

# Digital Gimbal: End-to-end Deep Image Stabilization with Learnable Exposure Times

Omer Dahary<sup>1</sup>, Matan Jacoby<sup>1</sup>, and Alex M. Bronstein<sup>1</sup>

<sup>1</sup>Technion - Israel Institute of Technology  
 {omerd,matanj,bron}@cs.technion.ac.il

## Abstract

Mechanical image stabilization using actuated gimbals enables capturing long-exposure shots without suffering from blur due to camera motion. These devices, however, are often physically cumbersome and expensive, limiting their widespread use. In this work, we propose to digitally emulate a mechanically stabilized system from the input of a fast unstabilized camera. To exploit the trade-off between motion blur at long exposures and low SNR at short exposures, we train a CNN that estimates a sharp high-SNR image by aggregating a burst of noisy short-exposure frames, related by unknown motion. We further suggest learning the burst's exposure times in an end-to-end manner, thus balancing the noise and blur across the frames. We demonstrate this method's advantage over the traditional approach of deblurring a single image or denoising a fixed-exposure burst on both synthetic and real data.

## 1. Introduction

Through advances in imaging technology, optical systems have become lighter and more portable than ever. These improvements, however, are sometimes negated by the need for stabilization. Due to the natural tremor of handheld cameras, or the movement of mounted vehicles, camera motion is often unavoidable. As a result, images captured this way may suffer from motion blur, which is especially evident during long exposures or with long focal lengths. Image stabilization, the task of mitigating this phenomenon, has been mostly explored in the optical and mechanical domains. In the optical regime, stabilization emerges from using high-cost, uniquely tailored lenses or cheaper but less effective shift sensors. On the other hand, mechanical stabilization devices, such as actuated gimbals, can be attached to any camera with no need for complicated optical equipment. Such systems are highly popular but

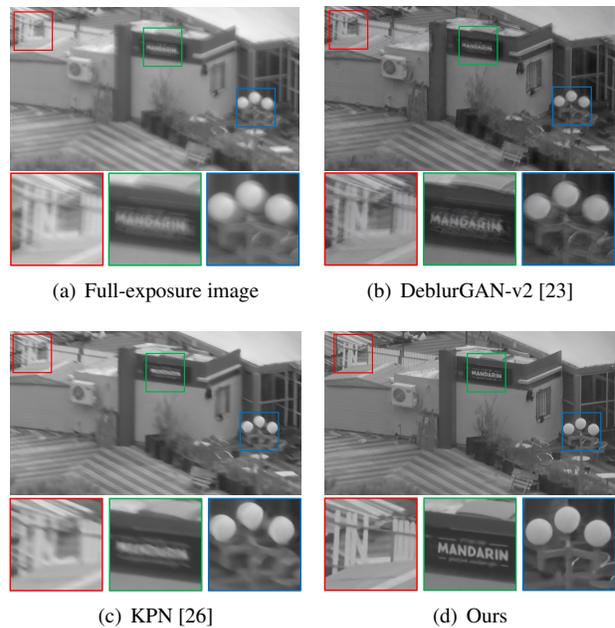


Figure 1. A qualitative evaluation of our approach on images captured by a vibrating camera with a long focal length. (See the supplement for a description of our setup.) The fully-exposed image (a) suffers from considerable motion blur, which is hard to mitigate using image deblurring methods (b). Employing the same time budget to capture a fixed-exposure burst and merging it via a burst denoising algorithm (c) results in over-smoothing and ghosting artifacts. Our approach (d) of learning a non-uniform exposure regime enables utilizing the sharpness of short-exposed frames and the high SNR of long-exposed ones to yield a superior reconstruction.

limited in their ability to compensate for blur due to scene motion or atmospheric turbulence. However, the most severe limitation of mechanical image stabilization is its cost, form factor, weight, and power requirements, which might be prohibitive in many scenarios.

In recent years, deep learning approaches have been successfully utilized for solving traditional image processing tasks. The field of burst and multi-frame imaging, in particular, has seen rising attention, with a variety of works on denoising [26] and deblurring [1, 40]. These methods exploit the information entangled in different degraded realizations of the same scene to reassemble them into a single high-quality image. Relying on this concept, together with the well-known trade-off between strong blur at long exposures and low SNR at short exposures, we propose solving the image stabilization task using a burst of images, each captured consecutively over a short period. Thus we reduce the problem to joint burst denoising and deblurring, as the two are already well-established. While most burst methods assume the data across the burst are complementary, we further suggest explicitly intervening in the acquisition pipeline and forcing diversity in noise and blur levels. We achieve this by designing an end-to-end learned system that includes both a deep burst image processing network and the camera’s exposure configuration. The latter is implemented using a novel network layer that numerically models the frame acquisition process, and can potentially be further tuned for other imaging tasks. To the best of our knowledge, this approach has never been attempted before for still image reconstruction from SNR-limited bursts in the presence of camera motion.

## 2. Related work

Burst imaging aims to compensate for non-optimal optics or imaging conditions by fusing frames captured sequentially over a short period. This ongoing research field has been excessively studied for denoising and deblurring, with most works focusing on a single kind of degradation.

Delbracio and Sapiro [6] propose an elegant scheme for weighted spectral domain averaging of uniformly-blurred bursts to produce sharp images. Wieschollek *et al.* [40] expand on this idea and introduce a CNN for predicting deconvolution filters and weights for Fourier burst accumulation. Aittala and Durand [1] suggest a permutation-invariant network for fusing unordered sets of blurry images. Mildenhall *et al.* [26] propose an adaptive kernel prediction network for jointly aligning and merging noisy frames. Sim and Kim [35] further utilize this approach for video deblurring by fusing adjacent frames and a generated residual image.

The task of image restoration from bursts with non-uniform exposure times was first explored using traditional image processing tools. Ben-Ezra and Nayar [2] introduce a hybrid imaging system that records camera motion to predict the point spread function of a blurry image. Zhang *et al.* [44] use a sparse prior to adaptively combine information from noisy and blurry inputs. Yuan *et al.* [42] employ a noisy/blurry image pair of the same scene to estimate the blur kernel and reduce ringing artifact. More recently, re-

searchers have been adopting the idea of using short- and long- exposure pairs as the input to deep learning-based restoration algorithms. These include tasks such as high dynamic range (HDR) imaging [17], deblurring [45], and low-light restoration [5]. However, these works rely on fixed and pre-selected timing regimes, with two separately-tuned data generation pipelines: one for the short-exposure frames, and another one for the long-exposure ones.

In parallel, recent works by Google Research have shown that burst imaging can mitigate smartphone cameras’ limitations by increasing their dynamic range [9], spatial resolution [41], and performance in low light conditions [24]. These improvements are achieved by an alternative image signal processing (ISP) pipeline that robustly aligns and merges frames and applies post-processing effects. Furthermore, it introduces dynamic exposure selection: either by interpolating a hand-crafted database, or by aggregating motion prediction with inertial sensor data.

## 3. Approach

We assume the scenario in which a latent scene is captured by a moving camera over a temporal interval  $[0, T]$ . With some abuse, we denote the irradiance image at time  $t$  as a latent transformation  $\tau_t$  of a latent irradiance  $E$ ,  $E_t = \tau_t(E)$ . The irradiance images are not directly observable; instead, the camera acquires a discrete set of  $n$  frames  $Y_1, \dots, Y_n$ . Each frame is captured during its integration interval  $[t_i, t_i + \Delta t_i] \subset [0, T]$ , where  $t_i$  denotes the opening time of the shutter, and  $\Delta t_i$  the frame exposure. Each frame is related to the latent image through the sensor forward model  $F$ :

$$Y_i = F(\tau_{t \in [t_i, t_i + \Delta t_i]}(E)). \quad (1)$$

Assuming the exposure times  $\Delta t_i$  are relatively low (around a few milliseconds), each frame in the burst suffers from low SNR due to imaging noise. Furthermore, we expect the frames to be slightly blurred due to the camera movement. The trade-off between these two sources of degradation is controlled by the exposure time  $\Delta t_i$ , with longer exposures corresponding to better SNR and stronger blur, and vice versa.

Our aim is to utilize the information entangled in these measurements of the scene, by combining the frames into a single sharp high-SNR estimation of the latent irradiance image,

$$\hat{E} = I(Y_1, \dots, Y_n), \quad (2)$$

where  $I$  is the reconstruction (inverse) operator.

We propose to learn the latter reconstruction operator concurrently with the user-controlled parameters of the camera, which in our case is the shutter schedule  $\{t_i, \Delta t_i\}$ .

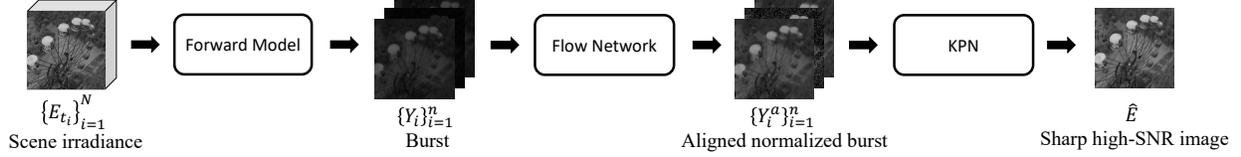


Figure 2. Our reconstruction network comprises a flow network, which aligns the burst, and a KPN, which merges the aligned frames. At train time, our forward model generates the burst from a video representing the scene’s irradiance values. This extension allows learning the frames’ optimal exposure times.

This leads to a learning problem of the form

$$\min_{\Delta, I} \mathbb{E} \ell \left( I \left( \left\{ F \left( \tau_{t \in [t_i, t_i + \Delta t_i]}(E) \right) \right\}_{i=1}^n \right), E \right). \quad (3)$$

Here  $\mathbb{E} \ell$  denotes the expectation of a loss function measuring the discrepancy between the latent  $E$  and its estimated version  $\hat{E}$ , and  $\Delta$  describes the burst parameters detailed in the sequel. In what follows, we describe the construction of differentiable forward and inverse models and their learning scheme.

## 4. Sensor forward model

This section describes our numerical forward model (denoted as  $F$  above) of the imaging process, starting with the scene irradiance and ending with a fully formed raw image. We generally follow the noise estimation of Konnik and Welsh [22], while paying particular attention to time-dependent properties and visibly dominant factors at short exposures.

### 4.1. Image formation

Our model describes a moving camera capturing short-exposure images of a static scene. For simplicity, we assume that both the camera and the light source are monochromatic in relation to the same wavelength  $\lambda$ . We further assume device-specific parameters are known, as they are customarily supplied by manufacturers [12].

**From photons to electrons.** Let  $E_t(\mathbf{x})$  be the irradiance of a pixel  $\mathbf{x}$  at time  $t$ . We can express it in terms of the photon flux

$$\gamma_t(\mathbf{x}) = \frac{\lambda A E_t(\mathbf{x})}{hc}, \quad (4)$$

where  $\lambda$  denotes the wavelength,  $A$  is the effective pixel area,  $h$  is the Planck constant, and  $c$  is the speed of light.

Each collected photon deposits its energy in the form of electric charge, generating photoelectrons inside the pixel. Their mean amount is given by

$$\bar{e}(\mathbf{x}) = \eta_\lambda \int_t^{t+\Delta t} \gamma_t(\mathbf{x}) dt, \quad (5)$$

where  $[t, t + \Delta t]$  is the time interval on which the shutter was open, and  $\eta_\lambda$  is the quantum efficiency of the pixel at wavelength  $\lambda$ .

**Noise generation.** The photons are not the sole source of charge in the pixel. Some amount of current, named the dark current, thermally deposits charge in it, even if the scene is completely dark. The mean number of electrons it generates grows linearly with the exposure time:

$$\bar{e}_0 = \frac{I_0 \Delta t}{q_e}, \quad (6)$$

where  $I_0$  is the average dark current, and  $q_e$  is the elementary charge.

The measured number of electrons fluctuates randomly, obeying Poisson statistics due their discrete nature. This phenomenon, known as shot noise, is often approximated in the image processing domain by a Gaussian distribution [5, 17, 26], which is a valid assumption in high-light regimes. However, since we are dealing with short exposures, at which shot noise dominates other components, we model photoelectron production as a realization of a Poisson variable,

$$e_q(\mathbf{x}) \sim \text{Pois}(\bar{e}(\mathbf{x}) + \bar{e}_0). \quad (7)$$

Another major source of noise, the readout noise, is independent of the exposure time. It mostly originates in thermal fluctuations in the analog-to-digital converter (ADC) and follows a zero-mean Gaussian distribution with some device-specific standard deviation,  $e_{ro}(\mathbf{x}) \sim \mathcal{N}(0, \sigma_{ro}^2)$ .

Therefore, we can express the total number of collected electrons as

$$e(\mathbf{x}) = e_q(\mathbf{x}) + e_{ro}(\mathbf{x}). \quad (8)$$

**From electrons to digital numbers.** The pixel circuits convert the collected electrons into voltage, which is then amplified and translated into digital numbers (DNs). Following the EMVA 1288 standard [12], we assume this process to be described by an almost linear curve, whose sensitivity dampens towards the full-well capacity (FWC) of the

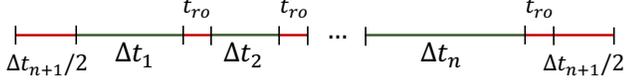


Figure 3. The shutter schedule is determined by the learned exposure times. Green and red lines represent time intervals in which the shutter is open and close, respectively. The idle time slot is optional.

pixel. We model this response function as

$$r(e) = \begin{cases} Ke, & e \leq \tau_1 \\ K \left( \tau_1 + \left( 1 - \exp\left(-\frac{e-\tau_1}{\tau_2}\right) \right) \tau_2 \right), & e \geq \tau_1 \end{cases}, \quad (9)$$

where  $K$  is the overall system gain,  $\tau_1$  is the threshold from which the curve becomes strictly concave, and  $\tau_1 + \tau_2 = \text{FWC}$ . Moreover, since DNs have discrete values, we assume the following quantization function:

$$Q(e) = \min \left\{ \max \left\{ \frac{[e]}{2^m - 1}, 0 \right\}, 1 \right\}, \quad (10)$$

with  $m$  denoting the number of bits per pixel, and  $[\cdot]$  the rounding operation. The DN value of the digital image at pixel  $\mathbf{x}$  is given by

$$Y(\mathbf{x}) = Q(r(e(\mathbf{x}) - \bar{e}_0)), \quad (11)$$

where the subtraction of the mean of the dark noise  $\bar{e}_0$  reflects the typical correction that many cameras apply.

## 4.2. Exposure parametrization

In order to make the sensor forward model amenable to learning, we assume the total time budget  $T$  and the number of frames  $n$  in the burst to be fixed, and parametrize the individual frame exposure parameters as

$$\Delta t_i = \Delta t_{\min} + \alpha_i (T - n(\Delta t_{\min} + \Delta t_{ro})). \quad (12)$$

Here  $\Delta t_{\min}$  is the minimum exposure time, and  $t_{ro}$  is the frame readout time (including any additional blank time needed between two consecutive frames). The parameters  $\alpha_i$ , representing the relative frame exposures, are, in turn, parametrized as

$$\alpha_i = \sigma(\mathbf{\Delta})_i = \frac{e^{\Delta_i}}{\sum_{j=1}^m e^{\Delta_j}}, \quad (13)$$

where  $\sigma$  denotes softmax. The vector  $\mathbf{\Delta}$ , serving as the trainable camera parameters, can be set to be either  $n$ -dimensional, in which case the burst consumes the entire available time budget, or  $(n+1)$ -dimensional, allowing not to utilize all the available time (Fig. 3).

## 4.3. Numerical approximation

The learning of the optimal burst exposure parameters  $\mathbf{\Delta}$  mandates a differentiable calculation of the forward model described by equations (5-11). While most of these computations are straightforward, special consideration should be taken in steps (5,7,10).

The forward pass of (5) calculates a temporal integral of a continuous irradiance function  $E_t(\mathbf{x})$ , scaled according to (4). At training, we approximate this integral via the trapezoidal method from a sequence of  $N \gg n$  uniformly-sampled irradiance values  $\{E_{t_i}(\mathbf{x})\}_{i=1}^N$  obtained from simulated scene flow.

We discuss the backpropagation through equations (7,10) in the supplement.

## 5. Reconstruction network

The following section describes our inverse model (denoted as  $I$  in Section 3) responsible for estimating the clean irradiance image  $E$  given the burst frames  $Y_1, \dots, Y_n$ . We base our reconstruction model on the recent work in kernel prediction networks (KPN), which were first introduced by Jia *et al.* [14], and have shown promising results in various image processing tasks, including video interpolation [28, 29], super-resolution [15], and deblurring [35], and burst denoising [26]. Its success in the latter two problems encourages its use in this work, as our goal is to merge different noisy and blurry realizations of the same scene. To do so, we predict temporally- and spatially-variant kernels and apply them to merge the captured burst into a clean and sharp image.

Since fixed-size kernels are limited in their ability to align frames, we pre-warp the burst according to a reference one, as customarily applied in video deblurring works [19, 36]. Gast and Roth [8] proposed using pre-trained flow networks as the most efficient option for the latter operation. However, this approach is impractical in our case, as the amount of noise and blur in the input frames is expected to change while the exposure times are learned. Therefore, we opted for an end-to-end training methodology, in which the flow network parameters are co-learned, not specifically for perfect alignment, but to compensate for the kernels' narrow receptive field.

A schematic representation of our network is summarized in Fig. 2.

### 5.1. Flow network

The goal of the flow network is to align the frames  $\{Y_i\}_{i=1}^n$  according to a pre-selected reference one  $Y_{i_0}$ . Since the entire burst is captured during continuous motion, we choose  $Y_{i_0}$  to be the middle frame, as we expect it to overlap the others significantly, making alignment easier. Aiming to deal with non-rigid motion and parallax, we

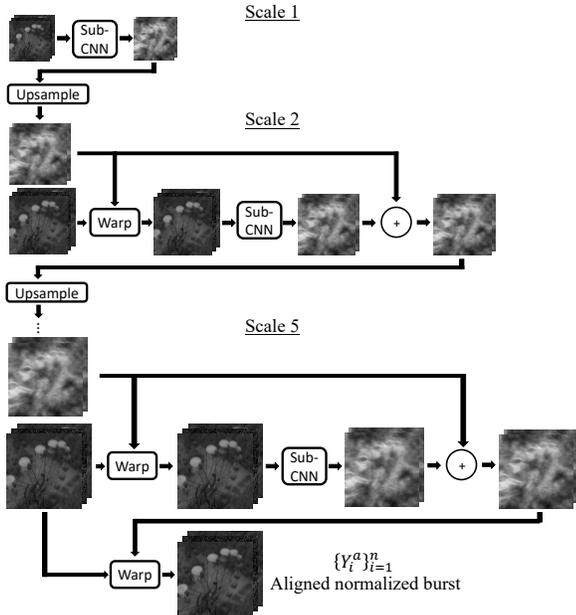


Figure 4. Our flow network obeys the architecture proposed by Kalantari and Ramamoorthi [17]. Our sub-CNN comprises four  $5 \times 5$  convolutions intertwined with ReLUs, with the number of channels being  $n, 100, 50, 25, 2(n-1)$ . In every scale, we feed the sub-CNN the appropriate pyramid level of the normalized burst, each frame in its own channel. We interpret the output as  $XY$  displacement fields for each non-reference frame.

use flow networks [7, 11, 31, 37]. However, these methods align individual images and therefore are prone to fail on degraded input. Kalantari and Ramamoorthi [17] handle this concern by jointly aligning the reference’s neighboring frames, thus complementing missing information. We follow the same architecture and simultaneously align the entire burst.

The network utilizes a hierarchical approach [17, 31, 38], which develops and refines a flow from a Gaussian pyramid of the input. At each level, from coarse to fine, we apply the same sub-CNN. At the coarsest level, we treat the sub-CNN’s output as a displacement field, which we then upsample and apply to the next level before feeding it to the sub-CNN again. Then, we sum the output flow with the upsampled one to produce the next flow in the chain and repeat. This architecture, depicted in Fig. 4, has the benefit of producing large-scale and high-precision pixel displacements.

Note that since the exposure times are not stable, the burst’s brightness varies from frame to frame and during training. Therefore, we normalize the frames according to their portion of the exposure time budget before feeding

them into the flow network:

$$Y_i^n = \frac{T}{\Delta t_i} Y_i. \quad (14)$$

## 5.2. Kernel prediction network

Fig. 5 depicts our kernel prediction architecture, which follows the noise-blind version of the network suggested by Mildenhall *et al.* [26]. It is designed as an encoder-decoder with skip connections, which predicts adaptive kernels for each input pixel. The kernels are then applied to merge all given frames into the resulting image.

The use of pixel-adaptive kernels has a few key benefits. They reduce noise by pixel averaging without risking crossing edges and can also fix non-uniform blur. The latter is especially evident in our problem setting, where long focal lengths are commonly applied [39]. Moreover, by picking the most reliable frames for each area in the final image, the kernels allow us to exploit the variability across the burst (see Fig. 6). This choice is affected by each frame’s exposure time and content but can also compensate for alignment artifacts by the flow network. Thus, similarly to Kalantari and Ramamoorthi [17], we merge the original and registered burst to allow the former to be picked over the other.

To generate the output, we first take the pixel-wise mean of the resulting frames:

$$\hat{E} = \Gamma \left( \frac{1}{2n-1} \left( \sum_{i=1}^n Y_i^n \otimes k_i^n + \sum_{i \neq i_0} Y_i^a \otimes k_i^a \right) \right). \quad (15)$$

Here  $\{Y_i^a\}_{i=1, i \neq i_0}^n$  are the aligned frames,  $\{k_i^n\}_{i=1}^n, \{k_i^a\}_{i \neq i_0}$  are the predicted location-dependent kernels, and  $\otimes$  denotes the application of the kernel. Furthermore, to improve perceptual quality, we apply the differentiable gamma correcting function  $\Gamma$  [26].

## 6. Training

### 6.1. Loss function

Given the ground-truth image  $E$ , we define our basic loss function as:

$$\ell_{\text{basic}}(\hat{E}, E) = \left\| \hat{E} - E \right\|_1 + \mu \left\| \nabla \hat{E} - \nabla E \right\|_1, \quad (16)$$

where  $\nabla$  are the combined horizontal and vertical Sobel filters, and  $\mu$  is a fixed constant. We choose the  $\ell_1$  loss as it was shown to promote sharp outputs, while the second term is applied as a regularization aiming to suppress patterned artifacts [45].

In our experiments, we found that while this loss yielded good results, the predicted kernels ignored  $\{Y_i^a\}_{i \neq i_0}$ , as the flow network fails to properly align the burst. Thus, similarly to Mildenhall *et al.* [26], we propose using an annealed

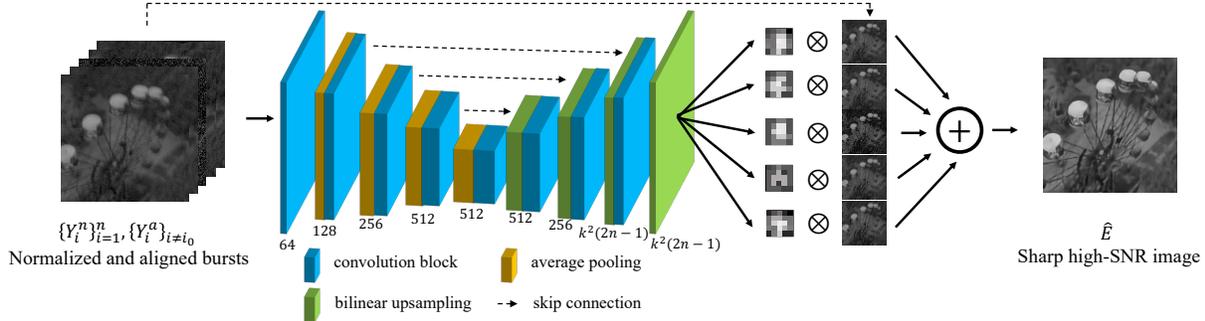


Figure 5. Our KPN follows the architecture of Mildenhall *et al.* [26]. All convolutions blocks are composed of three  $3 \times 3$  convolutions intertwined with ReLUs. We feed the network both the original normalized burst and the aligned one, each frame in its own channel, and predict  $k \times k$  kernels for each pixel in each given frame. We then apply the kernels to merge the frames into a single image.

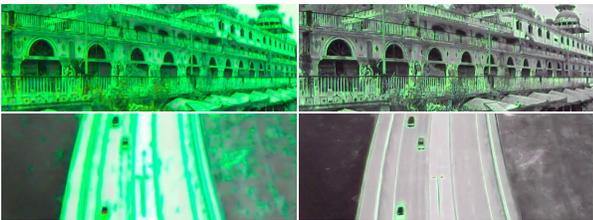


Figure 6. We reconstruct RGB images from the burst after applying the predicted kernels, for both KPN [26] (left) and our method (right). While KPN primarily uses the middle frame (corresponding to the color green), our algorithm manages to incorporate all frames. Since the middle frame’s learned exposure is lower than the others, it is mainly employed for restoring areas that suffer from more blur, such as edges and moving objects. The other two frames are averaged to denoise smoother surfaces or ones that are farther away from the camera and thus less sensitive to blur.

loss term to encourage frame alignment. Unlike the mentioned authors, we do not apply the loss to each individual frame, since it might hinder the network from learning heterogeneous timing regimes. Instead, we sum each kernel entries to produce  $1 \times 1$  kernels, and use them to merge the aligned burst:

$$\hat{E}^a = \Gamma \left( \frac{\kappa}{(2n-1)\kappa_a} \left( Y_{i_0}^n \otimes K_{i_0}^n + \sum_{i \neq 0} Y_i^a \otimes K_i^a \right) \right). \quad (17)$$

Here  $K$  denotes the sum over the corresponding indexed kernel,  $\kappa$  is the pixel-wise sum over all  $2n-1$  kernels, and  $\kappa_a$  is the pixel-wise sum over  $\{K_i^a\}_{i \neq i_0}$  and  $K_{i_0}^n$ . Thus, our overall loss is

$$\ell(\hat{E}, \hat{E}^a, E) = \ell_{\text{basic}}(\hat{E}, E) + \beta \alpha^t \ell_{\text{basic}}(\hat{E}^a, E), \quad (18)$$

where  $\beta, \alpha \in (0, 1)$  are fixed constant, and  $t$  is the current iteration number. Hence, the loss in the first phase of train-

ing will produce gradients that compensate for the kernels’ narrow receptive field by improving the flow network’s performance.

## 6.2. Synthetic dataset

While standard burst and video denoising and deblurring algorithms operate on a few neighboring frames, our method must be evaluated on burst frames matching the irregular exposure regime dictated by our training on dense irradiance maps. Therefore, in the absence of any existing dataset that allows this flexibility, we generated our own.

Since our network is proposed as an alternative to mechanical gimbals, we create our data using 720p scenery drone videos from YouTube. We divide each video into short clips of 31 frames while skipping 500 frames between each clip. This process yielded around 10,000 different clips. Since undersampling may cause discontinuous artifacts in the motion trajectory, we follow other blur simulating works [4, 27] and increase the frame rate using a CNN for frame interpolation [29]. Although deep interpolation algorithms may create unreliable individual frames, they can still handle nonlinear motion and produce naturally looking blur when averaged. We feed each clip to the CNN eight times, resulting in 241 frames. Since the middle frame is not artificially generated, we use it as the ground-truth.

At each training step, we randomly choose a  $512 \times 512$  patch from the input clip. Although it already depicts camera motion, which is imperative for producing parallax and occlusions, we further augment the set by adding a controlled amount of motion using homographies. Since drone videos are usually filmed using long focal lengths, their primary source of blur is ego-rotations [39]. We therefore apply randomly aggregated 3D rotations from frame to frame, which are computed using a pre-selected camera intrinsic matrix. Choosing the range of angles to rotate by affects the amount of blur in the final burst. However, these transformations may damage the original video quality and introduce black margins. To mitigate this degradation, we crop

Method	PSNR	SSIM
DMPHN [43]	17.99	0.6391
Analysis-synthesis network pair [18]	18.6	0.7308
DeblurGAN-v2 [23]	23.12	0.7230
KPN [26]	28.46	0.8296
Ours	<b>31.42</b>	<b>0.8948</b>

Table 1. Average PSNR and SSIM for our synthetic test set.

the patch around its center to generate a  $256 \times 256$  clip and spatially downsample it using an anti-aliasing filter. This process produces  $128 \times 128$  clips suffering from evident camera shake.

The final step is converting the clip to its corresponding irradiance values. To accomplish this, we first invert gamma correction, thus employing a linear color space, in which the pixel values are proportional to the mean number of incoming photons. Finally, we multiply the clip by a random scalar from a fixed range, which suits our target SNR.

## 7. Experiments

### 7.1. Baseline

We analyze the performance of our method against two traditional approaches: burst denoising and single image deblurring. While our algorithm incorporates non-uniformly-exposed bursts, which may suffer from both noise and blur, these alternatives often handle a single kind of degradation. Therefore, to allow a fair comparison, we apply burst denoising to a uniformly-exposed burst with the same overall budget and image deblurring to a single image captured using the budget in its entirety.

In our comparison, we use recent state-of-the-art deep learning algorithms. For image deblurring, we apply DMPHN [43], DeblurGAN-v2 [23], and the analysis-synthesis network pair [18]. To conduct a fair comparison, we re-train one representative method on million full-exposure images generated from our forward model. As the latter method does not have a publicly available training code, we re-train DeblurGAN-v2, which showed better results than DMPHN. As for burst denoising, we re-train KPN [26] on our dataset.

### 7.2. Synthetic images

We first evaluate our method on a test set of 276 clips obtained using the same procedure as our training set, although originating from different videos. We present a quantitative comparison to our baseline in Table 1. Our model outperforms traditional methods, obtaining an improvement of almost 3dB over KPN.

Fig. 7 presents an example result. As we can see, all three single image deblurring algorithms fail to deblur the

image: DMPHN’s output is distorted, while the other two exhibit ghosting artifacts. This is expected, as the full-exposure image demonstrates an evident amount of blur, which is hard to mitigate without any complementary information. Furthermore, this image also suffers from noise, raising the task difficulty even more.

On the other hand, multi-frame methods show better performance. KPN manages to clean the fixed-exposure bursts’ noise and moderately improve the resulting images’ sharpness. Nevertheless, due to the lack of frame diversity, it fails in capturing the fine details of the scene. Our approach of learning the burst exposures is the best alternative, producing a clean and sharp output.

### 7.3. Real images

We further evaluate our method on real images acquired using two different cameras at two different scenes: an indoor one with controlled light conditions and an outdoor one. (See the supplement for a full description of our setup.) Fig. 1 and 8 present a qualitative comparison, showing that the methods’ perceptual quality appears to be generally consistent with the synthetic assessment, with our learned-exposure model significantly outperforming other approaches. However, this time, the analysis-synthesis network pair achieves the best results among the single image deblurring methods, despite not being re-trained for the task, while the re-trained DeblurGAN-v2 yields poor results. This deviation can be explained by the training SNR not precisely matching the de-facto one. Nevertheless, it proves our method’s advantage even when the training conditions are not met.

Finally, we note that as the camera we used for the indoor experiment suffers from harsh fixed-pattern noise at low exposures, traces of such patterns can be found in the final output. However, this phenomenon was not replicated by the other camera, and we expect that incorporating non-uniformities in the forward model should mitigate it. We leave this revision for future work.

### 7.4. Impact of noise and blur levels

The main factors that determine the trade-off between deblurring and denoising are illumination and camera motion. Thus, they have a substantial effect on our method’s learned exposures and reconstruction quality. Fig. 9 explores these factors’ impact by displaying the method’s performance on the same synthetic scene, altered with varying noise and blur levels. To achieve optimal results, we re-train the model for each working point on suitable ranges of SNR and blur while keeping the rest of the hyper-parameters fixed, including the predicted kernel size, which we set to  $5 \times 5$ . (We list the full configuration in the supplement.)

Interestingly, the middle frame’s exposure, which acts as the reference frame in the flow network, corresponds

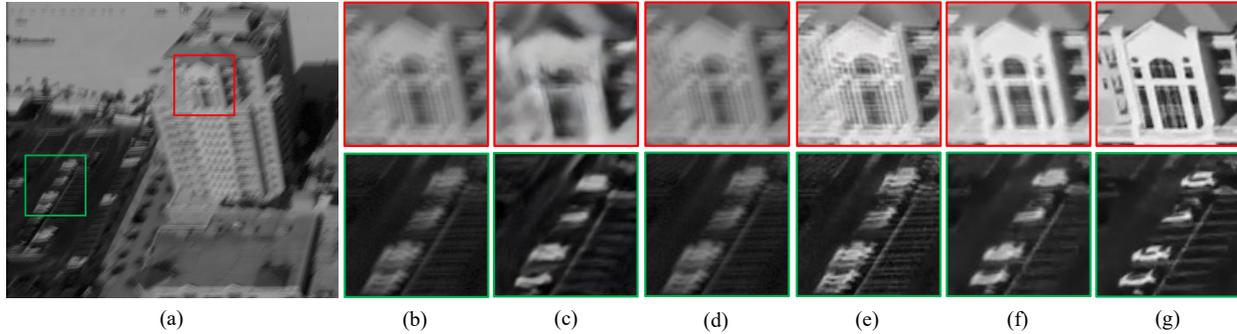


Figure 7. Example result from our synthetic test set. (a,b) Full-exposure image; Reconstruction using: (c) DMPHN [43]; (d) Analysis-synthesis network pair [18]; (e) DeblurGAN-v2 [23]; (f) KPN [26]; Our approach (g). We showcase more results in the supplement.

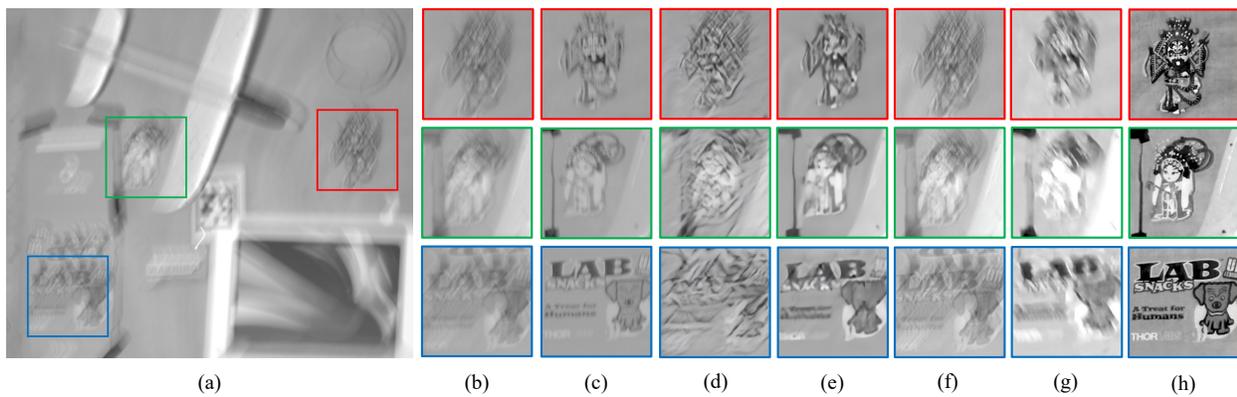


Figure 8. Example result on real images. (a,b) Full-exposure image; (c) Exemplary burst frame; Reconstruction using: (d) DMPHN [43]; (e) Analysis-synthesis network pair [18]; (f) DeblurGAN-v2 [23]; (g) KPN [26]; Our approach (h).

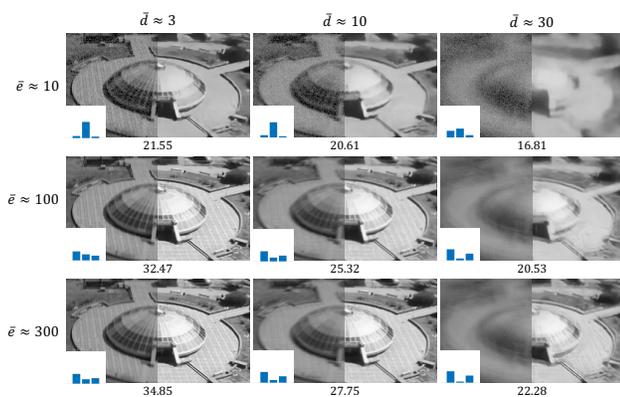


Figure 9. We compare our method’s performance for different noise and blur conditions, displaying the full-exposure image (left), the network’s output (right), its PSNR (under each image), and the learned exposures (bottom-left). Here  $\bar{d}$  denotes the mean blur kernel diameter in pixels, and  $\bar{e}$  is the mean number of photoelectrons, both estimated at full-exposure.

directly to the mentioned conditions, with it increasing as the SNR decreases and decreasing as camera motion increases, allowing the frame to be as sharp and clean as possible. The model’s performance also varies accordingly. It yields visually-pleasing results, even when the blur kernel exceeds the boundaries of the predicted one. However, like the floor’s checkered pattern, some details disappear in severe cases of noise and blur.

## 8. Conclusions

In this work, we presented a new approach for image stabilization via joint burst denoising and deblurring for fast unstabilized cameras. We further proposed an end-to-end learning scheme for optimizing the camera’s exposure regime along with a reconstruction model, made possible via a novel differentiable layer simulating the camera sensor. This method’s key benefit is its ability to exploit the trade-off between high SNR and strong blur at long exposure, and vice versa. Synthetic and real results suggest that this approach significantly improves current deep state-of-the-art methods, both perceptually and quantitatively.

## **9. Acknowledgement**

This research was kindly supported by the Smart Imaging Consortium that has been financed by MAGNET program of the Israel Innovation Authority (IIA).

# Supplemental Material

## A. Backpropagation through the sensor forward model

### A.1. Noise generation

Direct differentiation of (7) is impossible, as it involves sampling from a parametrized distribution of a discrete random variable. Therefore, we resort to estimating the gradients of the expectation over the loss. This is commonly achieved using one of two methods: the score function [10] or reparametrization [21]. Since score methods deviate from the orthodox backpropagation procedure [34], we opted for the latter.

Recently, Joo *et al.* [16] introduced the Generalized Gumbel-Softmax (GenGS) reparametrization, which can approximate any discrete, non-negative, and finite-mean random variable. They achieve this by truncating its support to a finite number of bins and relaxing the resulting categorical distribution into a continuous form using the Gumbel-Softmax reparametrization [13]. Nevertheless, this method has a few drawbacks which need to be addressed.

First, it requires setting two hyperparameters: the temperature  $\tau_{\text{GenGS}}$ , which controls the smoothness of the resulting distribution, and the number of bins in its categorical support  $n_{\text{GenGS}}$ . As  $\tau_{\text{GenGS}}$  approaches zero, sampling becomes increasingly discrete, while the gradient variance grows. Therefore, we apply the approach of Jang *et al.* [13] by starting with a high temperature and annealing it towards zero along training.

Choosing  $n_{\text{GenGS}}$  is less trivial. As the support of any Poisson distribution is infinite, increasing this hyperparameter will result in a better categorical approximation. However, since the latter are represented using one-hot vectors, memory complexity will linearly increase as well. This is especially problematic in our case, where we simultaneously sample from a different Poisson distribution for each pixel. To solve this issue, we rely on a corollary of the central limit theorem, which indicates that for high mean rate of arrivals (*e.g.* more than 1000 [25]), a Poisson distribution may be approximated by a Gaussian one. Therefore, we use GenGS for low photon counts, and a Gaussian distribution for larger ones. This allows setting  $n_{\text{GenGS}}$  to a relatively low value.

We observed that the above procedure increases performance by around 0.1dB for low to moderate SNR levels, compared to a simple Gaussian approximation.

### A.2. Quantization

Lastly, we note that the derivative of the rounding operator in (10) is a.e. zero. Upon first inspection, this disallows backpropagation through the quantization step in (11). However, if we consider the noisy additive perturbations to the quantizer input in (8), we may treat this process as stochastic and use the derivative of the expectation of the quantized value, which is smooth. We approximate this quantity by the identity straight through estimator [3].

## B. Technical details

### B.1. Implementation details

We implement our model using PyTorch [30], Kornia [33] and PyTorch3D [32], and train it for one million iterations on 4 NVIDIA GeForce RTX 2080 Ti GPUs. Since we work with 241 frames for each training example, we use NVIDIA DALI<sup>1</sup> to accelerate loading times and upload batches of 2 clips directly to each GPU. We optimize using the ADAM solver [20] with a learning rate of  $10^{-4}$ . Training takes roughly 1-2 days.

We produce bursts comprised of  $n = 3$  frames. The forward model was configured with  $\tau_1$  being 90% of the full-well capacity of the respective camera. GenGS was applied for less than 1000 incoming electrons with  $n_{\text{GenGS}} = 1200$ . We set  $\tau_{\text{GenGS}}$  to  $e^{-10^{-5}t}$ , where  $t$  is the iteration number, until a minimum of 0.1 is reached. The predicted kernel size is  $5 \times 5$ . Our loss hyperparameters are  $\mu = 1, \alpha = 0.9999886, \beta = 100$ . For a 10-bit 720p burst, evaluation takes approximately 0.7 seconds on a single GeForce RTX 2080 Ti GPU while requiring 7110MB of memory.

### B.2. Experiment setup

We list the selected camera and exposure configuration of each conducted experiment in Table 2.

<sup>1</sup><https://github.com/NVIDIA/DALI>

Experiment	Camera	$T$	$\Delta t_{\min}$	$\Delta t_{\text{ro}}$	Added blank slot ( $\Delta t_4$ )	Learned exposures
Synthetic	KAYA Instruments JetCam19M	3ms	0 $\mu$ s	500 $\mu$ s	✓	872, 234, 583 $\mu$ s
Indoor	KAYA Instruments JetCam19M	5ms	500 $\mu$ s	400 $\mu$ s	✗	1228, 503, 828 $\mu$ s
Outdoor	FLIR BFS-U3-16S2M	5ms	4 $\mu$ s	400 $\mu$ s	✗	1203, 347, 801 $\mu$ s

Table 2. Experiment configurations.



Figure 10. Outdoor experiment setup. The acquired scene is marked in red.

**Indoor experiment.** To enable different vibration modes with multiple degrees of freedom during acquisition, we mounted a Turnigy MultiStar 570KV drone motor with electronic speed control asymmetrically on top of the camera. We set the camera indoors, five meters in front of a wall depicting several printed signs. We illuminated it by a non-flickering 3000K led measured at around 50-80lx from the camera’s viewpoint. We used a 50mm fixed focal lens with an f-number of 1.8. To mitigate fixed-pattern noise (FPN), we applied dark frame subtraction and flat-field correction as provided by the camera’s ISP. Since we manually captured the burst, the blank interval between consecutive frames was not fixed in practice.

**Outdoor experiment.** We chose FLIR Blackfly S USB 3.1 as a low-cost camera with a high frame rate and a flexible interface. The camera offers 226 FPS at a resolution of  $1440 \times 1080$  pixels and a configurable array of up to 8 exposures, allowing the subsequent capture of the desired learned-exposure burst, corresponding fixed-exposure burst, and full-exposure image for comparison. We chose a lens with a focal length of 50mm and set the f-number to 22.

Missing component	PSNR
Exposure learning	28.72
Flow network	30.62
Exposure normalization	30.96
Annealed loss term	31.18
None	<b>31.42</b>

Table 3. Average PSNR of ablated models.

We set the camera on a tripod and attached the same motor from the previous experiment underneath it to create vibrations, as depicted in Fig. 10. We supplied the motor with a voltage of 13.8V, resulting in approximately 7850 RPM. Finally, we increased the vibration amplitude by unbalancing the rotor blades.

## C. Additional results

### C.1. Additional synthetic results

We include more results on our synthetic test set in Fig. 11.

### C.2. Ablation study

We conduct an ablation study to assess the contribution of each network module to our proposed pipeline. Table 3 presents the average PSNR obtained on our synthetic test set by different ablated models. As we can see, the most impactful component is the forward model, which enables learning optimal exposures and yields a significant performance gain of 2.7dB. This improvement is further demonstrated by comparing our model’s uniform-exposure output to the learned-exposure one in Fig. 11. Moreover, pre-aligning the frames via the flow network improves results by 0.8dB. It is also evident that exposure normalization (14) and our annealed loss term (17) are critical to the proper convergence of the learning process.

Careful examination of the flow network’s performance further reveals the importance of using learned exposures besides forcing frame diversity. Fig. 12 presents the reference frame and an aligned neighboring frame in both the uniform- and learned-exposure regime. As we can see, while the reference frame in the uniform case suffers from noticeable blur, its adaptive counterpart is much sharper due to its shorter exposure. This difference gives the latter model an advantage over the other, which depicts severe

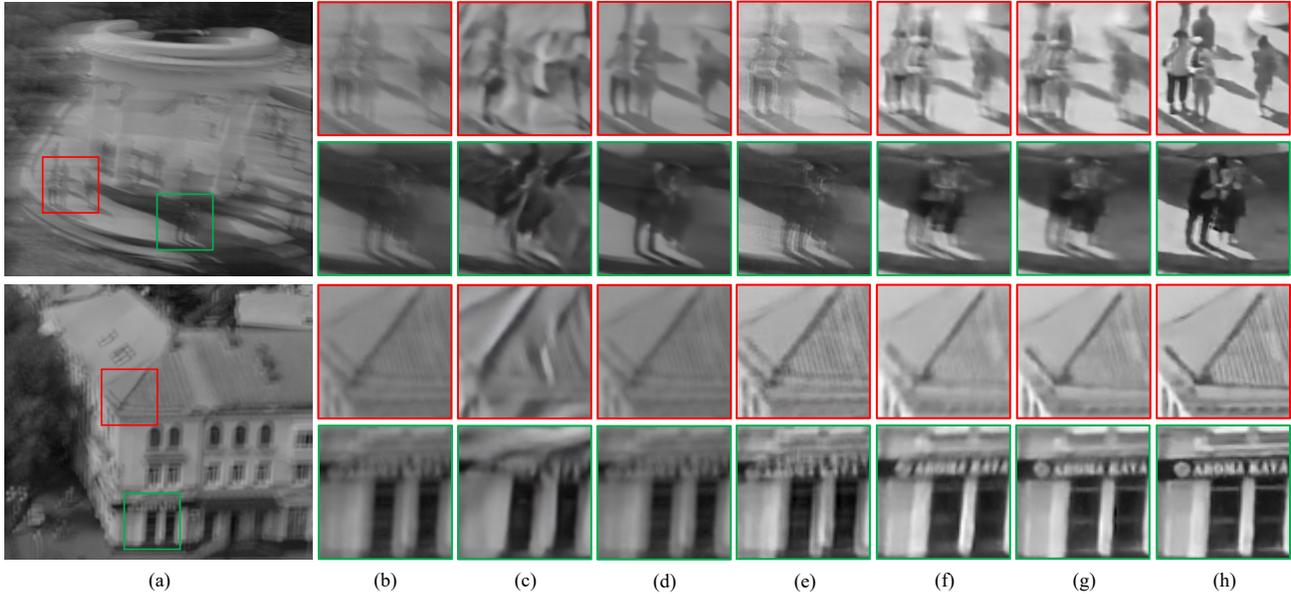


Figure 11. Example results from our synthetic test set. (a,b) Full-exposure image; Reconstruction using: (c) DMPHN [43]; (d) Analysis-synthesis networks pair [18]; (e) DeblurGAN-v2 [23]; (f) KPN [26]; Our approach using (g) fixed uniform, and (h) learned exposures.

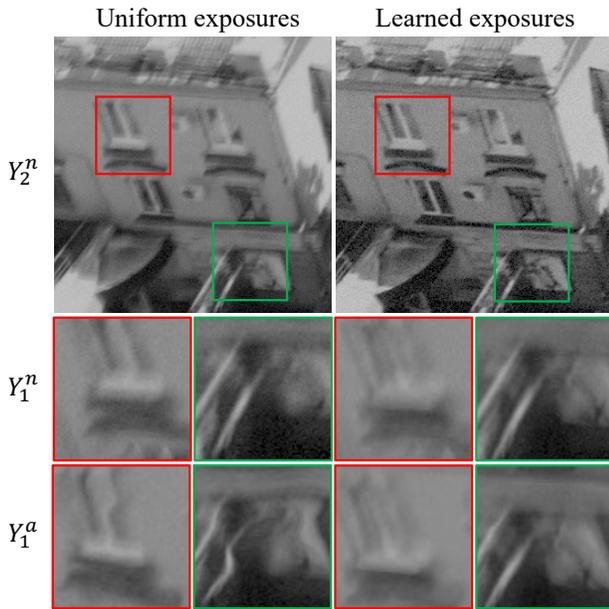


Figure 12. We compare the flow network’s performance when using uniform and learned exposures. Since the uniform burst has a blurry reference frame, alignment results in evident artifacts. On the other hand, learning the burst’s exposures enables proper alignment.

alignment artifacts. This observation demonstrates that exposure learning is not only beneficial for our end-goal reconstruction task but can also contribute to the performance of intermediate layers.

## References

- [1] Miika Aittala and Frédo Durand. Burst image deblurring using permutation invariant convolutional neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 731–747, 2018.
- [2] Moshe Ben-Ezra and Shree K Nayar. Motion deblurring using hybrid imaging. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 1, pages I–I. IEEE, 2003.
- [3] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.
- [4] Tim Brooks and Jonathan T Barron. Learning to synthesize motion blur. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6840–6848, 2019.
- [5] Meng Chang, Huajun Feng, Zhihai Xu, and Qi Li. Low-light image restoration with short-and long-exposure raw pairs. *arXiv preprint arXiv:2007.00199*, 2020.
- [6] Mauricio Delbracio and Guillermo Sapiro. Burst deblurring: Removing camera shake through fourier burst accumulation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2385–2393, 2015.
- [7] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766, 2015.
- [8] Jochen Gast and Stefan Roth. Deep video deblurring: The devil is in the details. In *Proceedings of the IEEE Inter-*

- national Conference on Computer Vision Workshops*, pages 0–0, 2019.
- [9] Samuel W Hasinoff, Dillon Sharlet, Ryan Geiss, Andrew Adams, Jonathan T Barron, Florian Kainz, Jiawen Chen, and Marc Levoy. Burst photography for high dynamic range and low-light imaging on mobile cameras. *ACM Transactions on Graphics (TOG)*, 35(6):1–12, 2016.
- [10] Shane G Henderson and Barry L Nelson. *Handbooks in operations research and management science: simulation*. Elsevier, 2006.
- [11] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2462–2470, 2017.
- [12] Bernd Jähne. Emva 1288 standard for machine vision: Objective specification of vital camera data. *Optik & Photonik*, 5(1):53–54, 2010.
- [13] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.
- [14] Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc V Gool. Dynamic filter networks. In *Advances in neural information processing systems*, pages 667–675, 2016.
- [15] Younghyun Jo, Seoung Wug Oh, Jaeyeon Kang, and Seon Joo Kim. Deep video super-resolution network using dynamic upsampling filters without explicit motion compensation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3224–3232, 2018.
- [16] Weonyoung Joo, Dongjun Kim, Seungjae Shin, and Il-Chul Moon. Generalized gumbel-softmax gradient estimator for various discrete random variables. *arXiv preprint arXiv:2003.01847*, 2020.
- [17] Nima Khademi Kalantari and Ravi Ramamoorthi. Deep hdr video from sequences with alternating exposures. In *Computer Graphics Forum*, volume 38, pages 193–205. Wiley Online Library, 2019.
- [18] Adam Kaufman and Raanan Fattal. Deblurring using analysis-synthesis networks pair. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5811–5820, 2020.
- [19] Tae Hyun Kim, Mehdi SM Sajjadi, Michael Hirsch, and Bernhard Schölkopf. Spatio-temporal transformer network for video restoration. In *European Conference on Computer Vision*, pages 111–127. Springer, 2018.
- [20] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [21] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- [22] Mikhail Konnik and James Welsh. High-level numerical simulations of noise in ccd and cmos photosensors: review and tutorial. *arXiv preprint arXiv:1412.4031*, 2014.
- [23] Orest Kupyn, Tetiana Martyniuk, Junru Wu, and Zhangyang Wang. Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8878–8887, 2019.
- [24] Orly Liba, Kiran Murthy, Yun-Ta Tsai, Tim Brooks, Tianfan Xue, Nikhil Karnad, Qiurui He, Jonathan T Barron, Dillon Sharlet, Ryan Geiss, et al. Handheld mobile photography in very low light. *ACM Transactions on Graphics (TOG)*, 38(6):1–16, 2019.
- [25] William M Mendenhall and Terry L Sincich. *Statistics for Engineering and the Sciences*. CRC Press, 2016.
- [26] Ben Mildenhall, Jonathan T Barron, Jiawen Chen, Dillon Sharlet, Ren Ng, and Robert Carroll. Burst denoising with kernel prediction networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2502–2510, 2018.
- [27] Seungjun Nah, Sungyong Baik, Seokil Hong, Gyeongsik Moon, Sanghyun Son, Radu Timofte, and Kyoung Mu Lee. Ntire 2019 challenge on video deblurring and super-resolution: Dataset and study. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.
- [28] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive convolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 670–679, 2017.
- [29] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive separable convolution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 261–270, 2017.
- [30] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037, 2019.
- [31] Anurag Ranjan and Michael J Black. Optical flow estimation using a spatial pyramid network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4161–4170, 2017.
- [32] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv preprint arXiv:2007.08501*, 2020.
- [33] Edgar Riba, Dmytro Mishkin, Daniel Ponsa, Ethan Rublee, and Gary Bradski. Kornia: an open source differentiable computer vision library for pytorch. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 3674–3683, 2020.
- [34] John Schulman, Nicolas Heess, Theophane Weber, and Pieter Abbeel. Gradient estimation using stochastic computation graphs. In *Advances in Neural Information Processing Systems*, pages 3528–3536, 2015.
- [35] Hyeonjun Sim and Munchurl Kim. A deep motion deblurring network based on per-pixel adaptive kernels with residual down-up and up-down modules. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.
- [36] Shuo Chen Su, Mauricio Delbracio, Jue Wang, Guillermo Sapiro, Wolfgang Heidrich, and Oliver Wang. Deep video deblurring for hand-held cameras. In *Proceedings of the*

*IEEE Conference on Computer Vision and Pattern Recognition*, pages 1279–1288, 2017.

- [37] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8934–8943, 2018.
- [38] Ting-Chun Wang, Jun-Yan Zhu, Nima Khademi Kalantari, Alexei A Efros, and Ravi Ramamoorthi. Light field video capture using a learning-based hybrid imaging system. *ACM Transactions on Graphics (TOG)*, 36(4):1–13, 2017.
- [39] Oliver Whyte, Josef Sivic, Andrew Zisserman, and Jean Ponce. Non-uniform deblurring for shaken images. *International journal of computer vision*, 98(2):168–186, 2012.
- [40] Patrick Wieschollek, Bernhard Schölkopf, Hendrik PA Lensch, and Michael Hirsch. End-to-end learning for image burst deblurring. In *asian conference on computer vision*, pages 35–51. Springer, 2016.
- [41] Bartłomiej Wronski, Ignacio Garcia-Dorado, Manfred Ernst, Damien Kelly, Michael Krainin, Chia-Kai Liang, Marc Levoy, and Peyman Milanfar. Handheld multi-frame super-resolution. *ACM Transactions on Graphics (TOG)*, 38(4):1–18, 2019.
- [42] Lu Yuan, Jian Sun, Long Quan, and Heung-Yeung Shum. Image deblurring with blurred/noisy image pairs. In *ACM SIGGRAPH 2007 papers*, pages 1–es. 2007.
- [43] Hongguang Zhang, Yuchao Dai, Hongdong Li, and Piotr Koniusz. Deep stacked hierarchical multi-patch network for image deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5978–5986, 2019.
- [44] Haichao Zhang, David Wipf, and Yanning Zhang. Multi-image blind deblurring using a coupled adaptive sparse prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1051–1058, 2013.
- [45] Shuang Zhang, Ada Zhen, and Robert L Stevenson. Deep motion blur removal using noisy/blurry image pairs. *arXiv preprint arXiv:1911.08541*, 2019.