustrate this process in Fig 2(a), where the blue arrows are forward optical flow, purple arrows are backward optical flow. We threshold the cycle consistency error and accumulate the mask as our flow cycle consistency mask, i.e. the pixel is valid if the length of red arrows are small for all frames.

**Fundamental matrix outlier** is another method we use to detect flow regions with dynamic objects. Even though flow cycle consistency is able to detect inaccurate flows, some dynamic objects can survive the test because when the object is not moving very fast, optical flow models can still track the movement of the pixels correctly. Such cases are also observed when the moving object is relatively domi-

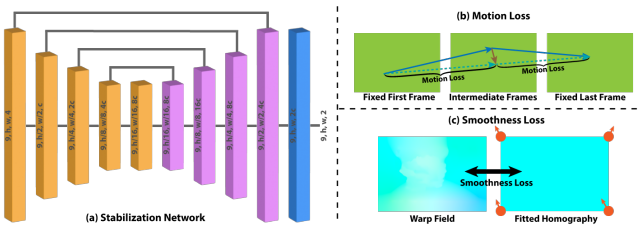


Figure 3. Network structure and training losses. (a) Our stabilization network follows a 3D autoencoder structure, the input tensor size for each layer is noted. (b) Motion loss aims to minimize the overall motion magnitude. (c) Smoothness loss enforces the warp field to be close to a global homography.

nant in the video frame.

To have a more robust detection of regions of moving objects we leverage the pixel correspondences to compute the fundamental matrix between the keyframe and its neighboring frames. The principle of this idea is that pixels belonging to the "static background" should follow the motion that satisfies the same fundamental matrix, no matter at which depth they reside. A pixel will be invalid if it is an outlier in any of the estimated fundamental matrices. Formally, the binary mask would be

$$\mathbf{M}_t^f = \prod_{\tau=t}^{t+\omega} \left\{(i,j) \mid (i,j) \in S_{inlier}^{t \to t \pm \tau}\right\} \tag{7}$$

where $S_{inlier}^{t \to t+\tau}$ denotes the set of inlier pixels for the fundamental matrix estimated with the flow from frame $t$ to frame $t+\tau$. A diagram of the fundamental matrices computed is depicted in Fig 2(b) where the blue arrows are forward optical flows, and the red arrow indicates the frame pairs that the fundamental matrices are computed on.

Therefore the overall optical flow invalid region mask is

$$\mathbf{M}_t = \mathbf{M}_t^c \cdot \mathbf{M}_t^f. \tag{8}$$

Note that to prevent artifact accumulation, our mask detection covers the entire sequence that stabilization network can observe (Sec. 4), and extended backward to prevent error propagated from previous frames. We use $\omega = 10$ in Eq. 6 and 7. In Fig. 2, we show examples of $\mathbf{M}_t^c$ which removes disocclusion and out-of-bound flow tracking regions, and $\mathbf{M}_t^f$ effectively removes the dynamic objects.

#### 3.3.1 Optical Flow Inpainting

As discussed above, we exclude the regions that might cause problems in the optical flow stabilization with various mask strategies. However, the flow stabilization model takes complete optical flow fields as input, which requires inpainting appropriate values in the invalid regions. We use Laplacian region filling [8], which guarantees the local smoothness of invalid regions by enforcing each pixel value equal to the mean of its four adjacent pixels.

## 4. Optical Flow Stabilization Network

### 4.1. 3D CNN for Stabilization

In this section, we introduce the video stabilization network structure. Unlike previous optical flow based video stabilization network with a 2D autoencoder structure [38], our stabilization network has a 3D convolutional structure that operates on both spatial and temporal dimension. The intuition behind our design is that video stabilization is a temporal filtering problem, while certain spatial constraints should be satisfied, e.g. shapes of objects should remain the same and the warp field should be spatially smooth. Therefore, the stabilization network should be trained as a 3D filter that performs this filtering process intelligently.

Fig. 3(a) shows the network structure of our video stabilization model. The input of the network is a stacked tensor of the bi-directional optical flow fields $\{\mathbf{F}_\tau, \mathbf{B}_\tau\}_t^{t+l} \in \mathbb{R}^{l \times h \times w \times 4}$. Our stabilization network is a 3D autoencoder that convolves both temporal ($l$) and spatial ($h$ and $w$) dimension, except that the temporal dimension does not downsample between layers and the size is always $l$. The output tensor size of our network is $l \times h \times w \times 2$. Note that the number of output warp fields is $l$, but only the first $l-1$ are effective and used for the training loss that we will discuss in Sec. 4.2. Our network takes a limited length window as input. To process arbitrary long video, we utilize the same sliding window process in [38].

### 4.2. Training Loss

The training process is self-supervised like [38]. Given a window of $l+1$ video frames, of which the input is the stacked optical flow fields $\{\mathbf{F}_\tau, \mathbf{B}_\tau\}_t^{t+l}$ to the network, our goal is to train a neural network to infer the warp fields $[\mathbf{W}_{t+1} ... \mathbf{W}_{t+l-1}]$ for each intermediate frames from $t+1$ to $t+l-1$ so that the overall motion of the window is minimized with the first and last frame being fixed ($\mathbf{W}_t = 0$ and $\mathbf{W}_{t+l} = 0$).

The training loss consists of the motion loss and the smoothness loss. The former one mainly focuses on the temporal stability, and the latter one mainly focuses on the spatial smoothness.

**Motion loss** The motion loss can be formulated as:

$$\begin{aligned} L_m = &\sum_{\tau=t}^{t+l-1} \left\|\mathbf{F}_\tau + \mathbf{W}_{\tau+1} - \mathbf{W}_\tau\right\|_1 + \\ &\sum_{\tau=t}^{t+l-1} \left\|\mathbf{B}_\tau + \mathbf{W}_\tau - \mathbf{W}_{\tau+1}\right\|_1. \end{aligned} \tag{9}$$

Fig. 3(b) demonstrates the motion loss. The green arrow indicates the optical flow between consecutive frames, and the motion loss aims to minimize the residual motion (blue arrow) with inferred warp field (black arrow).

**Smoothness loss** In real practice, optical flow is not completely reliable for video stabilization purposes. Our flow

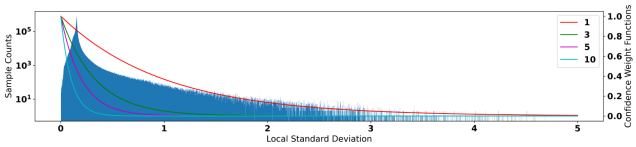


Figure 4. Local standard deviation distribution and the confidence weight functions that map $\mathbf{D}_t$ to the confidence weight $\widetilde{\mathbf{D}}_t$.

masking strategy (Sec. 3.3) is designed for dynamic objects, but does not cover other potential optical flow problems like spatial noise and inaccuracy in uniform color/repetitive pattern regions. To this end, we introduce a regularization term called smoothness loss to enforce the spatial smoothness of the warp field (Fig. 3(c)). We parameterize a global homography as the interpolated motion of the 4 corner pixels' motion (denoted as $\mathbf{H}_t \in \mathbb{R}^{4\times 2}$), and estimate this virtual global homography from the inferred warp field:

$$\mathbf{H}_t = (\mathbf{J}^T\mathbf{J})^{-1}\mathbf{J}^T\overline{\mathbf{W}}_t, \tag{10}$$

where $\overline{\mathbf{W}}_t \in \mathbb{R}^{hw\times 2}$ is the reshaped warp field, and $\mathbf{J} \in \mathbb{R}^{hw\times 4}$ is the bilinear interpolation weights. The spatial smoothness of the warp field can be evaluated as the error of this homography fitting:

$$L_s = \sum_{\tau=t+1}^{t+l-1} \left\|\mathbf{J}\mathbf{H}_t - \overline{\mathbf{W}}_t\right\|_1. \tag{11}$$

**Occlusion-aware Training** In addition to the global spatial smoothness, we also introduce a stronger local weighting on the training loss to reinforce the robustness to erroneous optical flow. The intuition is to emphasize the importance in the reliable optical flow region, but penalize the unreliable optical flow region in the training. In our work, we evaluate the reliability of optical flow using a $10 \times 10$ local standard deviation of the optical flow field:

$$\mathbf{D}_t = \sigma(\mathbf{F}_t(x)) + \sigma(\mathbf{F}_t(y)), \tag{12}$$

where smaller $\sigma$ indicates a more reliable flow region.

To use $\mathbf{D}_t$ as a spatial weight to the training loss, we introduce a mapping between the local standard deviations and the training weights. Fig. 4 shows the histogram of the $\mathbf{D}_t$ value of $2.76 \times 10^8$ flow vectors from our training data. According to the distribution of $\mathbf{D}_t$, we use the inverse exponential function to map $\mathbf{D}_t$ to training weights:

$$\widetilde{\mathbf{D}}_t = e^{-\delta\mathbf{D}_t}, \tag{13}$$

where $\delta$ is a hyperparameter controlling the amount of penalty to the unreliable optical flow regions. Mapping functions with various $\delta$ values are shown in Fig. 4. Note that we will discuss the effect of $\delta$ in Sec. 5.3. Applying the reliability weighting on the motion loss yields:

$$\begin{aligned} L_d = &\sum_{\tau=t}^{t+l-1} \widetilde{\mathbf{D}}_\tau \odot \left\|\mathbf{F}_\tau + \mathbf{W}_{\tau+1} - \mathbf{W}_\tau\right\|_1 + \\ &\sum_{\tau=t}^{t+l-1} \widetilde{\mathbf{D}}_{\tau+1} \odot \left\|\mathbf{B}_\tau + \mathbf{W}_\tau - \mathbf{W}_{\tau+1}\right\|_1. \end{aligned} \tag{14}$$

To sum up, our training loss can be written as $L = L_d + \lambda L_s$, where $\lambda$ is the smoothness loss weight.

### 4.3. Training Data

We use the RealEstate10K dataset [44] for our model training. It provides point clouds and camera poses, but only RGB frames are used in our training. Each training batch consists of ten frames in $320 \times 240$ resolution. Note that although our training data only consists of smooth camera motions and static scenes, it is rich in occlusion and parallax due to its nature. We augment the motion patterns with random affine transformations, so that the resulting optical flows used to train the stabilization network are complex enough to simulate real-world occlusions introduced by dynamic objects. The affine transformation is generated by randomly sampling a rotation matrix and a 2D translation to the image frame. We generate the rotation matrix by first sampling a random rotation degree from uniform distribution and then applying it to a random rotation center that is close to the frame center. After applying the transformation to the input image, the image is also cropped to ensure no invalid regions are visible in the final augmented result.

## 5. Results

In this section, we compare our method with various video stabilization methods including sensor/optical flow hybrid method (Shi et al. [31]), sparse optical flow based method (Zhang et al. [41]), dense optical flow based method (DUT [35] and Yu et al. [38]), image based method (PWStableNet [42] and Wang et al. [34]), 3D video stabilization (Deep3D [13]), interpolation based method(Choi et al. [7]), and traditional feature track based method (Grundmann et al. [11]). Beyond academic works in video stabilization, we also compare our method with the industry leading video stabilization in iPhone 15 Pro in the supplementary video.

We test all the methods on a collection of 34 testing videos proposed in Shi et al. [31] as it contains the sensor data. For fair comparison, we also provide a version of our results without using any sensor information, where our sensor-based pre-stabization (Sec. 3.2) is replaced by a simple optimization based on feature track smoothing. We show that our method achieves superior result both qualitatively (Sec. 5.1) and quantitatively (Sec. 5.2), and can achieve competitive result even without the sensor information. Since dynamics are not observable in still figures, we encourage reader to watch the supplementary video.

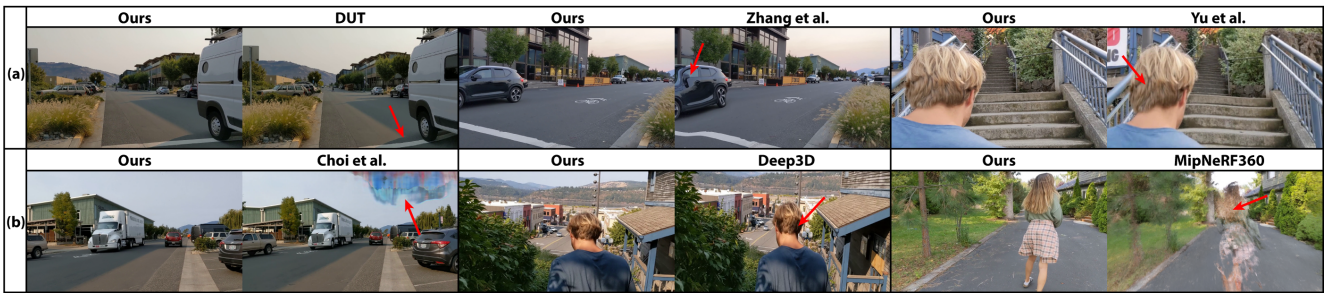


Figure 5. Qualitative comparison on example video stills. (a) Optical flow based methods DUT [35], Zhang et al. [41], and Yu et al. [38] usually distort local regions due to disocclusion. (b) Non-traditional based methods, like interpolation (Choi et al. [7]), 3D stabilization (Deep3D [13]) and novel view synthesis (MipNeRF360 [2]) are not robust enough and introduce visual defects. All artifacts are marked with red arrows.

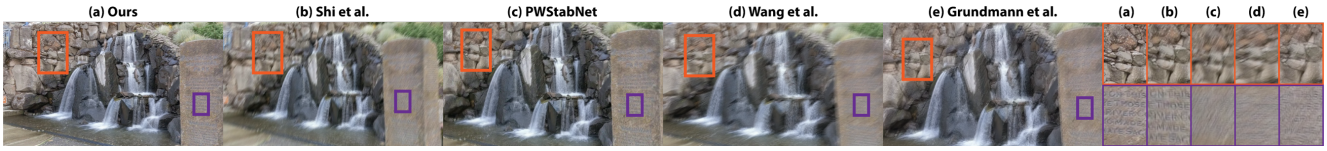


Figure 6. Stability comparison with image based methods (PWStableNet [42] and Wang et al. [34]) and feature based method (Grundmann et al. [11]). Each image is generated by averaging 11 consecutive frames of the stabilized video. Insets show zoomed local regions for better comparison. Less blurry indicates a more stable result.

### 5.1. Qualitative Comparison

We show the qualitative comparison in Fig. 5. In Fig. 5(a), we compare our result with optical flow based methods DUT [35], Zhang et al. [41], and Yu et al. [38]. As discussed in Sec. 1, optical flow is not directly suitable for video stabilization due to disocclusion. Even with additional optical flow handling like sparse optical flow (Zhang et al. [41]), deep learning motion refinement (DUT [35]) and smooth flow fitting (Yu et al. [38]), these methods still introduce significant distortion around dynamic objects, e.g. distorted straight lines/objects (marked by red arrows in Fig. 5(a)). Our optical flow masking algorithm, combined with the occlusion-aware training, is able to handle dynamic objects by its nature and achieves distortion-free results. Fig. 5(b) shows comparison with non-traditional video stabilization methods, including Deep3D [13] and Choi et al. [7]. We also include a result generated by the novel view synthesis method MipNeRF360 [2], where we render the scene on a Gaussian filtered camera path. Due to the challenges in the 3D reconstruction and the 2D image interpolation, these methods are less robust and produces visually unsatisfactory results (artifacts are marked with red arrows in Fig. 5(b)). Note that since MipNeRF360 introduced serious artifacts and failed metric computation, we do not include its quantitative result in Sec. 5.2. Our stabilization network is able to generate spatially smooth warp field, and the warp stabilized frames do not contain visual artifacts.

While maintaining visual quality of the stabilized video, our method also achieves significantly superior stability compared to the comparison methods. Fig. 6 shows comparison of the averaged frame over 11 consecutive frames. Note that the more stable the video is, the less blurry the averaged frame is. Although the image based and feature based methods (PWStableNet [42], Wang et al. [34] and Grundmann et al. [11]) do not introduce local distortion in most scenarios, the stability are usually sacrificed. Thanks to the additional optical flow based stabilization network, our method is able to reduce the residual shake that cannot be modeled from sensor data. This makes our result more stable compared to previous methods that mainly depends on sensor data (Shi et al. [31]).

### 5.2. Quantitative Comparison

For quantitative comparison, we evaluate the performance on the Shi et al. [31] test set using the three metrics commonly used in video stabilization method evaluations [11, 38]: *Stability*, *Distortion*, and *Cropping*.

**Stability** measures the proportion of the low frequency motion in the overall motion of the stabilized video, which is the most important metric for evaluating video stabilization algorithms. However, in our experiment under the standard definition of stability metric [23], we observe that 1) most methods have only subtle differences compared to the scale of the metric score and 2) significant discrepancy exists in visual stability and the metric score ranking, which is also observed in Shi et al. [31]. The reason of these phenomena is that the motion signal used for FFT contains the intentional motion of the input video which dominates the low frequency band, making the effect of stabilization subtle. Therefore, stability metric in [23] is not ideal for evaluating the stability of a video. To justify this claim, we show a synthetic example that demonstrates this general problem in Fig. 7: for frame motions with different levels of shake (a) $\sigma = 0.1$ and (b) $\sigma = 1$, the standard stability scores are similar (0.4922 vs. 0.4954) and the more shaky one (b) achieves

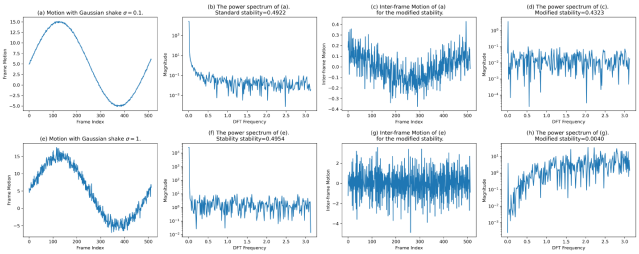

Figure 7. Comparison of stability metric proposed in Liu et al. [23] and our stability metric. Comparing two arbitrary intentional motions with different shake magnitudes (a and e), the stability metric proposed by Liu et al. [23] failed and the more shaky motion gets a higher score (b and f). With our improved stability metric, we ruled out the influence of intentional motion (c and g) and properly evaluates the stability (d and h). Best viewed when zoomed in.

| Methods | (a) Shi et al. Dataset [31] | | | (b) NUS Dataset [23] | | |
|---|---|---|---|---|---|---|
| | Stability | Distortion | Cropping | Stability | Distortion | Cropping |
| Input | 0.2906 | 1.0000 | 1.0000 | 0.3245 | 1.0000 | 1.0000 |
| Zhang et al. [41] | 0.1980 | 0.7051 | 0.6989 | 0.2432 | **0.8980** | 0.7737 |
| DUT [35] | 0.4694 | 0.8457 | 0.7986 | 0.3853 | 0.8180 | 0.6912 |
| Yu et al. [38] | 0.3778 | 0.7905 | **0.8680** | 0.3581 | 0.7242 | 0.6473 |
| PWStabNet [42] | 0.3280 | **0.8993** | 0.8458 | 0.3049 | 0.8267 | 0.6667 |
| Wang et al. [34] | 0.2371 | 0.6745 | 0.7185 | 0.1831 | 0.6693 | 0.6599 |
| Deep3D [13] | 0.4413 | 0.8066 | 0.8563 | 0.3446 | 0.7530 | 0.7087 |
| Choi et al. [7] | 0.3692 | 0.7931 | 1.0000 | 0.4076 | 0.8260 | 1.0000 |
| L1 Stab [11] | 0.2560 | 0.8418 | 0.7454 | 0.3334 | 0.8772 | 0.7411 |
| Shi et al. [31] | 0.4286 | 0.8298 | 0.7081 | N/A | N/A | N/A |
| Ours | **0.5398** (+15.0%) | 0.8708 (−3.2%) | 0.8332 (−4.0%) | **0.4245** (+4.1%) | 0.8904 (−0.8%) | **0.8016** (+3.6%) |

Table 1. Quantitative comparison on (a) test set from Shi et al. [31] and (b) NUS dataset from Liu et al. [23]. Larger number indicates a better performance. Best entry is marked with red, and second best is marked with blue. Our method achieves significantly better stability (+15.0%) and comparable distortion (−3.2%) and cropping (−4.0%) with SOTA methods for Shi et al. dataset [31]. Using only flow stabilization network, we are able to handle general videos without sensor information (NUS dataset [23]), and achieve most stable results with least FOV cropped. Complete table with breakdowns for each video category can be found in our supplementary material.

a higher score. To this end, we improve the stability metric by considering the inter-frame motion instead of the accumulated motion in the frequency spectrum analysis. In this context, stable video should have a more constant motion signal and an unstable video will have a more varying motion signal. Therefore, we also include DC portion of FFT as the low frequency component in the improved stability metric computation. In our improved stability metric, larger stability score still indicates a more stable video. With our modified stability metric, we remove the low-frequency intentional motion and focus on the net stability; the modified stability metric effectively distinguishes the stability of the two signals in Fig. 7 (0.4323 vs. 0.0040). The modified metric shows good consistency with user study (Sec. 5.3) and aligns well with the comparisons in past works.

In Table 1(a), we compare the stability score with existing methods. In general, optical flow based methods like DUT [35] and our method are able to achieve more stable result since dense warp field can provide more flexibility compared to parametric motion models. However, our method is much more robust to disocclusion and dynamic objects in the video that exist in the test set. This makes our method achieves significantly better stability (+12.6% compared to the second best method) on average.

| Ours vs. | Ours Preferred | Ours vs. | Ours Preferred |
|---|---|---|---|
| Zhang et al. [41] | 97.9 ± 4.3% | Deep3D [13] | 89.4 ± 9.2% |
| DUT [35] | 76.6 ± 12.6% | Choi et al. [7] | 97.7 ± 4.6% |
| Yu et al. [38] | 97.7 ± 4.6% | L1 Stab [11] | 100.0 ± 0.0% |
| PWStabNet [42] | 93.2 ± 7.8% | Shi et al. [31] | 84.1 ± 11.2% |
| Wang et al. [34] | 97.7 ± 4.6% | **Average** | **92.6 ± 2.6%** |

Table 2. User study showing the percentage of our results being preferred over other methods, and the 95% confidence interval.

**Distortion and Cropping** measures the anisotropic scaling and remaining FOV after stabilization. Larger metric score indicates better performance (less distortion and larger FOV). Since our method uses dense warp field for generating stabilized frame, the distortion is comparable (−3.2%) to the best method PWStableNet [42]. Our method does not focus on preserving the field of view, so the cropping metric is slightly inferior (−4.0%) compared to the best method Yu et al. [38] in the existing methods following the stabilize and crop process. However, note that our method is able to achieve more stable results (+61.1% compared to PWStableNet [42] and +39.9% compared to Yu et al. [38]) with comparable distortion and cropping.

**Comparison on general videos** In addition to videos containing sensor information, our method also demonstrates ability to stabilize general videos. For stabilizing general videos, we replace the sensor-based pre-stabilization with a simple grid warping process solely based on image information. The grid warping is optimized so that the feature trajectories tracked by the optical flow are Gaussian smoothed. In our experiment, we track the features for 20 frames and smooth with a $\sigma = 5$ Gaussian kernel.

Table 1(b) shows the quantitative comparison on the general video NUS dataset [23]. With this weakened pre-stabilization, our flow stabilization network is able to eliminate the residual motion and achieve better stability with more remaining FOV compared to the two best-perform methods: DUT [35] (+10.2%) and Choi et al. [7] (+4.1%).

**Runtime** On a desktop with a NVIDIA A100 GPU, the average *per-frame*(1080P) processing time breakdown is as follows: sensor-based pre-stabilization 1.5ms, optical flow 10ms, mask computation 223ms, 3D CNN inference 138ms. Note that our goal is high-quality stabilization and not to pursue real-time performance. However, with proper optimization, our method can potentially be real-time as it adapts the sliding window scheme [20, 38, 39] and does not require knowledge from the entire video.

### 5.3. User Study and Ablation Study

**User study** Besides quantitative comparison, we also conduct a user study on human preference between our method and comparison methods. We recruit 27 participants, each was shown with 15 randomly selected video pairs. Each


Figure 8. The effect of optical flow masking and occlusion-aware parameter $\delta$. Insets with red rectangle shows the region of interest.

| | (a) Effect of $\lambda$ on a complete pipeline | | | | (b) Effect of pipeline modules | | |
|---|---|---|---|---|---|---|---|
| | $\lambda = 0$ | $\lambda = 0.5$ | $\lambda = 1$ | $\lambda = 3$ | Gyro only | Flow only | Complete |
| Stability | 0.4709 | 0.5131 | 0.5398 | 0.5337 | 0.4725 | 0.4800 | 0.5398 |
| Distortion | 0.8303 | 0.8704 | 0.8708 | 0.8721 | 0.8710 | 0.9123 | 0.8708 |
| Cropping | 0.8137 | 0.8307 | 0.8332 | 0.8330 | 0.8063 | 0.8764 | 0.8332 |

Table 3. Ablation study on (a) effect of smoothness loss weights $\lambda$ and (b) effect of pipeline modules.

video pair consists of a video stabilized by our method and the same video stabilized by one of the existing methods. The user is asked to consider all aspects including video stability, distortions and the cropping ratio, and decide which video is more well-stabilized. Each existing method appears approximately the same number of times in the questions presented to the human viewers. Table 2 shows the percentages of our results being preferred over existing methods. As shown, our method is over 90% more preferred when compared to most existing methods, and even surpasses the strongest baseline by a huge margin.

**Optical flow masking** In our pipeline, the optical flow masking plays an important role in avoiding distortion artifacts in stabilized video. The leftmost column in Fig. 8 shows two examples stabilized with optical flow masking, while columns on the right of the dash line show the results stabilized without optical flow masking. When large occlusion and dynamic object present, the optical flow field could not represent the actual movement in these regions, causing significant distortion in the warp stabilized result. The distorted regions are shown in the insets in Fig. 8.

**Occlusion-aware parameters** In the columns on the right of the dash line in Fig. 8, we compare stabilized result with the stabilization network trained with different occlusion-aware parameter ($\delta$ in Eqn. 13). To show the effect of the occlusion-aware parameter, we disable the flow masking in this comparison. It can be observed that larger $\delta$ makes the stabilization network more robust to occlusion in its nature, resulting in a more rigid warp field for stabilization. However, the averaged stability is negatively affected with larger $\delta$. We use $\delta = 3$ in our experiments as a trade-off between robustness and stability.

**Smoothness loss weights** In Table 3(a), we show the effect of smoothness loss weights $\lambda$ in Sec. 4.2. In general, $\lambda$ controls the similarity between the stabilization network output warp field and a global homography. Larger $\lambda$ results in less distortion, but sacrifices the stability to enforce the rigidity of warp field. In our experiment, we use $\lambda = 1$ as a trade-off between distortion and stability.

**Pipeline modules** We also perform ablation study on the two modules in our pipeline, i.e. sensor-based pre-stabilization and flow stabilization network. In Table 3(b), we compare the performance of our method under 3 scenarios: "Gyro only" means we only apply sensor-based pre-stabilization; "Flow only" means we replace sensor-based pre-stabilization with a grid warping process that smooths the motion tracked by optical flow, then apply flow stabilization network; "Complete" means we process the video with the complete pipeline. Both our modules are able to achieve superior stability independently compared to existing methods. When combined, the overall performance greatly benefits from the sensor-based stabilization's ability in large motion removal and the flow-based stabilization's ability in jitter removal.

## 6. Conclusion and Limitation

In this work, we present a novel sensor and image fusion framework for robust, high-quality video stabilization. First, we introduce a novel angular velocity domain optimization using sensor data (gyroscope and OIS) to achieve pre-stabilized videos that are artifact free. We then remove the translational motion and residual jitter with a occlusion-aware 3D convolutional neural network that takes the stabilized optical flows as input and outputs per-pixel dense warping fields. A novel masking scheme is introduced to further avoid distortion artifacts caused by erroneous optical flow. We have demonstrated that our method achieves visually pleasing and the state-of-the-art video stabilization results by comparing against alternative methods.

Our method has the following limitations. First, the pipeline requires optical flow as input. Though the existing optical flow solutions work well, this requirement introduces the risk of visual artifacts from potentially inaccurate flows. Second, the model's ability to correct parallax from 2D warp field is still limited. Ideally the stabilization can be done through novel view synthesis with a proper 3D dynamics representation. Finally, while this work focuses on high-quality video stabilization, it can be adapted to online (even real-time) stabilization with further performance optimizations. We leave exploration of these potential directions for future research.

## References


[1] Simon Baker and Iain Matthews. Lucas-kanade 20 years on: A unifying framework. *International journal of computer vision*, 56:221–255, 2004. 1

[2] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5479, 2022. 6

[3] Steven Bell, Alejandro Troccoli, and Kari Pulli. A non-linear filter for gyroscope-based video stabilization. In *ECCV*, 2014. 2

[4] Chris Buehler, Michael Bosse, and Leonard McMillan. Non-metric image-based rendering for video stabilization. In *CVPR*, 2001. 2

[5] Hung-Chang Chang, Shang-Hong Lai, and Kuang-Rong Lu. A robust real-time video stabilization algorithm. *Journal of Visual Communication and Image Representation*, 17(3):659–673, 2006. 2

[6] Yu-Ta Chen, Kuan-Wei Tseng, Yao-Chih Lee, Chun-Yu Chen, and Yi-Ping Hung. Pixstabnet: Fast multi-scale deep online video stabilization with pixel-based warping. In *ICIP*, 2021. 2

[7] Jinsoo Choi and In So Kweon. Deep iterative frame interpolation for full-frame video stabilization. *ACM TOG*, 39(1):1–9, 2020. 1, 2, 5, 6, 7

[8] Steve Eddins. Region filling and Laplace's equation, 6 2015. 4

[9] Amit Goldstein and Raanan Fattal. Video stabilization using epipolar geometry. *ACM TOG*, 31(5):1–10, 2012. 2

[10] Matthias Grundmann, Vivek Kwatra, and Irfan Essa. Auto-directed video stabilization with robust L1 optimal camera paths. In *CVPR*, 2011. 1

[11] Matthias Grundmann, Vivek Kwatra, and Irfan Essa. Auto-directed video stabilization with robust l1 optimal camera paths. In *CVPR*, 2011. 2, 5, 6, 7

[12] Alexandre Karpenko, David Jacobs, Jongmin Baek, and Marc Levoy. Digital video stabilization and rolling shutter correction using gyroscopes. *CSTR*, 2011. 2

[13] Yao-Chih Lee, Kuan-Wei Tseng, Yu-Ta Chen, Chien-Cheng Chen, Chu-Song Chen, and Yi-Ping Hung. 3d video stabilization with depth estimation by cnn-based optimization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10621–10630, 2021. 2, 5, 6, 7

[14] Chen Li, Li Song, Shuai Chen, Rong Xie, and Wenjun Zhang. Deep online video stabilization using imu sensors. *IEEE TMM*, 25:2047–2060, 2023. 2

[15] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6498–6508, 2021. 1, 2

[16] Zhengqi Li, Qianqian Wang, Forrester Cole, Richard Tucker, and Noah Snavely. Dynibar: Neural dynamic image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4273–4284, 2023. 2

[17] Chia-Kai Liang and Fuhao Shi. Fused video stabilization on the Pixel 2 and Pixel 2 XL. https://ai.googleblog.com/2017/11/fused-video-stabilization-on-pixel-2.html, 2017. 1, 2

[18] Feng Liu, Michael Gleicher, Hailin Jin, and Aseem Agarwala. Content-preserving warps for 3D video stabilization. *ACM TOG*, 28(3):44:1–44:9, 2009. 1, 2

[19] Feng Liu, Michael Gleicher, Jue Wang, Hailin Jin, and Aseem Agarwala. Subspace video stabilization. *ACM TOG*, 30(1):4:1–4:10, 2011. 1, 2

[20] Shuaicheng Liu, Ping Tan, Lu Yuan, Jian Sun, and Bing Zeng. Meshflow: Minimum latency online video stabilization. In *ECCV*, 2016. 1, 2, 7

[21] Shuaicheng Liu, Yinting Wang, Lu Yuan, Jiajun Bu, Ping Tan, and Jian Sun. Video stabilization with a depth camera. In *CVPR*, 2012. 2

[22] Shuaicheng Liu, Lu Yuan, Ping Tan, and Jian Sun. Bundled camera paths for video stabilization. *ACM TOG*, 32(4):78:1–78:10, 2013. 1, 2

[23] Shuaicheng Liu, Lu Yuan, Ping Tan, and Jian Sun. Bundled camera paths for video stabilization. *ACM TOG*, 32(4), 2013. 6, 7

[24] Shuaicheng Liu, Lu Yuan, Ping Tan, and Jian Sun. Steadyflow: Spatially smooth optical flow for video stabilization. In *CVPR*, 2014. 1, 2

[25] Yu-Lun Liu, Wei-Sheng Lai, Ming-Hsuan Yang, Yung-Yu Chuang, and Jia-Bin Huang. Hybrid neural fusion for full-frame video stabilization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2299–2308, 2021. 2

[26] Yasuyuki Matsushita, Eyal Ofek, Weina Ge, Xiaoou Tang, and Heung-Yeung Shum. Full-frame video stabilization with motion inpainting. *IEEE TPAMI*, 28(7):1150–1163, 2006. 2

[27] 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. *Communications of the ACM*, 65(1):99–106, 2021. 1

[28] Carlos Morimoto and Rama Chellappa. Evaluation of image stabilization algorithms. In *ICASSP*, 1998. 2

[29] Zhan Peng, Xinyi Ye, Weiyue Zhao, Tianqi Liu, Huiqiang Sun, Baopu Li, and Zhiguo Cao. 3d multi-frame fusion for video stabilization. In *CVPR*, 2024. 2

[30] Fuhao Shi, Sung-Fang Tsai, Youyou Wang, and Chia-Kai Liang. Steadiface: Real-time face-centric stabilization on mobile phones. In *ICIP*, 2019. 2

[31] Zhenmei Shi, Fuhao Shi, Wei-Sheng Lai, Chia-Kai Liang, and Yingyu Liang. Deep online fused video stabilization. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 1250–1258, 2022. 2, 3, 5, 6, 7

[32] Brandon M Smith, Li Zhang, Hailin Jin, and Aseem Agarwala. Light field video stabilization. In *ICCV*, 2009. 2

[33] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 402–419. Springer, 2020. 1, 3

[34] 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 TIP*, 2018. 2, 5, 6, 7

[35] Yufei Xu, Jing Zhang, Stephen J. Maybank, and Dacheng Tao. Dut: Learning video stabilization by simply watching unstable videos. *IEEE TIP*, 2022. 1, 5, 6, 7

[36] Jiyang Yu and Ravi Ramamoorthi. Selfie video stabilization. In *ECCV*, 2018. 2

[37] Jiyang Yu and Ravi Ramamoorthi. Robust video stabilization by optimization in cnn weight space. In *CVPR*, 2019. 1, 2

[38] Jiyang Yu and Ravi Ramamoorthi. Learning video stabilization using optical flow. In *CVPR*, 2020. 1, 2, 4, 5, 6, 7

[39] Jiyang Yu, Ravi Ramamoorthi, Keli Cheng, Michel Sarkis, and Ning Bi. Real-time selfie video stabilization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12036–12044, 2021. 2, 7

[40] Fang-Lue Zhang, Xian Wu, Hao-Tian Zhang, Jue Wang, and Shi-Min Hu. Robust background identification for dynamic video editing. *ACM TOG*, 35(6):1–12, 2016. 2

[41] Zhuofan Zhang, Zhen Liu, Ping Tan, Bing Zeng, and Shuaicheng Liu. Minimum latency deep online video stabilization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 23030–23039, 2023. 2, 5, 6, 7

[42] Minda Zhao and Qiang Ling. Pwstablenet: Learning pixel-wise warping maps for video stabilization. *IEEE TIP*, 2020. 2, 5, 6, 7

[43] Weiyue Zhao, Xin Li, Zhan Peng, Xianrui Luo, Xinyi Ye, Hao Lu, and Zhiguo Cao. Fast full-frame video stabilization with iterative optimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 23534–23544, 2023. 2

[44] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817*, 2018. 5

[45] Binnan Zhuang, Dongwoon Bai, and Jungwon Lee. 5D video stabilization through sensor vision fusion. In *ICIP*, 2019. 2