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

Omer Dahary[1], Matan Jacoby[1], and Alex M. Bronstein[1]

[1]Technion - Israel Institute of Technology
{omerd,matanj,bron}@cs.technion.ac.il

1. Backpropagation through the sensor forward model

1.1. 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 [2] or reparametrization [7]. Since score methods deviate from the orthodox backpropagation procedure [14], we opted for the latter.

Recently, Joo *et al*. [4] 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 [3]. Nevertheless, this method has a few drawbacks which need to be addressed.

First, it requires setting two hyperparameters: the temperature τ_{GenGS}, which controls the smoothness of the resulting distribution, and the number of bins in its categorical support n_{GenGS}. As τ_{GenGS} approaches zero, sampling becomes increasingly discrete, while the gradient variance grows. Therefore, we apply the approach of Jang *et al*. [3] by starting with a high temperature and annealing it towards zero along training.

Choosing n_{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 [9]), 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_{GenGS} to a relatively low value.

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

1.2. 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 [1].

2. Technical details

2.1. Implementation details

We implement our model using PyTorch [11], Kornia [13] and PyTorch3D [12], 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[1] to accelerate loading times and upload batches of 2 clips directly to each GPU. We optimize using the ADAM solver [6] 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 τ_1 being 90% of the full-well capacity of the respective camera. GenGS was applied

[1]https://github.com/NVIDIA/DALI

Figure 1. Outdoor experiment setup. The acquired scene is marked in red.

for less than 1000 incoming electrons with $n_{\text{GenGS}} = 1200$. We set τ_{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 5×5. 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.

2.2. Experiment setup

We list the selected camera and exposure configuration of each conducted experiment in Table 1.

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 1440×1080 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.

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. 1. 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.

3. Additional results

3.1. Additional synthetic results

We include more results on our synthetic test set in Fig. 2.

3.2. Ablation study

We conduct an ablation study to assess the contribution of each network module to our proposed pipeline. Table 2 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. 2. 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.

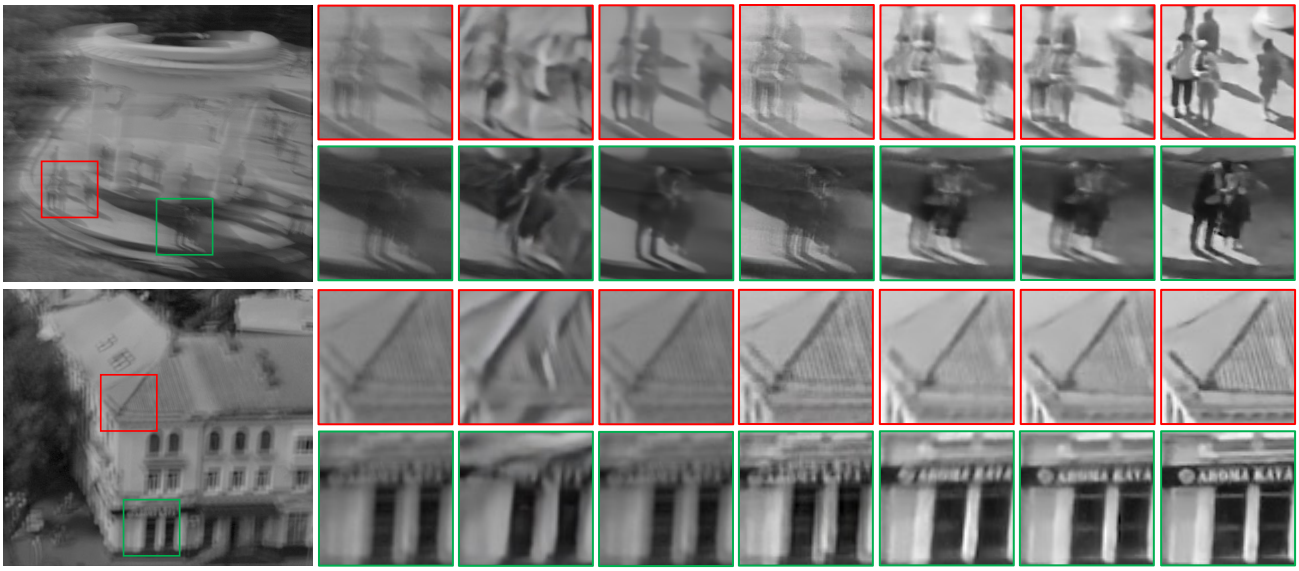
Careful examination of the flow network's performance further reveals the importance of using learned exposures besides forcing frame diversity. Fig. 3 presents the reference frame and an aligned neighboring frame in both the uniform- and learned-exposure regime. As we can see, while the reference frame 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 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] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Experiment	Camera	T	$\Delta t_{\min}$	Δt_{ro}	Added blank slot (Δt_4)	Learned exposures
Synthetic	KAYA Instruments JetCam19M	3ms	0µs	500µs	✓	872, 234, 583µs
Indoor	KAYA Instruments JetCam19M	5ms	500µs	400µs	✗	1228, 503, 828µs
Outdoor	FLIR BFS-U3-16S2M	5ms	4µs	400µs	✗	1203, 347, 801µs

Table 1. Experiment configurations.



(a) (b) (c) (d) (e) (f) (g) (h)

Figure 2. Example results from our synthetic test set. (a,b) Full-exposure image; Reconstruction using: (c) DMPHN [15]; (d) Analysis-synthesis networks pair [5]; (e) DeblurGAN-v2 [8]; (f) KPN [10]; Our approach using (g) fixed uniform, and (h) learned exposures.

Missing component	PSNR
Exposure learning	28.72
Flow network	30.62
Exposure normalization	30.96
Annealed loss term	31.18
None	**31.42**

Table 2. Average PSNR of ablated models.

[2] Shane G Henderson and Barry L Nelson. *Handbooks in operations research and management science: simulation*. Elsevier, 2006.

[3] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[4] 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.

[5] 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.

[6] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[7] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[8] 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.

[9] William M Mendenhall and Terry L Sincich. *Statistics for Engineering and the Sciences*. CRC Press, 2016.

[10] 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.

[11] 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.

[12] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia

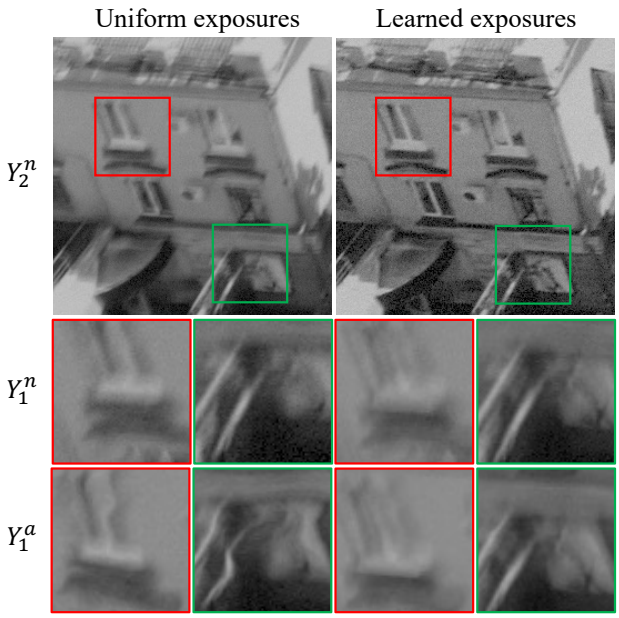


Figure 3. We compare the flow network's performance when using uniform and learned exposures. Since the uniform burst has a blurry reference frame, alignment results in evident artifacts. On the other hand, learning the burst's exposures enables proper alignment.

Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv preprint arXiv:2007.08501*, 2020.

[13] 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.

[14] 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.

[15] 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.