Deep Equilibrium Optical Flow Estimation

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

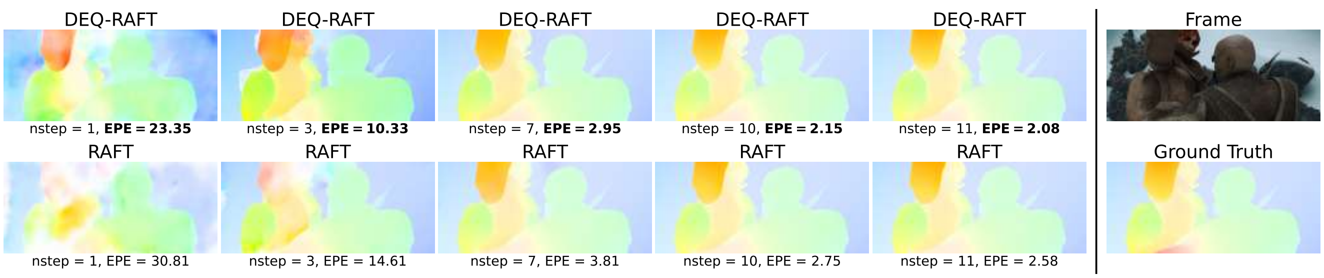


Figure 1. **A deep equilibrium (DEQ) flow estimator directly models the flow as a path-independent, "infinite-level" fixed-point solving process.** We propose to use this implicit framework to replace the existing recurrent approach to optical flow estimation. The DEQ flows converge faster, require less memory, are often more accurate, and are compatible with prior model designs (e.g., RAFT [1] and GMA [2]).

Abstract

Many recent state-of-the-art (SOTA) optical flow models use finite-step recurrent update operations to emulate traditional algorithms by encouraging iterative refinements toward a stable flow estimation. However, these RNNs impose large computation and memory overheads, and are not directly trained to model such "stable estimation". They can converge poorly and thereby suffer from performance degradation. To combat these drawbacks, we propose deep equilibrium (DEQ) flow estimators, an approach that directly solves for the flow as the infinite-level fixed point of an implicit layer (using any black-box solver) [3], and differentiates through this fixed point analytically (thus requiring $O(1)$ training memory). This implicit-depth approach is not predicated on any specific model, and thus can be applied to a wide range of SOTA flow estimation model designs (e.g., RAFT [1] and GMA [2]). The use of these DEQ flow estimators allows us to compute the flow faster using, e.g., fixed-point reuse and inexact gradients, consumes $4 \sim 6\times$ less training memory than the recurrent counterpart, and achieves better results with the same computation budget. In addition, we propose a novel, sparse fixed-point correction scheme to stabilize our DEQ flow estimators, which addresses a longstanding challenge for DEQ models in general. We test our approach in various realistic settings and show that it improves SOTA methods on Sintel and KITTI datasets with substantially better computational and memory efficiency.

*Equal contribution. Our code is available here.

1. Introduction

Optical flow estimation is the classic computer vision task of predicting the pixel-level motions between video frames [1, 4–7]. Learning-based approaches to this problem, which outperformed classical approaches, proposed the use of conventional deep convolutional networks to learn a flow estimate [6–8]. Recent progress has shown that finite-step, unrolled and recurrent update operations significantly improve the estimation performance, exemplified by the emergence of the RAFT [1] method. Contemporary optical flow models that employ this approach typically rely on a Gated Recurrent Unit (GRU) [9] to *iteratively refine* the optical flow estimate. This approach was motivated to emulate traditional optimization-based methods, and the update operators defined accordingly have become the standard design for state-of-the-art flow models [1, 2, 10–12].

Despite their superior performance, these rolled-out recurrent networks suffer from a few drawbacks. First, training these models involves tracking a long hidden-state history in the backpropagation-through-time (BPTT) algorithm [13], which yields a significant computational and memory burden. Therefore, these models tend to scale poorly with larger images and more iterations. Second, although these models were designed to emulate *traditional optimization approaches* which solve for a "stable estimate" with as many steps as needed, the recurrent networks do not directly model such a minimum-energy optima state. Rather, they stop after a predefined L update steps, and are still trained in a path-dependent way using BPTT. We also show later in Fig. 3 that the GRUs frequently oscillate instead of converging.

In this work, we introduce deep equilibrium (DEQ) flow estimators based on recent progresses in implicit deep learning, represented by DEQ models [3, 14–17]. Our method functions as a superior and natural framework to replace the existing recurrent, unrolling-based flow estimation approach. There are multiple reasons why this approach is preferable. **First**, instead of relying on the naïve iterative layer stacking, DEQ models define their outputs as the fixed points of a single layer f_θ using the input $\mathbf{x}$, *i.e.,* $\mathbf{z}^* = f_\theta(\mathbf{z}^*, \mathbf{x})$, modeling an "infinite-layer" equilibrium representation. We can directly solve for the fixed point using specialized black-box solvers, *e.g.,* quasi-Newton methods [18, 19], in a spirit much more consistent with the traditional optimization-based perspective [5, 20]. This approach expedites the stable flow estimation process while often yielding better results. **Second**, we no longer need to perform BPTT. Instead, DEQ models can directly differentiate through the final fixed point $\mathbf{z}^*$ without having to store intermediary states during the forward computation, considerably lowering the training memory cost. **Third**, this fixed-point formulation justifies numerous implicit network enhancements such as 1) fixed-point reuse from adjacent video frames; and 2) inexact gradients [21–23]. The former helps avoid redundant computations, thus substantially accelerating flow estimations; and the latter makes the backward pass computationally *almost free*! **Fourth**, the DEQ approach is not predicated on any specific structure for f_θ. Therefore, DEQ is a *framework* that applies to a wide range of these SOTA flow estimation model designs (*e.g.,* RAFT [1], GMA [2], and Depthstillation [24]), and we can obtain the aforementioned computational and memory benefits with even additional gain based on the specific structure of f_θ.

In addition to suggesting DEQ flow estimators as a superior replacement to the existing recurrent approach, we also tackle the longstanding instability challenge of training DEQ networks [3, 15, 25, 26]. Inspired by the RAFT model, we propose a novel, sparse fixed-point correction scheme that substantially stabilizes our DEQ flow estimators.

The contributions of this paper are as follows. **First**, we propose the deep equilibrium (DEQ) approach as a new natural starting point for formulating optical flow methods. A DEQ approach directly models and substantially accelerates the fixed-point convergence of the flow estimation process, avoids redundant computations across video frames, and comes with an almost-free backward pass. **Second**, we show that the DEQ approach is orthogonal to, and thus compatible with, the prior modeling efforts (which focus on the model design and feature extraction) [1, 2] and data-related efforts [10]. With DEQ, these prior arts are now more computationally and memory efficient as well as more accurate. For instance, on KITTI-15 [27] (*train*) a zero-shot DEQ-based RAFT model further reduces the state-of-the-art F1-all measure by 14.0% while using the same underlying RAFT operator. **Third**, we introduce a sparse fixed-point correction scheme that significantly stabilizes DEQ models on optical flow problems while only adding minimal cost, and show that on flow estimation tasks this approach is superior to the recently proposed Jacobian-based regularization [26].

2. Related Work

Iterative Optical Flow. Although optical flow is a classical problem, there has recently been substantial progress in the area. Earlier methods [5, 28–31] formulated the optical flow prediction as an energy minimization problems using continuous optimization with different objective terms. This perspective inspired multiple improvements that used discrete optimization to model optical flows, *i.e.,* those based on conditional random fields [32], global optimization [33], and inference on the global 4D cost volume [34]. More recently, with the advancement of deep learning, there have been an explosion of efforts trying to emulate these optimization steps via deep neural networks. For example, a number of optical flow methods are based on deep architectures that rely on coarse-to-fine pyramids [6, 8, 35–39]. Specifically, recent research efforts have turned to iterative refinements, which typically involves stacking multiple direct flow prediction modules [38, 40]. The RAFT model [1], which inspired this work, first showed they could achieve state-of-the-art performance on optical flow tasks using a correlation volume and a convolutional GRU update operator that mimics the behavior of traditional optimizers, which tends to converge to a stable flow estimate. Built on top of this recurrent unrolling framework of RAFT, Jiang et al. [2] introduced an additional self-attention-style global motion aggregation (GMA) module prior to the recurrent stage to improve the modeling of the occlusions. Another contemporary work, AutoFlow [10], exploits bilevel optimization to automatically render and augment training data for optical flow. Finally, Jiang et al. [41] proposes to speed up these flow estimators by replacing the dense correlation volume with a sparse alternative.

The focus of this paper is on a direction that is largely orthogonal to and thus complementary to these modeling efforts. We challenge and improve the "default" *recurrent, unrolled* formulation of training flow estimators themselves. With the help of the recent progress in implicit deep learning (see below), we can maintain the same convergent flow estimation formulation while paying substantially less computation and memory costs.

Implicit deep learning. Recent research has proposed a new class of deep learning architectures that do not have prescribed computation graphs or hierarchical layer stacking like conventional networks. Instead, the output of these implicit networks is typically defined to be the solution of an underlying dynamical system [3, 17, 25, 42, 43]. For example, Neural ODEs [25] model infinitesimal steps of a residual block as an ODE flow. A deep equilibrium (DEQ) network [3] (which this work is primarily inspired by) is another class of implicit model that directly solves for a fixed-point representation of a shallow layer f_θ (*e.g.,* , a Transformer block) and differentiates through this fixed point without storing intermediate states in the forward pass. This allows one to train implicit networks with *constant* memory, while fully decoupling the forward and backward passes of training. However, it is known that these implicit models suffer from a few serious issues that have been studied by later works, such as computational inefficiency [25, 44], instability [3, 25, 26], and lack of theoretical convergence guarantees [15, 16]. On a positive note, followup works have also shown that DEQ-based models can achieve competitive results on challenging tasks such as language modeling [3], generative modeling [45], semantic segmentation [14], etc. However, to the best of our knowledge, these implicit models have not been applied to the task of optical flow estimation. In this paper, we show that this task could substantially benefit from the DEQ formulation as well.

3. Method

We start by introducing some preliminaries of existing flow estimators. These modules are typically applied directly on raw image pairs, with the extracted representations then passed into the iterative refinement stage. We use RAFT [1] as the illustrative example here while noting that cutting-edge flow estimators generally share similar structure (*i.e.,* , for context extraction and visual correlation computations).

3.1. Preliminaries

Given an RGB image pair $\mathbf{p}^1, \mathbf{p}^2 \in \mathbb{R}^{3\times H\times W}$, an optical flow estimator aims to learn a correspondence $\mathbf{f} \in \mathbb{R}^{2\times H\times W}$ between two coordinate grids $\mathbf{c}^1, \mathbf{c}^2$ (*i.e.,* $\mathbf{f} = \mathbf{c}^2 - \mathbf{c}^1$), which describes the per-pixel motion between consecutive frames in the horizontal (dx) and vertical (dy) directions. To process the matched image pair, we first encode features $\mathbf{u}^1, \mathbf{u}^2 \in \mathbb{R}^{C\times H\times W}$ of $\mathbf{p}^1, \mathbf{p}^2$, and produce a context embedding $\mathbf{q}$ from the first image $\mathbf{p}^1$. Then, we construct a group of pyramid global correlation tensors $\mathcal{C} = \{\mathcal{C}^0, \cdots, \mathcal{C}^{p-1}\}$, where $\mathcal{C}^k \in \mathbb{R}^{H\times W\times H/2^k\times W/2^k}$ is found by first calculating the inner product between all pairs of hyperpixels in $\mathbf{u}^1$ and $\mathbf{u}^2$ as $\mathcal{C}^0$, *i.e.,*

$$\mathcal{C}^0_{ijmn} = \sum_d \mathbf{u}^1_{ijd}\mathbf{u}^2_{mnd} \tag{1}$$

followed by downsampling the last two dimensions to produce $\mathcal{C}^k$ ($k > 0$). The correlation pyramid $\mathcal{C}$ and context embedding $\mathbf{q}$, which allow the model to infer large motions and displacements in a global sense, are then passed as inputs into the *iterative refinement* stage.

In this work, we keep the correlation and context computation part intact (see Fig. 2) and concentrate on the iterative refinement stage. We refer interested readers to Teed and Deng [1] for a more detailed description of the feature extraction process.

3.2. Deep Equilibrium Flow Estimator

Due to the inherent challenges of the flow estimation task, prior works have shown that explicit neural networks struggle to predict the flow accurately, requiring a prohibitively large number of training iterations [6]. Recent works [1, 2, 24] have resorted to mimicking the flavor of traditional optimization-based algorithms [5] with RNNs (*e.g.,* convGRUs). However, these methods are still quite different from the traditional methods in a few ways. For example, optimization-based methods 1) have an adaptive and well-defined stopping criteria (*e.g.,* whenever they reach the optima); 2) are agnostic to the choice of solver (*e.g.,* first- or second-order methods); and 3) are essentially path-independent (i.e., the output alone is the only thing we should need). None of these properties are directly characterized by the finite-step unrolling of recurrent networks.

We propose to close this gap with a DEQ-based approach. Specifically, given the context embedding $\mathbf{q}$ and the pyramid correlation tensor $\mathcal{C}$, a DEQ flow estimator simultaneously solves for the fixed-point convergence of two alternate streams: 1) a latent representation $\mathbf{h}$, which constructs the flow updates; and 2) the flow estimate $\mathbf{f}$ itself, whose updates are generically related as follows:

$$\begin{aligned} \mathbf{h}^{[t+1]} &= \mathcal{H}(\mathbf{h}^{[t]}, \mathbf{f}^{[t]}, \mathbf{q}, \mathcal{C}) \\ \mathbf{f}^{[t+1]} &= \mathcal{F}(\mathbf{h}^{[t+1]}, \mathbf{f}^{[t]}, \mathbf{q}, \mathcal{C}). \end{aligned} \tag{2}$$

This formulation captures the form of prominent flow estimator model designs like RAFT [1] or GMA [2]. Formally, the input $\mathbf{x} = (\mathbf{q}, \mathcal{C})$ and model parameters $f_\theta = (\mathcal{H}, \mathcal{F})$ jointly define a dynamical system that the DEQ flow model can *directly* solve the fixed-point for using the following flow update equation in its forward pass:

$$(\mathbf{h}^*, \mathbf{f}^*) = \mathbf{z}^* = f_\theta(\mathbf{z}^*, \mathbf{x}) = f_\theta((\mathbf{h}^*, \mathbf{f}^*), \mathbf{x}). \tag{3}$$

Intuitively, this corresponds to an "infinite-depth" feature representation $\mathbf{z}^*$ where, if we perform one more flow update step f_θ, both flow estimation $\mathbf{f}$ and latent state $\mathbf{h}$ will not change (thus reaching a fixed point, *i.e.,* an "equilibrium"). Importantly, we can leverage much more advanced root solving methods like quasi-Newton methods (*e.g.,* Broyden's method [18] or Anderson mixing [19]) to find the fixed point. These methods guarantee a much faster (superlinear) and better-quality convergence than if we perform infinitely many naïve unrolling steps (as do recurrent networks but only up to a finite number of steps due to computation and memory constraints). Moreover, we note that prior works on implicit networks have shown that the exact structure of f_θ subsumes a wide variety of model designs, such as a

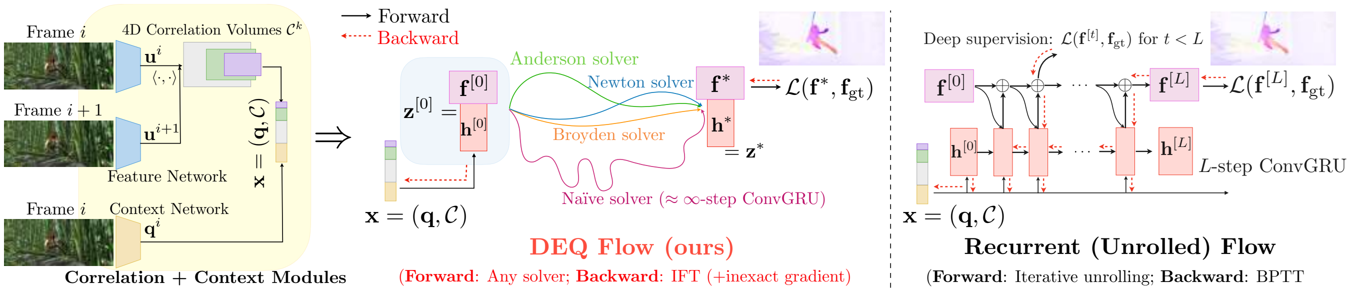


Figure 2. A visual comparison of the DEQ flow estimator and the recurrent unrolled flow estimator. After the correlation and context modules (see Sec. 3.1), a DEQ flow uses a fast, black-box fixed-point solver (e.g., Anderson) to directly solve for a stable (fixed-point) flow $\mathbf{z}^* = (\mathbf{h}^*, \mathbf{f}^*)$, and differentiate through $\mathbf{z}^\star$ with a cheap inexact gradient. This makes a DEQ flow's backward pass almost free. In contrast, a recurrent flow estimator has to be unrolled for many steps, and needs to perform BPTT, which is costly in both computation and memory.

Transformer block [3, 46], a residual block [14, 47], or a graph layer [48–50]. Similarly, for the deep equilibrium flow estimator, Eq. (2) engulfs exactly the designs of state-of-the-art optical flow models, which we follow and use without modification. For example, for RAFT [1],

$$\begin{aligned} \mathbf{x} &= \text{Conv2d}\left([\mathbf{q},\, \mathbf{f}^*, \mathcal{C}(\mathbf{f}^* + \mathbf{c}^0)]\right) \\ \mathbf{h}^* &= \text{ConvGRU}\left(\mathbf{h}^*, [\mathbf{x},\, \mathbf{q}]\right) \\ \mathbf{f}^* &= \mathbf{f}^* + \text{Conv2d}\left(\mathbf{h}^*\right), \end{aligned} \tag{4}$$

where $\mathcal{C}(\mathbf{f}^* + \mathbf{c}^0)$ stands for the correlation lookup as in RAFT [1]. We also show in Appendix that GMA [2] can be easily written in a similar update form.

The key question is, how do we update and train a DEQ flow estimator. It turns out that we can directly differentiate through this "infinite-level" flow state, $(\mathbf{h}^*, \mathbf{f}^*)$, without any knowledge of the forward fixed-point trajectory:

Theorem 1. *(Implicit Function Theorem (IFT) [3, 51]) Given the fixed-point flow representation $\mathbf{z}^* = (\mathbf{h}^*, \mathbf{f}^*)$, the corresponding flow loss $\mathcal{L}(\mathbf{h}^*, \mathbf{f}^*, \mathbf{f}_{gt})$ and input $\mathbf{x} = (\mathbf{q}, \mathcal{C})$, the gradient of DEQ flow is given by*

$$\frac{\partial \mathcal{L}}{\partial \theta} = \frac{\partial \mathcal{L}}{\partial \mathbf{z}^*}\left(I - \frac{\partial f_\theta}{\partial \mathbf{z}^*}\right)^{-1} \frac{\partial f_\theta(\mathbf{z}^*, \mathbf{x})}{\partial \theta} \tag{5}$$

For the proof, see Bai et al. [3]. Importantly, this theorem enables us to *decouple* the forward and backward passes of a DEQ flow estimator; *i.e.,* to perform gradient update, we only need the final output $\mathbf{z}^*$ and do not need to run backpropagation-through-time (BPTT). It means a huge memory reduction: whereas an L-step recurrent flow estimator takes $O(L)$ memory to perform BPTT, a DEQ estimator reduces the overhead by a factor of L to be $O(1)$ (*e.g.,* RAFT uses $L = 12$ for training, so using a DEQ flow can theoretically reduce the iterative refinement memory cost by $12\times$).

To summarize, a DEQ flow's forward pass directly solves a fixed-point flow-update equation via black-box solvers; and its backward pass relies only on the final optimum $\mathbf{z}^*$, which make this flow estimation process much more akin to the optimization-based perspective [5].

3.3. Accelerating DEQ Flows

Formulating optical flow estimation as a deep equilibrium solution also enables us to fully exploit the toolkit from implicit deep learning. We elaborate below on examples of how this equilibrium formulation can substantially help us improve the forward and backward pipeline and significantly simplify the overall overhead of modern flow estimators.

Inexact Gradients for Training DEQs. Despite the niceness of the implicit function theorem (IFT), inverting the Jacobian term could quickly become intractable as we deal with high-dimensional feature maps. To combat this, Bai et al. [3] proposed exploiting fast vector-Jacobian products and solving a linear fixed-point system $\mathbf{g}^\top = \mathbf{g}^\top \frac{\partial f_\theta}{\partial \mathbf{z}^*} + \frac{\partial \mathcal{L}}{\partial \mathbf{z}^*}$. However, this approach is still iterative in nature, and in practice, it is no cheaper than the forward flow solving process.

Recent works on implicit networks' backward dynamics [21–23] suggest that they can typically be trained, and even benefit from, simple approximations of the IFT, while still modeling an "infinite-depth" representation through the fixed-point forward pass. That is, we do not need the exact solution to Thm. 1 to train these networks. Instead we use

$$\frac{\partial \mathcal{L}}{\partial \theta} \approx \widehat{\frac{\partial \mathcal{L}}{\partial \theta}} = \frac{\partial \mathcal{L}}{\partial \mathbf{z}^*} A \frac{\partial f_\theta(\mathbf{z}^*, \mathbf{x})}{\partial \theta} \tag{6}$$

where A is a Jacobian (inverse) approximation term. For example, [21, 22] proposes to use $A = I$ (i.e., Jacobian-free), which simplifies the backward pass of a DEQ flow estimator to a *single step* computation $\frac{\partial \mathcal{L}}{\partial \theta} \approx \frac{\partial \mathcal{L}}{\partial \mathbf{z}^*} \frac{\partial f_\theta(\mathbf{z}^*, \mathbf{x})}{\partial \theta}$. Therefore, unlike the BPTT-based recurrent framework used by existing flow estimators, a DEQ flow estimator's backward pass that uses inexact gradient consists of a single step (and thus is almost free)! Empirically, since we almost eliminate the backward pass cost, the inexact gradients significantly reduce the total training time for DEQ flow estimator further by a factor of almost $2\times$. The capability of using inexact gradients is a direct and unique consequence of the fixed-point formulation and assumes a certain level of stability for

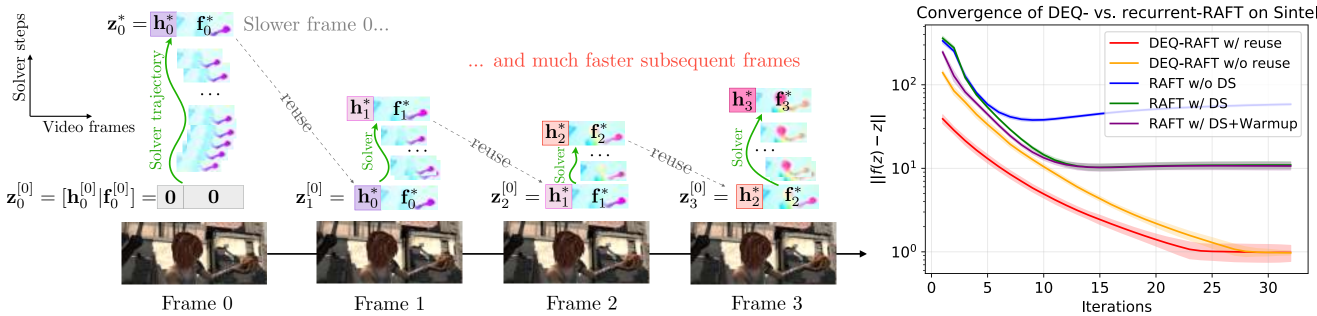


Figure 3. (Left) By reusing fixed-point $\mathbf{z}^*$ from the previous frame's flow estimation, we can "jump start" the subsequent equilibrium solving, essentially amortizing the solver cost and speeding up convergence. (Right) Comparing forward convergence of DEQ and recurrent flow estimators on Sintel videos (50 frames). "DS" stands for deep supervision used by RAFT [1]. DEQ flow with fixed-point reuse converges best; and overall, DEQ flows converge faster than RAFT [1].

the underlying dynamics [21–23, 26]. We discuss an additional approach that further improves the stability of these estimates next.

Sparse fixed-point correction of DEQ flows. A long-standing challenge in training implicit networks is *the growing instability problem* [3, 15, 22, 23, 25, 26, 44, 52]. In short, since DEQ flow estimators have no discrete layers (or steps), they struggle to converge during training. In other words, the stable flow estimate $\mathbf{z}^* = (\mathbf{h}^*, \mathbf{f}^*)$ could become computationally expensive to reach. This suggests that the optical flow estimation process gets slower during training.

In this work, we propose sparsely applying a fixed-point correction term to stabilize the DEQ flow convergence. Formally, suppose the black-box solver (e.g., Broyden's method) yields a convergence path $(\mathbf{z}^{[0]}, \dots, \mathbf{z}^{[i]}, \dots \mathbf{z}^*)$, where $\mathbf{z}^{[0]}$ is the initial guess and $\mathbf{z}^*$ is the final flow estimate. We then randomly pick $\mathbf{z}^{[i]} = (\mathbf{h}^{[i]}, \mathbf{f}^{[i]})$ on this path (e.g., can be uniformly spaced), and define our total loss to be

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{main}} + \mathcal{L}_{\text{cor}} = \underbrace{\|\mathbf{f}^* - \mathbf{f}_{\text{gt}}\|_2^2}_{\text{main loss}} + \gamma \underbrace{\|\mathbf{f}^{[i]} - \mathbf{f}_{\text{gt}}\|_2^2}_{\text{fixed-point correction}} \quad (7)$$

where $\gamma < 1$ is a loss weight hyperparameter. This was inspired by the dense step-wise deep supervision used by conventional flow estimators like RAFT [1]. However, our application here differs in two significant ways. First, we apply this in a sparse manner, with our primary goal being correcting instability. Second, unlike in RAFT, which performs costly BPTT through the RNN chain, this fixed-point correction loss is still *path-independent* and can be understood as a coarse-grained fixed-point estimate. Therefore, we could also perform inexact gradient updates on this correction loss as well; *i.e.,*

$$\frac{\partial \mathcal{L}_{\text{cor}}}{\partial \theta} \approx \gamma \frac{\partial \mathcal{L}_{\text{cor}}}{\partial \mathbf{z}^{[i]}} \frac{\partial f_\theta(\mathbf{z}^{[i]}, \mathbf{x})}{\partial \theta}. \quad (8)$$

Empirically, we find this significantly stabilizes the DEQ flow estimator while having no noticeable negative impact on performance. This result is in sharp contrast to existing stabilization methods like Jacobian regularization [26, 53] which 1) apply only locally to $\mathbf{z}^*$; and 2) usually hurt model accuracy (see the ablation study in Sec. 4). Moreover, due to the inexact gradient in Eq. (8), our method adds almost no extra computation or memory cost.

While our scope is limited to flow estimation here, we believe this approach suggests a potentially valuable and lightweight solution to the generic instability issue of implicit models, which we leave for future work.

Fixed-point reuse for better initialization. The DEQ flow estimator's unique formulation also inherits many useful properties from the general optimization framework. One of these nice properties is the ability to perform fixed-point reuse to further accelerate flow estimation convergence. The motivation for this comes from the fact that consecutive frames of a video are typically highly correlated. For instance, perhaps only a few objects are moving in the foreground, while most of the other content and background are nearly identical across these adjacent frames. More formally, if $\mathbf{p}^i$, $\mathbf{p}^{i+1}$, and $\mathbf{p}^{i+2}$ are 3 consecutive video frames, then the ground-truth optical flow $\mathbf{f}_i$ (between $\mathbf{p}^i$ and $\mathbf{p}^{i+1}$) is usually highly correlated to the next ground-truth optical flow $\mathbf{f}_{i+1}$. Thus, when we perform real-time flow estimation with conventional networks like FlowNet [6] and RAFT [1], we frequently perform a lot of *redundant* computations. In contrast, with a DEQ flow, we can recycle the fixed-point solution $\mathbf{z}_i^*$ of the previous frame, which estimates $\mathbf{f}_i$, as the initial guess $\mathbf{z}_{i+1}^{[0]}$ for the subsequent frame's fixed-point solver. Intuitively, these DEQ flows are able to automatically adjust their forward optimization by exploiting this more informed initial guess, which facilitates convergence speed. It amortizes the cost of flow estimation over long video sequences, since only frame 0 requires full fixed-point solving while the remaining frames can all recycle their predecessor's flow. We note that such reuse is related to, but still different from the warm-up scheme of RAFT [1], which only applies to $\mathbf{f}$, excludes $\mathbf{h}$, and still has to be unrolled for many steps. In our case, because a DEQ flow directly models a fixed point, such an adaptive computation by exploiting the inductive bias of video data is well-justified.

Fig. 3 shows the practicality of fixed-point reuse on Sintel video sequences. By re-using the fixed point, we can further accelerate the DEQ flow estimator's inference speed by a factor of about $1.6\times$. Interestingly, while RAFT's iterative unrolling aims to mimic the iterative convergence, we find its activations usually oscillate at a relatively high level after about 15 update iterations.

To summarize, while a conventional recurrent flow estimator like RAFT needs to be unrolled for some finite L steps and back-propagated through the same L-step chain, a deep equilibrium flow estimator: 1) leverages the IFT and requires only $O(1)$ training memory, 2) uses inexact gradients to reduce the backward pass to $O(1)$ computation, and 3) can take advantage of correlation between adjacent frames to amortize the flow estimation cost across a long sequence, thus significantly accelerating the forward pass.

4. Experiments

We present the results of our experiments in this section. Specifically, we highlight the computational and memory efficiency of DEQ flow estimators and analyze how the fixed-point correction improves the DEQ flow. Our method achieves state-of-the-art zero-shot performance on both the MPI Sintel [54] dataset and the KITTI 2015 [27] dataset, with an astonishing 12.9% error reduction in the F1-all score and 6.6% improvement in EPE for KITTI-15 (while still using a similar training budget to RAFT [1]).

4.1. Results

Our quantitative evaluation is presented in Tab. 1. Following previous work [1, 2], we first pretrain the DEQ flow model on the FlyingChairs [6] and FlyingThings3D [55] datasets. We then test the model on the training set of MPI Sintel [54] and KITTI 2015 [27] datasets. This model is denoted "C + T"; it evaluates the *zero-shot generalization* of the DEQ flow model. Then, we fine-tune the DEQ flow estimator on FlyingThings3D [55], MPI-Sintel [54], KITTI 2015 [27], and HD1K [56] for the test submission.

The models we train are of exactly the same size as RAFT (5.3M) [1] and GMA (5.9M) [2] except they use DEQ flow formulation instead of recurrent updates. They are denoted as DEQ-RAFT-B and DEQ-GMA-B, respectively. Exploiting the memory efficiency of the DEQ flow model (see Sec. 4.2), we can fit much larger models into the same compute budget of two 11 GB 2080Ti GPUs. To this end, we also trained DEQ-RAFT-L (8.4M) and DEQ-RAFT-H (12.8M) by increasing the the width of hidden layers inside the update operator. We also trained DEQ-RAFT-D (9.4M) by duplicating the ConvGRU within f_θ. As shown in Fig. 5, even the largest DEQ-RAFT-H model only consumes less than half of the flow estimation memory used by a standard-sized RAFT model, while achieving significantly better accuracy (4.38 AEPE and 14.9 F1-all score on KITTI-15, see Tab. 1).

4.2. Performance-Compute Tradeoff

We further verify the aforementioned computational and memory benefits of the DEQ flow model on the Sintel (clean) [54] dataset with a RAFT-based update operator (see Eq. (4)) trained on FlyingChairs [6] and FlyingThings3D [55]. The results are shown in Fig. 5. Specifically, when training the DEQ flow estimator on Sintel with a batch size of 3 per GPU (the maximum that RAFT can fit with a 11 GB GPU), we observe that the memory cost of the flow estimation process reduces by a factor of over $4\times$ (red bars). Note that since we keep the rest of the model intact (e.g., correlation pyramid and context extraction; see Sec. 3.1), the DEQ flow estimator does not improve those parts of the memory burden, which now becomes the new dominant source of memory overhead. In addition, when we use the model for inference, we follow Teed and Deng [1] using 32 recurrent steps for RAFT (with warm-start), and the Anderson solver for DEQ-RAFT (with reuse), which stops if relative residual falls below $\varepsilon = 10^{-3}$. Our results suggest that the DEQ flow converges to an accurate solution, and it is in practice about 20% faster than the RAFT models with the same structure and size (blue bars). Finally, we show that we can exploit such memory savings to build even larger and more accurate flow estimators (DEQ-RAFT-H), while still staying well within the compute and memory budget.

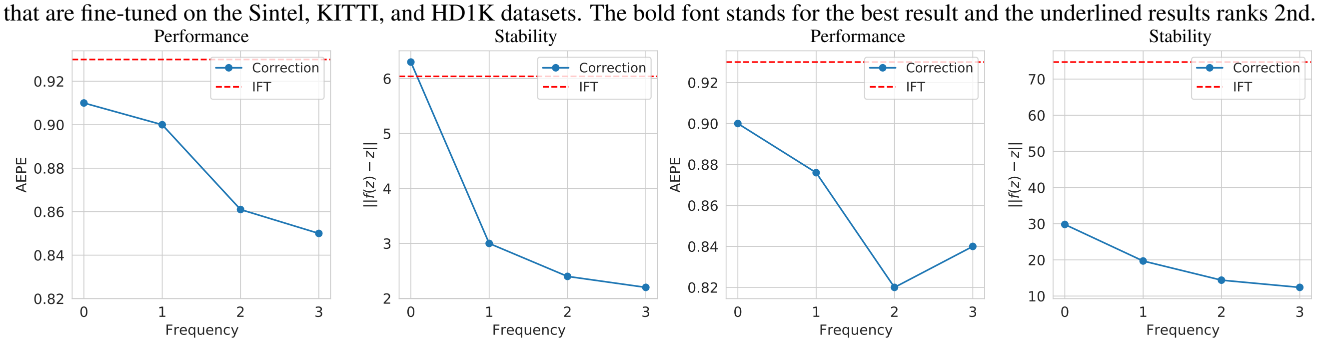
4.3. Ablation Study

In this subsection, we aim to answer the following questions: 1) How useful is the fixed-point correction compared with canonical IFT in performance, stability, and speed? 2) How does the convergence of a DEQ flow correlate with the quality of the flow estimation? As in Sec. 4.2, we use the model design from RAFT [1] to instantiate our DEQ flow. By default, we conduct the ablation experiments on the FlyingChairs [6] dataset using the default training hyperparameters of RAFT and report the Average End Point Error (AEPE) on its validation split.

Stabilizing DEQ by Fixed-Point Correction. As mentioned in Sec. 3.3, unregularized canonical DEQ models (as well as other implicit networks like Neural ODEs [25]) typically suffer from a growing instability issue typically symptomized by an increasingly costly forward fixed-point solving process. We perform an ablation experiment to study how our proposed sparse fixed-point correction scheme could help alleviate this issue. To understand the scheme's effect, we train a DEQ flow model using both an Anderson [19] and a Broyden [18] solver with 36 and 24 forward iterations, respectively. For simplicity, we equally divide the solver convergence trajectory into $r + 1$ segments (where r

Data	Method	Sintel (train)		KITTI-15 (train)		Sintel (test)		KITTI-15 (test)
		Clean	Final	AEPE	F1-all(%)	Clean	Final	F1-all(%)
C + T	LiteFlowNet[37]	2.48	4.04	10.39	28.5	-	-	-
	PWC-Net[35]	2.55	3.93	10.35	33.7	-	-	-
	LiteFlowNet2[8]	2.24	3.78	8.97	25.9	-	-	-
	VCN[39]	2.21	3.68	8.36	25.1	-	-	-
	MaskFlowNet[7]	2.25	3.61	-	23.1	-	-	-
	FlowNet2[38]	2.02	3.54	10.08	30.0	3.96	6.02	-
	RAFT[1]	1.43	2.71	5.04	17.4	-	-	-
	DEQ-RAFT-B	1.48	2.77	5.01	16.3	-	-	-
	DEQ-RAFT-L	1.40	**2.65**	4.76	16.1	-	-	-
	DEQ-RAFT-D	1.39	<u>2.67</u>	<u>4.41</u>	<u>15.4</u>	-	-	-
	DEQ-RAFT-H	1.41	2.75	**4.38**	**14.9**	-	-	-
	GMA[2]	**1.30**	2.74	4.69	17.1	-	-	-
	DEQ-GMA-B	1.35	2.90	4.84	16.2	-	-	-
	DEQ-GMA-L	<u>1.33</u>	2.71	4.72	16.4	-	-	-
C+T+S+K+H	LiteFlowNet2[8]	(1.30)	(1.62)	(1.47)	(4.8)	3.48	4.69	7.74
	PWC-Net+[36]	(1.71)	(2.34)	(1.50)	(5.3)	3.45	4.60	7.72
	VCN [39]	(1.66)	(2.24)	(1.16)	(4.1)	2.81	4.40	6.30
	MaskFlowNet[7]	-	-	-	-	2.52	4.17	6.10
	RAFT[1]	(0.76)	(1.22)	(0.63)	(1.5)	1.94	**3.18**	5.10
	DEQ-RAFT	(0.70)	(1.21)	(0.61)	(1.4)	**1.82**	3.23	**4.98**

Table 1. **Evaluation on Sintel and KITTI 2015 datasets.** We report the Average End Point Error (AEPE) and the F1-all measure for the KITTI dataset (lower is better). "C+T" refers to results that are pre-trained on the Chairs and Things datasets. "S+K+H" refers to methods that are fine-tuned on the Sintel, KITTI, and HD1K datasets. The bold font stands for the best result and the underlined results ranks 2nd.



(a) Training DEQ with Anderson forward solver

(b) Training DEQ with Broyden forward solver

Figure 4. Performance and convergence stability (measured by absolute residual error) of the DEQ flow. Frequency indicates how many correction terms we pick, with 0 meaning no correction. See the comparison with Jacobian Regularization [26] in the Appendix. DEQ flows trained with our proposed correction enjoy superior performance and stability. In practice, we find 1 correction loss already suffices.

is the frequency in Fig. 4) and impose a correction loss after each trajectory clip. As mentioned in Sec. 3.3, we apply the one-step gradient [21, 22] to the correction loss.

We visualize results of DEQ flow models trained with 3 different settings: 1) a DEQ flow trained by IFT directly without an auxiliary correction loss; 2) a DEQ flow trained by inexact gradient without an auxiliary correction loss; and 3) DEQ flows trained by inexact gradient *as well as* 1-3 fixed-point correction terms. Our results are reported in terms of AEPE (which measures performance) and absolute fixed-point residual error $\|f_\theta(\mathbf{z}^*;\mathbf{x})-\mathbf{z}^*\|_2$ (which measures stability). As shown in Fig. 4, our proposed fixed-point correction significantly outperforms the standard IFT training protocol by about 9%, and reduces the fixed-point error by a conspicuous margin, *e.g.,* over 60%. Moreover, we find that this significant improvement in stability quickly diminishes as we apply > 1 corrections; therefore in practice, we usually use one correction term. Together with the inexact gradient, the total training time can be streamlined over 45%, while the backward pass of a DEQ flow is still almost *free*.

Correlation between Performance and Convergence. A potential question is whether better fixed-point convergence

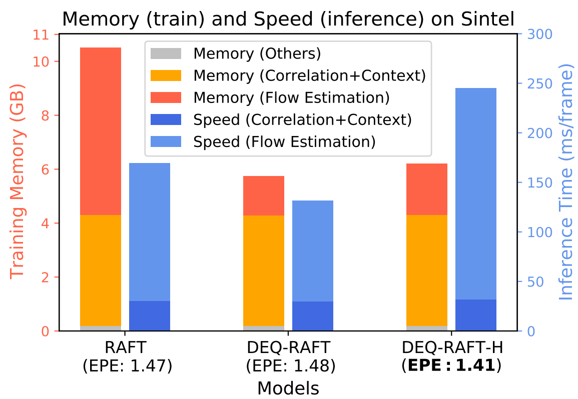


Figure 5. Comparing the training memory, inference speed and performance on Sintel (clean) with image size 436×1024. The same model design (based on RAFT) consumes much less memory and computes much quicker than the recurrent counterpart. All results are benchmarked on a single Quadro RTX 8000 GPU.

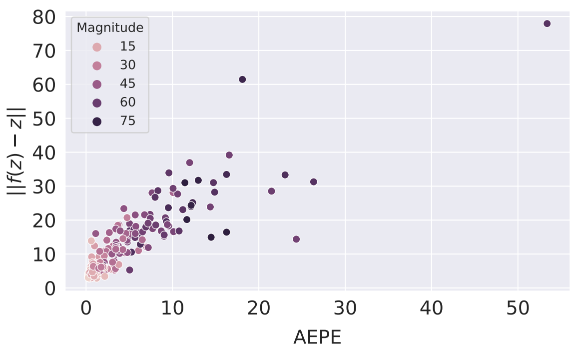


Figure 6. Correlation between convergence and performance. We also observe that harder examples (e.g., those with large motion) typically lead to more challenging fixed-point convergence.

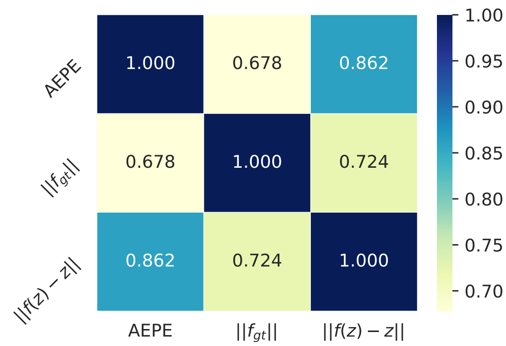


Figure 7. Pearson correlation coefficient across per-frame EPE, fixed-point error, and the magnitude of flow.

can lead to better performance. To tackle this, we evaluate the DEQ flow model trained using the standard "C+T" training protocol (see Sec. 4.1) on the KITTI-15 [27] training dataset. We visualize the per-frame EPE and the convergence measured by the absolute fixed point error in Sec. 4.3 and dye the scatter plot with the average norm of per-pixel flow across the frame, which can be understood as an indicator of hardness due to the large displacements. The Pearson correlation coefficient between the fixed-point error and EPE is over **0.86** (see Fig. 7) supporting the claim that convergence is strongly correlated with the flow performance. From Fig. 7, we see that hard flows, with large motions are also challenging for a naive solver. This demonstrates the necessity of advanced solvers in DEQ flow estimation.

5. Limitations

The improved performance and efficiency of our approach comes at the cost of a slightly more complex training pipeline. Implementing the naïve unrolled flow estimation as presented in Teed and Deng [1] and Jiang et al. [2] is simple using most libraries equipped with automatic differentiation that directly handle BPTT. On the other hand, our approach, albeit much cheaper, does involve some finagling of the training protocol (*e.g.*, fixed-point solvers, IFT, inexact gradients, etc.). To help alleviate this complexity and promote the use of DEQ-flows, we release our code at `https://github.com/locuslab/deq-flow`.

In addition, while DEQ flows provide a novel and more efficient framework to train and use these flow estimators, we still occasionally need to be careful about the stability of this approach. For example, what would happen if the solver *converges poorly* (or even diverges) on a dynamical system? In such case, the behavior of the DEQ flow estimation would not be well-defined. In practice, we rarely observe such instability (as long as we spend enough solver steps); but as we analyzed in Sec. 4, harder examples also typically lead to more lengthy convergence path. We leave a more thorough study of the flow estimation stability to future work, while noting that recent progress on using neural solvers might suggest an interesting solution to this issue [57].

6. Conclusion

In this work, we introduce a new framework for modeling optical flow estimation. A deep equilibrium (DEQ) flow directly models and solves a fixed-point *stable* flow estimate, and offers a set of tools that make these flow models' training and inference process highly efficient (e.g., they enjoy an almost-free backward pass). Moreover, the use of such equilibrium formulation is largely orthogonal to, and thus complements, the prior modeling- and data-related efforts. We empirically show that it is possible to integrate the DEQ flow estimator with these model designs (like RAFT [1]) and achieve better performance on realistic optical flow datasets.

We therefore argue that this implicit framework provides a strong (drop-in) replacement for existing recurrent, unrolled update operators used by most cutting-edge flow estimators. The DEQ flows are both more performant and lightweight — both computationally and memory-wise. We believe this suggests an exciting direction for building more efficient, large-scale and accurate optical flow models in the future.

References

[1] 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, 3, 4, 5, 6, 7, 8

[2] 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, 3, 4, 6, 7, 8

[3] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. Deep equilibrium models. In *Neural Information Processing Systems (NeurIPS)*, 2019. 1, 2, 3, 4, 5

[4] Bruce D Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision. 1981. 1

[5] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981. 2, 3, 4

[6] 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. 1, 2, 3, 5, 6

[7] Shengyu Zhao, Yilun Sheng, Yue Dong, Eric I Chang, Yan Xu, et al. Maskflownet: Asymmetric feature matching with learnable occlusion mask. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6278–6287, 2020. 1, 7

[8] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. A lightweight optical flow cnn–revisiting data fidelity and regularization. *arXiv preprint arXiv:1903.07414*, 2019. 1, 2, 7

[9] Kyunghyun Cho, B van Merrienboer, Caglar Gulcehre, F Bougares, H Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, 2014. 1

[10] Deqing Sun, Daniel Vlasic, Charles Herrmann, V. Jampani, Michael Krainin, Huiwen Chang, Ramin Zabih, William T. Freeman, and Ce Liu. Autoflow: Learning a better training set for optical flow. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10088–10097, 2021. 1, 2

[11] Abdelrahman Eldesokey and Michael Felsberg. Normalized convolution upsampling for refined optical flow estimation. *arXiv preprint arXiv:2102.06979*, 2021.

[12] 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. 1

[13] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990. 1

[14] Shaojie Bai, Vladlen Koltun, and J. Zico Kolter. Multiscale Deep Equilibrium Models. In *Neural Information Processing Systems (NeurIPS)*, pages 5238–5250, 2020. 2, 3, 4

[15] Ezra Winston and J. Zico Kolter. Monotone operator equilibrium networks. In *Neural Information Processing Systems (NeurIPS)*, pages 10718–10728, 2020. 2, 3, 5

[16] Kenji Kawaguchi. On the Theory of Implicit Deep Learning: Global Convergence with Implicit Layers. In *International Conference on Learning Representations (ICLR)*, 2020. 3

[17] J. Zico Kolter, David Duvenaud, and Matthew Johnson. Deep implicit layers tutorial - neural ODEs, deep equilibirum models, and beyond. *Neural Information Processing Systems Tutorial*, 2020. 2

[18] Charles G Broyden. A Class of Methods for Solving Nonlinear Simultaneous Equations. *Mathematics of computation*, 19 (92):577–593, 1965. 2, 3, 6

[19] Donald G. Anderson. Iterative procedures for nonlinear integral equations. *Journal of the ACM (JACM)*, 12(4):547–560, October 1965. 2, 3, 6

[20] David Fleet and Yair Weiss. Optical flow estimation. In *Handbook of mathematical models in computer vision*, pages 237–257. Springer, 2006. 2

[21] Zhengyang Geng, Meng-Hao Guo, Hongxu Chen, Xia Li, Ke Wei, and Zhouchen Lin. Is Attention Better Than Matrix Decomposition? In *International Conference on Learning Representations (ICLR)*, 2021. 2, 4, 5, 7

[22] Samy Wu Fung, Howard Heaton, Qiuwei Li, Daniel McKenzie, Stanley J. Osher, and Wotao Yin. Fixed Point Networks: Implicit Depth Models with Jacobian-Free Backprop. *arXiv preprint arXiv:2103.12803*, 2021. 4, 5, 7

[23] Zhengyang Geng, Xin-Yu Zhang, Shaojie Bai, Yisen Wang, and Zhouchen Lin. On training implicit models. *ArXiv*, abs/2111.05177, 2021. 2, 4, 5

[24] Filippo Aleotti, Matteo Poggi, and Stefano Mattoccia. Learning optical flow from still images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15201–15211, 2021. 2, 3

[25] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Neural Information Processing Systems (NeurIPS)*, 2018. 2, 3, 5, 6

[26] Shaojie Bai, Vladlen Koltun, and J. Zico Kolter. Stabilizing Equilibrium Models by Jacobian Regularization. In *International Conference on Machine Learning (ICML)*, 2021. 2, 3, 5, 7

[27] 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, 6, 8

[28] Michael J Black and Padmanabhan Anandan. A framework for the robust estimation of optical flow. In *1993 (4th) International Conference on Computer Vision*, pages 231–236. IEEE, 1993. 2

[29] Christopher Zach, Thomas Pock, and Horst Bischof. A duality based approach for realtime tv-l1 optical flow. In *DAGM-Symposium*, 2007.

[30] Andreas Wedel, Thomas Pock, Christopher Zach, Horst Bischof, and Daniel Cremers. An improved algorithm for tv-l 1 optical flow. In *Statistical and Geometrical Approaches to Visual Motion Analysis*, 2008.

[31] Thomas Brox and Jitendra Malik. Large displacement optical flow: Descriptor matching in variational motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33:500–513, 2011. 2

[32] Moritz Menze, Christian Heipke, and Andreas Geiger. Discrete optimization for optical flow. In *GCPR*, 2015. 2

[33] Qifeng Chen and Vladlen Koltun. Full flow: Optical flow estimation by global optimization over regular grids. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4706–4714, 2016. 2

[34] Jia Xu, René Ranftl, and Vladlen Koltun. Accurate optical flow via direct cost volume processing. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5807–5815, 2017. 2

[35] 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. 2, 7

[36] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Models matter, so does training: An empirical study of cnns for optical flow estimation. *arXiv preprint arXiv:1809.05571*, 2018. 7

[37] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. Liteflownet: A lightweight convolutional neural network for optical flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8981–8989, 2018. 7

[38] 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. 2, 7

[39] Gengshan Yang and Deva Ramanan. Volumetric correspondence networks for optical flow. In *Advances in Neural Information Processing Systems*, pages 793–803, 2019. 2, 7

[40] 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. 2

[41] Shihao Jiang, Yao Lu, Hongdong Li, and Richard Hartley. Learning optical flow from a few matches. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16592–16600, 2021. 2

[42] Brandon Amos and J. Zico Kolter. OptNet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning (ICML)*, 2017. 2

[43] Laurent El Ghaoui, Fangda Gu, Bertrand Travacca, and Armin Askari. Implicit deep learning. *arXiv:1908.06315*, 2019. 2

[44] Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented neural ODEs. In *Neural Information Processing Systems (NeurIPS)*, 2019. 3, 5

[45] Cheng Lu, Jianfei Chen, Chongxuan Li, Qiuhao Wang, and Jun Zhu. Implicit normalizing flows. In *International Conference on Learning Representations (ICLR)*, 2021. 3

[46] 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. 4

[47] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 4

[48] Fangda Gu, Heng Chang, Wenwu Zhu, Somayeh Sojoudi, and Laurent El Ghaoui. Implicit graph neural networks. *arXiv preprint arXiv:2009.06211*, 2020. 4

[49] Junyoung Park, Jinhyun Choo, and Jinkyoo Park. Convergent graph solvers. *arXiv preprint arXiv:2106.01680*, 2021.

[50] Juncheng Liu, Kenji Kawaguchi, Bryan Hooi, Yiwei Wang, and Xiaokui Xiao. EIGNN: Efficient infinite-depth graph neural networks. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. 4

[51] Steven G Krantz and Harold R Parks. *The implicit function theorem: History, theory, and applications*. Springer, 2012. 4

[52] Jacob Kelly, Jesse Bettencourt, Matthew James Johnson, and David Duvenaud. Learning differential equations that are easy to solve. In *Neural Information Processing Systems (NeurIPS)*, 2020. 5

[53] 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. 5

[54] 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. 6

[55] 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. 6

[56] Daniel Kondermann, Rahul Nair, Katrin Honauer, Karsten Krispin, Jonas Andrulis, Alexander Brock, Burkhard Gussefeld, Mohsen Rahimimoghaddam, Sabine Hofmann, Claus Brenner, et al. The hci benchmark suite: Stereo and flow ground truth with uncertainties for urban autonomous driving. *CVPR Workshop*, 2016. 6

[57] Shaojie Bai, Vladlen Koltun, and J Zico Kolter. Neural deep equilibrium solvers. In *International Conference on Learning Representations*, 2022. 8