Deep Equilibrium Optical Flow Estimation: Appendix

Shaojie Bai[1*] Zhengyang Geng[2*] Yash Savani[1] J. Zico Kolter[1,3]
[1]Carnegie Mellon University [2]Peking University [3]Bosch Center for AI
{shaojieb,ysavani,zkolter}@cs.cmu.edu zhengyanggeng@gmail.com

Algorithm 1 DEQ flow (PyTorch-style). Note that we reuse the fixed point and perform fixed-point correction.

```
# solver: fixed-point solver, e.g., Broyden [2].
# func: layer f_θ that defines dynamic system
# dist: loss function for fixed point correction
# x: input information x_t = (q_t, C_t) of frame t,
    whose flow estimation we want to compute
# z: fixed-point flow estimation z*_t
# freq: frequency of correction.
# gamma: coefficient of correction.
# prev_z: z*_{t-1} of the last frame (if exists)
# training: bool indicating training/inference.

# Forward pass (w/ backward pass by autodiff)
def forward(x, f, gamma, freq=1,
   training=True, prev_z=None):
   with torch.no_grad():
      # Fixed-Point Reuse
      z, z_m = solver(func, x, freq, z0=prev_z)

   if training:
      loss = dist(f, func(z, x))
      # Fixed Point Correction w/ 1-step gradient
      for i in range(freq):
         z_mi = func(z_m[i], x)
         loss += gamma[i] * dist(f, z_mi)
      return z, loss
   return z
```

A. Pseudo Code

We provide a PyTorch-style [1] pseudo-code for the DEQ flow in Alg. 1. Besides fixed-point reuse and an inexact (one-step) gradient as shown previously, we also include the fixed-point correction loss (applied with frequency `freq`). In practice, we set `freq`$=1$, and use either Broyden's method [2] or Anderson acceleration [3] for `solver`. This sparse fixed-point correction scheme encourages stable training dynamics, which we analyze further in Fig. 1.

B. Experiment Settings

In this section, we present the detailed experiment settings for training and inference with the DEQ flow estimators. The code will be made publicly available upon acceptance.

B.1. Model Design

As mentioned previously, a deep equilibrium (DEQ) flow estimator subsumes a wide variety of model designs, and can be integrated with the latest, cutting-edge update operators. We show the integration of two of the most prominent designs that have achieved state-of-the-art optical flow results below, while noting in general that other alternatives are also possible.

DEQ flow by RAFT. Without any modification to the original design of RAFT [4], we can instantiate a DEQ-RAFT by defining the equilibrium system as follows,

$$\begin{aligned}\mathbf{x} &= \text{Conv2d}\left([\mathbf{q},\, \mathbf{f}^*, \mathcal{C}(\mathbf{f}^* + \mathbf{c}^0)]\right)\\ \mathbf{h}^* &= \text{ConvGRU}\left(\mathbf{h}^*, [\mathbf{x},\, \mathbf{q}]\right)\\ \mathbf{f}^* &= \mathbf{f}^* + \text{Conv2d}\left(\mathbf{h}^*\right),\end{aligned} \tag{1}$$

where $\mathcal{C}(\mathbf{f}^* + \mathbf{c}^0)$ stands for the correlation lookup as in RAFT [4], Conv2d stands for 2D convolutional layers with ReLU activations, and ConvGRU represents a GRU-style gated activation following convolutions, respectively. We refer the readers to Teed and Deng [4] and the code base[1] for more details.

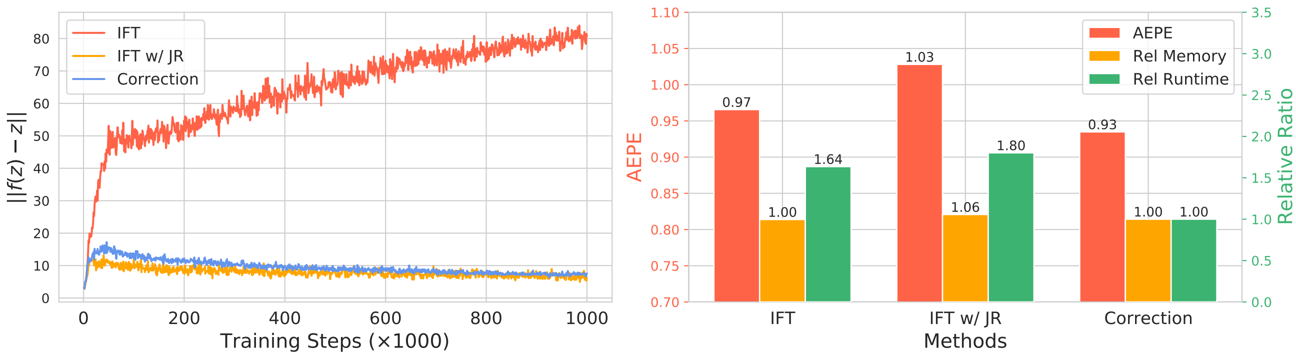
DEQ flow by GMA. More recently, Jiang et al. [5] show that we can improve on the formulation of RAFT above by adding an attention module to better model the occlusion scenarios in video frames. Specifically, we also provide an instantiation of such Global Motion Aggregation (GMA) update operator [5] in the context of DEQ flows, where we solve for the equilibrium $\mathbf{z}^* = (\mathbf{h}^*, \mathbf{f}^*)$ that satisfies

$$\begin{aligned}\mathbf{x} &= \text{Conv2d}\left([\mathbf{q},\, \mathbf{f}^*, \mathcal{C}(\mathbf{f}^* + \mathbf{c}^0)]\right)\\ \hat{\mathbf{x}} &= \text{Attention}\left(\mathbf{q},\, \mathbf{q},\, \mathbf{x}\right)\\ \mathbf{h}^* &= \text{ConvGRU}\left(\mathbf{h}^*, [\hat{\mathbf{x}},\, \mathbf{x},\, \mathbf{q}]\right)\\ \mathbf{f}^* &= \mathbf{f}^* + \text{Conv2d}\left(\mathbf{h}^*\right)\end{aligned} \tag{2}$$

where Attention is a self-attention-based operation module, see [5, 6].

*Equal contribution.

[1] https://github.com/princeton-vl/RAFT



(a) Training stability (b) Training cost and generalization performance

Figure 1. **Comparison of canonical IFT, Jacobian Regularization and Fixed-Point Correction.** Given a limited forward solver budget, the fixed-point correction protocol successfully stabilizes training and shows accelerated fixed-point convergence with visible performance improvements over Jacobian Regularization [7] for DEQ.

Model Hyperparameters For the base models, DEQ-RAFT-B and DEQ-GMA-B, we employ the *exact same architecture and hyperparameter choice* for the equilibrium module f_θ as originally used by RAFT [4] and GMA [5]. We merely replace the recurrent (and BPTT-based) formulation with a fixed-point system-based one (and the backward pass with IFT and inexact gradients). For DEQ-RAFT-D, we reparameterize all the linear projections of the ConvGRU layer in Eq. (1) by convolutional layers using kernel sizes of 1×5, 5×1, and 3×3 with a dilation rate of 4, and then duplicate the ConvGRU layer of f_θ. For the Large models and Huge models, *i.e.,* DEQ-RAFT-L, DEQ-GMA-L, and DEQ-RAFT-H, we use the same designs as the backbone models while increasing the number of hidden dimensions by a factor of $1.5\times$ and $2\times$, respectively.

B.2. Training Details

Following the settings of prior work [4, 5, 8], we apply a four-stage training protocol using the default hyperparameters unless specified otherwise. First, we train DEQ-RAFT-L and DEQ-RAFT-H models on FlyingChairs [9] for 120K iterations and then on FlyingThings [10] for another 120K iterations. In addition, we also train a DEQ-GMA instantiation on FlyingChairs [9] using a batch size of 10 and a learning rate of 4e-4, with the same setting as RAFT [4]. We note that the original recurrent GMA model quickly exhausts the memory budget even with a much smaller batch size and mixed-precision training, which is not a concern for DEQ-GMA due to the implicit modeling framework.

We used an Anderson acceleration [3] solver with up to 40 forward steps and 1 correction term for the base model (DEQ-RAFT-B). For the larger models (DEQ-RAFT-L, DEQ-RAFT-H, DEQ-GMA-B, and DEQ-GMA-L) we used an Anderson solver with up to 36 forward steps and 1-2 correction terms. For DEQ-RAFT-D, we used a Broyden [2] solver with up to 21 steps and 2 correction terms. Our implementation of the fixed point solvers is based directly on that of Bai et al. [11]. In all those cases, the fixed-point solvers stop either when the iterations reach the limit, or if the absolute residual error falls below 1. Moreover, note that we use one-step gradient, which suggests almost-free backward passes. Our results suggest that, with minimal tuning, these settings are sufficient to achieve state-of-the-art results (while imposing much lower compute and memory cost). For the last training stage, we fine-tune DEQ-RAFT-D on KITTI [12] using 30 solver steps. Empirically, we note that a DEQ flow model trained using a stronger solver (*e.g.,* the Broyden solver) and with more steps typically leads to slightly better performance. This implies the prospect of potentially further boosting the DEQ-flow performance by more precise fixed-point solving; e.g., with more advanced (or even learnable) solvers. All the experiments were conducted on two 11 GB GPUs (NVIDIA 2080Ti's).

C. DEQ Flows with Fixed-point Correction

The growing instability problem has been a longstanding challenge in training implicit neural networks like DEQs. One of the contributions of this paper is also the introduction of fixed-point correction term to stabilize the DEQ flow estimation convergence. Specifically, previous methods rely on more constrained regularization settings such as Jacobian-based losses [7], *i.e.,* penalizing the upper bound of Jacobian spectral radius,

$$\rho(J_{f_\theta}(\mathbf{z}^*)) \leq \|J_{f_\theta}(\mathbf{z}^*)\|_F = \sqrt{\operatorname{tr}(J_{f_\theta}^\top J_{f_\theta})},$$

where $J_{f_\theta}(\mathbf{z}^*) \in \mathbb{R}^{d\times d}$ denotes the Jacobian of f_θ at $\mathbf{z}^*$, ρ corresponds to the spectral radius of a square matrix. By the stochastic Hutchinson trace estimator [13], we have

$$\operatorname{tr}(J_{f_\theta}^\top J_{f_\theta}) = \mathbb{E}_{\epsilon\sim p(\epsilon)}\left[\epsilon^\top J_{f_\theta}^\top J_{f_\theta}\epsilon\right] \approx \sum_{\epsilon\sim p(\epsilon)} \|J_{f_\theta}\epsilon\|_2^2,$$

where $p(\epsilon)$ can be the Gaussian distribution $\mathcal{N}(0, I_d)$ or the Rademacher distribution. Different from prior works, we advocate for exploiting the benefit of IFT and inexact gradient to *sparsely* apply a fixed-point correction scheme to the convergence path.

In this section, we present an ablation study on FlyingChairs [9] that compare the stability and generalization performance of DEQ flow models trained in three different settings: 1) standard implicit differentiatio (i.e., IFT); 2) standard IFT with Jacobian regularization [7]; and 3) our proposed fixed-point correction scheme with a single correction term. As mentioned previously, we perform one-step inexact gradient on the correction loss as well. For the purpose of this ablation, we run the forward fixed-point solver for a limited compute budget of 16 Anderson [3] steps in all three settings, and analyze their convergence behavior accordingly.

As shown in Fig. 1, the model trained using the standard implicit function theorem (IFT) suffers from the "growing instability" issue (see red curve in Fig. 1 (a)), as described in prior works indeed [7, 11, 14, 15]. While strong enough Jacobian regularization can indeed stabilize the training process and lead to good overall convergence (see orange curve in Fig. 1 (a)), we observe that it is usually at a heavy cost of optical flow estimation accuracy (see Fig. 1 (b)). This agrees with the conclusion of Bai et al. [7]. In contrast, we find it suffices to use a single fixed-point correction term in DEQ flow to achieve the same stabilizing effect (see blue curve in Fig. 1 (a)) *at no extra cost* to the average EPE on the validation set. We hypothesize that such a fixed-point correction method may suggest an elegant and lightweight solution to the growing instability problem in the broader implicit deep learning community as well (*i.e.,* beyond the scope of optical flow estimation), which we leave for future work.

D. Qualitative Results

We visualize the flow estimation by the DEQ flow model in Fig. 2, Fig. 3, Fig. 4, Fig. 5, and Fig. 6, using consecutive frames of the MPI Sintel [16] test set.

References

[1] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-performance Deep Learning Library. In *Neural Information Processing Systems (NeurIPS)*, pages 8026–8037, 2019. 1

[2] Charles G Broyden. A Class of Methods for Solving Nonlinear Simultaneous Equations. *Mathematics of computation*, 19 (92):577–593, 1965. 1, 2

[3] Donald G. Anderson. Iterative procedures for nonlinear integral equations. *Journal of the ACM (JACM)*, 12(4):547–560, October 1965. 1, 2, 3

[4] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European conference on computer vision*, pages 402–419. Springer, 2020. 1, 2

[5] Shihao Jiang, Dylan Campbell, Yao Lu, Hongdong Li, and Richard Hartley. Learning to estimate hidden motions with global motion aggregation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9772–9781, October 2021. 1, 2

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Neural Information Processing Systems (NeurIPS)*, 2017. 1

[7] Shaojie Bai, Vladlen Koltun, and J. Zico Kolter. Stabilizing Equilibrium Models by Jacobian Regularization. In *International Conference on Machine Learning (ICML)*, 2021. 2, 3

[8] Shihao Jiang, Yao Lu, Hongdong Li, and Richard I. Hartley. Learning optical flow from a few matches. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16587–16595, 2021. 2

[9] 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. 2, 3

[10] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4040–4048, 2016. 2

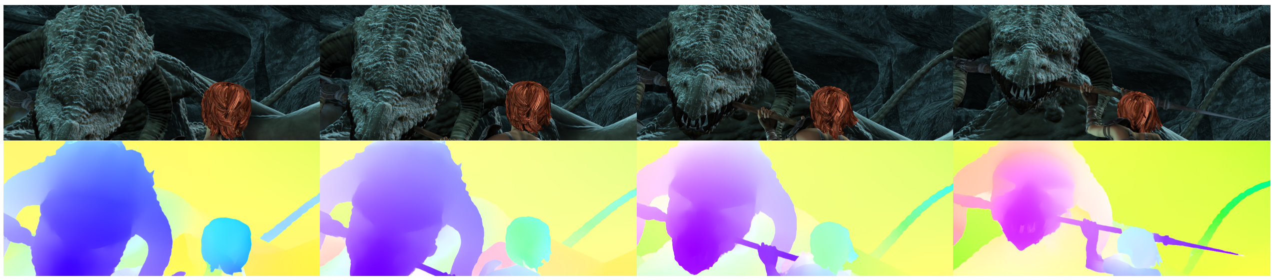
[11] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. Deep equilibrium models. In *Neural Information Processing Systems (NeurIPS)*, 2019. 2, 3

[12] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. 2

[13] Michael F Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3): 1059–1076, 1989. 2

(a) Frame 12 (b) Frame 13 (c) Frame 14 (d) Frame 15

Figure 2. Visualization on the Sintel test set, `ambush_1` sequence of the clean split.

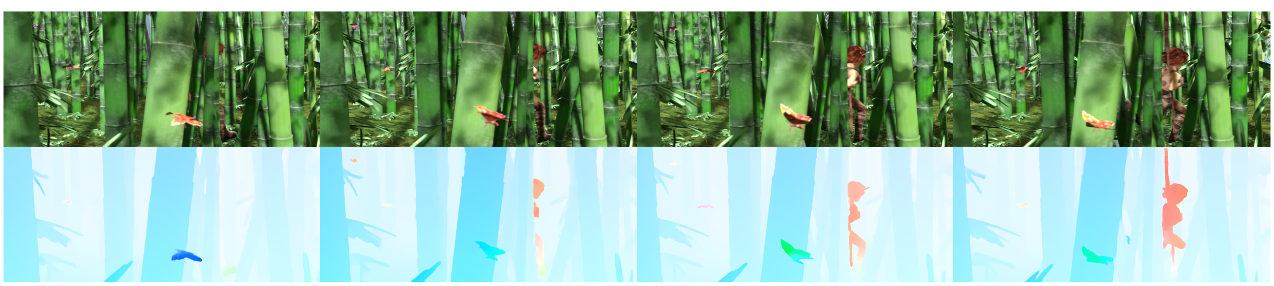


(a) Frame 34 (b) Frame 35 (c) Frame 36 (d) Frame 37

Figure 3. Visualization on the Sintel test set, `cave_3` sequence of the clean split.

(a) Frame 25 (b) Frame 26 (c) Frame 27 (d) Frame 28

Figure 4. Visualization on the Sintel test set, `market_1` sequence of the clean split.



(a) Frame 25 (b) Frame 26 (c) Frame 27 (d) Frame 28

Figure 5. Visualization on the Sintel test set, `bamboo_3` sequence of the final split.

(a) Frame 25 (b) Frame 26 (c) Frame 27 (d) Frame 28

Figure 6. Visualization on the Sintel test set, `temple_1` sequence of the final split.

[14] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Neural Information Processing Systems (NeurIPS)*, 2018. 3

[15] Ezra Winston and J. Zico Kolter. Monotone operator equilibrium networks. In *Neural Information Processing Systems (NeurIPS)*, pages 10718–10728, 2020. 3

[16] Daniel J Butler, Jonas Wulff, Garrett B Stanley, and Michael J Black. A naturalistic open source movie for optical flow evaluation. In *European conference on computer vision*, pages 611–625. Springer, 2012. 3