DiffPoseNet: Direct Differentiable Camera Pose Estimation

Chethan M. Parameshwara, Gokul Hari, Cornelia Fermüller, Nitin J. Sanket, Yiannis Aloimonos
University of Maryland, College Park
College Park, MD
{cmparam9,hgokul,fermulcm,nitinsan,jyaloimo}@umd.edu

Abstract

Current deep neural network approaches for camera pose estimation rely on scene structure for 3D motion estimation, but this decreases the robustness and thereby makes cross-dataset generalization difficult. In contrast, classical approaches to structure from motion estimate 3D motion utilizing optical flow and then compute depth. Their accuracy, however, depends strongly on the quality of the optical flow. To avoid this issue, direct methods have been proposed, which separate 3D motion from depth estimation, but compute 3D motion using only image gradients in the form of normal flow. In this paper, we introduce a network NFlowNet, for normal flow estimation which is used to enforce robust and direct constraints. In particular, normal flow is used to estimate relative camera pose based on the cheirality (depth positivity) constraint. We achieve this by formulating the optimization problem as a differentiable cheirality layer, which allows for end-to-end learning of camera pose. We perform extensive qualitative and quantitative evaluation of the proposed DiffPoseNet's sensitivity to noise and its generalization across datasets. We compare our approach to existing state-of-the-art methods on KITTI, TartanAir, and TUM-RGBD datasets.

1. Introduction

The ability to localize is imperative for applications in mobile robotics, and solutions based on vision are often the preferred choice because of size, weight, power constraints and the availability of robust localization methods. Many mathematical frameworks and deep learning approaches have been developed for the problem of visual localization [12, 38] under the umbrella of Visual Odometry (VO) or Simultaneous Localization and Mapping (SLAM). However, their performance is subpar for commonly encountered challenging conditions in-the-wild that involve changing lighting, scenes with textureless regions, and dynamic objects.

Classical approaches [5, 8, 9, 24] for localization rely either on sparse feature correspondences between images or on the computation of dense motion fields (optical flow). One of the difficulties in optical flow estimation is bias due to noise [15, 16]. For example, if in a patch there are more gradients in one direction than another, their estimated optical flow will be biased towards the dominant direction. Even though over the past decade many learning-based approaches have proposed to improve optical flow estimation, this behavior still persists in optical flow approaches. This is demonstrated in Fig. 1, which shows the errors produced by the normal flow algorithm presented in this paper in comparison to three optical flow algorithms from the literature. As can be seen, all flow algorithms have large errors in regions of non-uniform gradient distributions.

To this end, the pioneers of the field remarked on the observation that the projection of optical flow on the image gradient direction is resilient to the bias and can be computed robustly. This projection is called the *normal flow*. Over the past few decades, a number of methods have been proposed that use the spatial gradient directly for 3D motion recovery. These methods are commonly called direct methods. In principle, such methods are robust and computationally cheaper than flow-based feature based approaches as they use the image brightness directly. However, despite the advantages of the normal flow formulation, the computational methods to estimate normal flow have not been robust enough to allow deployment in the wild. Thus, optical flow has been the go-to representation for ego-motion estimation, supported in recent years by the high accuracy and speed of deep learning algorithms [22, 26, 35]. To improve the robustness of camera pose estimation (ego-motion), we propose the first normal flow network *NFlowNet*.

Further, to estimate pose independent of scene structure from normal flow, direct approaches utilize minimal constraints. When optical flow or correspondence are available, pose is estimated using the depth-independent epipolar constraint. However different from these 2D measurements, normal flow is 1D and thus depth cannot be eliminated from the equations relating it to scene geometry and 3D motion. Not making assumptions about the scene structure, the only

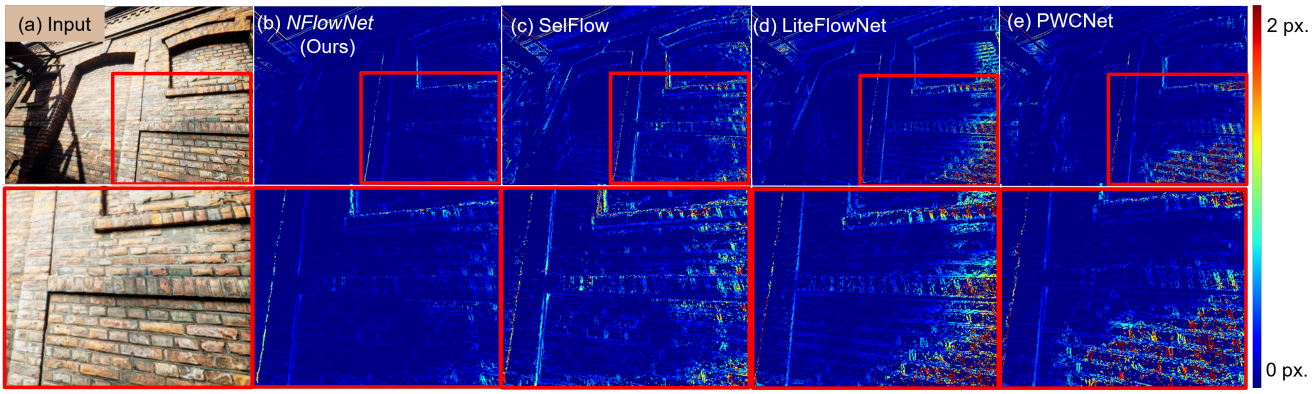


Figure 1. Top Row: Endpoint error map of *NFlowNet* compared to three different optical flow approaches (SelFlow [30], LiteFlowNet [21], and PWC-Net [43]). Bottom Row: enlarged endpoint error map of the region highlighted in red in the top row. The optical flow is due to the camera undergoing translation parallel to the wall. It has large errors in regions of non-uniform gradient directions. Notably, on the bricks of rectangular shape, there are many more (vertical) gradients due to the horizontal edges than (horizontal) gradients due to vertical edges, and this causes erroneous flow estimation. Similarly, on the edges of the niches, where there is only one gradient direction, there is error. *All the images in this paper are best viewed in color on a computer screen at a zoom of 200%.*

constraint that can be imposed on the scene, is the depth positivity [34] or cheirality constraint [49]. Cheirality states that the scene has to be in front of the camera for it to be imaged, and thus the depth has to be positive. This constraint when enforced on normal flow can be utilized to estimate camera pose without making assumptions on scene depth or shape. Since the cheirality condition is an inequality constraint and hence is not differentiable, until recently it was not possible to employ it in a deep learning pipeline. To this end, we utilize the differentiable programming paradigm [2] implemented with the implicit differentiation [18] framework to reformulate the cheirality optimization into a differentiable layer and hence train our pose network in an end-to-end fashion.

In this work, we design a novel normal flow network *NFlowNet* and couple it with a differentiable cheirality layer for robust pose estimation. Our contributions (in the order for ease of understanding) can be summarized as follows:

- We introduce a network *NFlowNet* to estimate normal flow. This estimated robust normal flow, beyond this paper, is useful for applications requiring computationally efficient solutions for navigation tasks in computer vision and robotics.
- We formulate the estimation of pose from normal flow using the cheirality (or depth positivity) constraint as a differentiable optimization layer.
- Extensive qualitative and quantitative experimental results highlighting the robustness and cross-dataset generalizability of our approach without any fine-tuning and/or re-training.

2. Related Work

2.1. Normal Flow and Camera Pose

Several works have developed direct methods that use normal flow for pose estimation. The idea is that normal flow can be interpreted as "the projection of optical flow on the gradient direction." Thus, given a normal flow vector, the optical flow is constrained to a half-plane [1]. If the 3D motion is only due to translation, this constrains the focus of expansion, i.e., the intersection of the translation axis with the image plane [34] to a half plane. Based on this concept, [20] proposed different algorithms to solve for the case of translation only, and [40] analyzed the method's stability in the presence of small rotations. Modeling the scene as piece-wise planar, [7] solved for 3D motion and calibration [6], and [23] added constraints for combining multiple flow fields. Not making depth assumptions, [14] developed constraints on the sign of normal flow, which geometrically separate the rotational and translational flow components and can be implemented as pattern matching. Others proposed techniques for separating 3D motion components by searching for lines in the image, where certain 3D motion components cancel out [39, 51]. Recently [4] modeled the cheirality constraint by approximating it with a smooth function, which allowed the use of modern optimization techniques. The method first solves for 3D motion from normal flow, and then refines using a regularization defined on depth. Experimental results demonstrate that the proposed pipeline outperforms other flow based approaches. Inspired by these findings, we follow a similar pipeline, but we develop the constraints within a neural network approach.

2.2. Learning-based Camera Pose Estimation

Early studies of learning-based camera pose (VO) models [44] [28] [27] were mainly focused on supervised learning approaches modelled as either absolute pose/relative pose regression problems. However, these methods require real-world ground truth poses which is often difficult to obtain. In order to alleviate the need of ground truth data, self-supervised VO was proposed. SfMLearner [52] learns depth and pose simultaneously by minimizing photometric loss between warped and input image. [50] and [53] extend this idea to joint estimation of pose, depth and optical flow. Learning-based models suffer from generalization issues when tested on images from a new environment. Most of the VO models are trained and tested on the same dataset. Most recently, TartanVO [45] addresses generalization issues by incorporating the camera intrinsics directly into the model and training with a large amount of data. Recent advances in differentiable optimization layers (or differentiable programming) [2, 18] has enabled a new generation of generalizable pose learning approaches. [25] embeds an Epipolar geometric constraint into a self-supervised learning framework via bi-level optimization of camera pose and optical flow. BlindPnP [11] embeds geometric model fitting algorithms (PnP algorithm, RANSAC) into implicit differentiation layers. All the above work focus on reconstructing the structure (either through network or optimization) and/or optical flow correspondences for estimating camera motion. In our work, we address robust pose estimation by depending only on structure-less cheirality constraints and normal flow. Finally, we utilize the concepts of best of both-worlds to enable speedup using data prior and novel data generalization from mathematical optimization.

3. Overview of Proposed Approach

The network architecture is illustrated in Fig. 2. It consists of *NFlowNet*, a network for estimating normal flow (Section 4), which then is used for self-adaptive camera pose estimation (or odometry estimation) (Section 5). The camera pose estimation proceeds in three steps. First we *initialize* the PoseNet using supervised training with successive images as input (Sec. 5.1). Next, pose is estimated using differentiable optimization by embedding a cheirality constraint defined on normal flow (Sec. 5.2). This way, simliar to the classic SfM approach, pose is *estimated independent of depth*. In a last step, the pose in PoseNet is refined through a self-supervised refinement loss by using the pose estimates from the *cheirality constraint* to minimize the error in normal flow.

4. *NFlowNet* for Normal Flow Prediction

The first step in motion analysis from video is to compute an image motion representation. Most approaches either detect and track distinct features or compute gradient-based optical flow. The latter is estimated by assuming the intensity $I_{\mathbf{x}}$ (or a function of the intensity) at a point $\mathbf{x} = \begin{bmatrix} x & y \end{bmatrix}^T$ to remain constant over a short time interval δt. This is referred to as the brightness constancy constraint [49]:

$$I(x, y, t) = I(x + u\delta t, y + v\delta t, t + \delta t) \tag{1}$$

Here u and v refers to the image pixel motion. Approximating Eq.(1) with a first order Taylor expansion, we obtain Eq.(2):

$$\frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v = -\frac{\partial I}{\partial t} \tag{2}$$

We call the component of the flow along the gradient direction, the *normal flow*. Denoting the spatial gradients as $\nabla I = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right)$ and the flow as $\mathbf{u} = \begin{bmatrix} u & v \end{bmatrix}^T$, the normal flow vector $\mathfrak{n}$ (a 2d vector) is defined as:

$$\mathfrak{n} = \frac{(\mathbf{u} \cdot \nabla I)}{||\nabla I||^2} \nabla I \tag{3}$$

Using the brightness constancy constraint (eq. 1), the normal can be computed directly from the spatial and temporal image derivatives, I_t, as $\mathfrak{n} = \frac{-I_t}{||\nabla I||^2} \nabla I$. Since this constraint alone is not sufficient to determine two-dimensional image motion, additional constraints need to be introduced. Traditionally, variational methods combining multiple global smoothness assumptions, have been the dominant approach for optical flow estimation, and more recently they have been replaced by deep learning algorithms. However, these methods tend to perform subpar on regions with little features or repeated texture due to the strong reliance on the training dataset, and in boundary regions due to oversmoothing, especially when the size of the flow varies significantly in the image. Computing normal flow solely on spatio-temporal derivatives, is unreliable and prone to errors. Hence, we propose a novel Normal flow network called *NFlowNet*.

We use an encoder-decoder convolutional neural network, which we train in a supervised way. Given an image pair, normal flow describes the pixel motion parallel to the image derivatives. To learn normal flow, we utilize the TartanAir dataset. Specifically, we utilize Eq. 3 to compute ground-truth normal flow. We train the*NFlowNet* supervised using the l_2 loss between our network predictions $\tilde{\mathbf{n}}$ and ground truth $\hat{\mathbf{n}}$, i.e.,

$$\underset{\tilde{\mathfrak{n}}}{\operatorname{argmin}} \|\hat{\mathfrak{n}} - \tilde{\mathfrak{n}}\|_2 \tag{4}$$

In the experimental section (Sec. 7), we show that *NFlowNet* generalizes to the real-world and other datasets without any fine-tuning or re-training.

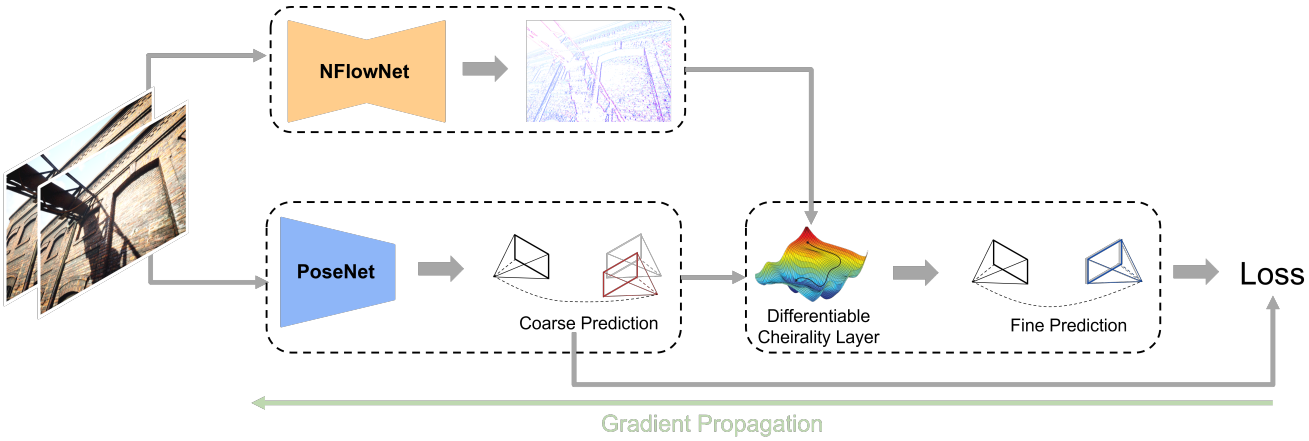


Figure 2. Overview of our proposed *DiffPoseNet* framework. Our network starts with a novel normal flow estimation network *NFlowNet* and a first coarse pose estimate. Next, fine pose is estimated using the proposed differentiable cheirality layer.

5. Self-Adaptive Pose Estimation from Normal Flow

We use a deep network to regress relative poses, that is, the 3D rigid motion of the camera between subsequent time steps and denoted as $\mathbf{P}_t^{t+1}$. The conversions between absolute poses and relative poses is explained next.

If the absolute pose at time t is given by $\begin{bmatrix}\mathbf{T}_t & R_t\end{bmatrix}^T$, where $\mathbf{T}_t \in \mathbb{R}^{3\times 1}$ and $R_t \in SO(3)$. The relative pose between t and $t+1$ under a linear velocity assumption is denoted as $\begin{bmatrix}\mathbf{V} & \mathbf{\Omega}\end{bmatrix}^T$ and is given by

$$\mathbf{V} = \frac{\mathbf{T}_{t+1} - \mathbf{T}_t}{dt}; \quad \mathbf{\Omega}_\times = \frac{\text{logm}\left(R_t^T R_{t+1}\right)}{dt} \tag{5}$$

Here, dt is the time increment between t and $t+1$, logm is the matrix logarithm operator and $\mathbf{\Omega}_\times$ converts the vector $\mathbf{\Omega}$ into the corresponding skew symmetric matrix

$$\mathbf{\Omega}_\times = \begin{bmatrix} 0 & -\Omega_z & \Omega_y \\ \Omega_z & 0 & -\Omega_x \\ -\Omega_y & \Omega_x & 0 \end{bmatrix} \tag{6}$$

5.1. PoseNet for Initializing Pose Estimation

In the first stage, we learn coarse relative poses using a CNN+LSTM. The feature representations learned by the CNN layers are passed to the LSTM for sequential modelling. We use a supervised l_2 loss between the ground truth ($\widehat{\mathbf{P}} = \begin{bmatrix}\widehat{\mathbf{V}} & \widehat{\mathbf{\Omega}}\end{bmatrix}^T$) and predicted poses ($\widetilde{\mathbf{P}} = \begin{bmatrix}\widetilde{\mathbf{V}} & \widetilde{\mathbf{\Omega}}\end{bmatrix}^T$). Here, $\mathbf{V}$ and $\mathbf{\Omega}$ represent the translational and rotational parts of the pose. The orientation is represented in $X-Y-Z$ Euler Angles. Denoting as λ a weighting parameter, we solve the following optimization using backpropagation:

$$\underset{\widetilde{\mathbf{P}}}{\text{argmin}} \left(\|\widehat{\mathbf{V}} - \widetilde{\mathbf{V}}\|_2^2 + \lambda \|\widehat{\mathbf{\Omega}} - \widetilde{\mathbf{\Omega}}\|_2^2 \right) \tag{7}$$

5.2. Differentiable Cheirality Layer for Fine Pose Estimation

To enable self-supervised learning of continuous pose (given a initialization value), we propose to utilize the cheirality constraint or depth positivity constraint, which states that all world points have to be in front of the camera, i.e., have positive depth. This condition has been classically used in structure from motion problems to disambiguate the physically possible camera poses from the set of computed solutions. The main reason for utilizing the cheirality rather than the acclaimed epipolar constaint or scene planarity constraint is due to the minimalism in the assumptions. Since in our formulation, depth positivity is enforced using the normal flow, which in-turn makes minimal assumptions about the scene structure, our formulation generalizes to novel scenes with remarkable accuracy (Sec. 6.2.2).

Let us mathematically define the constraints. Denoting the magnitude of the normal flow (a scalar) at pixel $\mathbf{x}$ as $n_\mathbf{x}$ and the direction of the image gradient as $\mathbf{g}_\mathbf{x}$ (a unit vector), we have

$$n_\mathbf{x} = \|\mathbf{n}\|_2 = \frac{1}{Z_\mathbf{x}}(\mathbf{g}_\mathbf{x} \cdot A)\mathbf{V} + (\mathbf{g}_\mathbf{x} \cdot B)\mathbf{\Omega}, \tag{8}$$

where, $\mathbf{V}$ denotes the constant translational velocity and $\mathbf{\Omega}$ the rotational velocity $\mathbf{\Omega}$ in a small time instant, and $Z_\mathbf{x}$ is the depth of the point under consideration. Intuitively, $\mathbf{V}, \mathbf{\Omega}$ warp the flow field and $Z_\mathbf{x}$ scales the flow field, and the matrices A and B determine how the motion flow field is projected onto the image plane due to translational and rotational velocities respectively and are given by

$$A = \begin{bmatrix} -1 & 0 & x \\ 0 & -1 & y \end{bmatrix} \tag{9}$$

$$B = \begin{bmatrix} xy & -(x^2+1) & y \\ (y^2+1) & -xy & -x \end{bmatrix} \tag{10}$$

Let us consider in eq. (8) the two components of normal flow: the translational component, which depends on depth, and the rotational component, which is independent of depth. If we subtract the rotational component from both sides, we obtain

$$n_{\mathbf{x}} - (\mathbf{g}_{\mathbf{x}} \cdot B)\mathbf{\Omega} = \frac{1}{Z_{\mathbf{x}}}(\mathbf{g}_{\mathbf{x}} \cdot A)\mathbf{V}. \tag{11}$$

We can enforce that the left hand side (the derotated normal flow) and the right hand side (the translational component) must have same sign. Since the depth ($Z_{\mathbf{x}}$) is positive, the following product, which we denote as $\rho_{\mathbf{x}}(\mathbf{V}, \mathbf{\Omega})$, is positive, i.e.,

$$\rho_{\mathbf{x}}(\mathbf{V}, \mathbf{\Omega}) = ((\mathbf{g}_{\mathbf{x}} \cdot A)\,\mathbf{V}) \cdot (n_{\mathbf{x}} - (\mathbf{g}_{\mathbf{x}} \cdot B)\,\mathbf{\Omega}) > 0 \tag{12}$$

To arrive at an objective function to be used in an optimization, we can model the cheirality constraint by passing $\rho_{\mathbf{x}}$ through a smooth function, such as the ReLU function [4]. Since, the deep learning pipeline requires the function to be twice differentiable, we choose the GELU function, a smooth approximation of the ReLu function. Denoting the negative GELU function as $\mathcal{R}$, and denoting the average over all $\mathbf{x}$ values as $\mathbb{E}$, we then obtain the following minimization, for the estimation of relative camera pose:

$$\underset{\{\mathbf{V},\mathbf{\Omega}\}}{\operatorname{argmin}}\, \mathbb{E}\left(\mathcal{R}(\rho_{\mathbf{x}}(\mathbf{V}, \mathbf{\Omega}))\right) \tag{13}$$

In the stage of re-estimating motion, we simply use constraint (13) in an optimization implemented by a robust Quasi-Newton optimization algorithm. We solve the optimization sequentially, in one step for V and in the other for Ω, because $\rho_{\mathbf{x}}(\mathbf{V}, \mathbf{\Omega})$ is bilinear in these parameters. The initial estimate comes from the PoseNet estimate in Sec. 5.1. In our implementation we use the L-BFGS algorithm [55]. These steps (Sec. 5.1 and 5.2) are performed in the forward pass in the network.

5.3. Self-Supervised Refinement

Let us denote the coarse pose obtained from our *PoseNet* as $\widetilde{\mathbf{P}}_c = \begin{bmatrix} \widetilde{\mathbf{V}}_c & \widetilde{\mathbf{\Omega}}_c \end{bmatrix}$. This is further refined by the Cheirality layer and we denote this refined pose as $\widetilde{\mathbf{P}}_r = \begin{bmatrix} \widetilde{\mathbf{V}}_r & \widetilde{\mathbf{\Omega}}_r \end{bmatrix}$. Now we use $\widetilde{\mathbf{P}}_r$ to refine our *PoseNet*'s coarse pose to obtain a more accurate prediction of pose.

The final self-adaptive pose estimation is performed as a bi-level minimization in the network [18], in which an upper-level problem is solved subject to constraints imposed by a lower-level problem and is formally defined next.

$$\begin{aligned} &\underset{\widetilde{\mathbf{P}}_c}{\operatorname{argmin}}\, \mathbb{E}\left(n_{\mathbf{x}} - \mathbf{g}_{\mathbf{x}} \cdot \left(\left(\frac{n_{\mathbf{x}} - (\mathbf{g}_{\mathbf{x}} \cdot B)\widetilde{\mathbf{\Omega}}_r}{(\mathbf{g}_{\mathbf{x}} \cdot A)\widetilde{\mathbf{V}}_r}\right) A\widetilde{\mathbf{V}}_c - B\widetilde{\mathbf{\Omega}}_c\right)\right) \\ &\text{subject to} \quad \underset{\widetilde{\mathbf{P}}_r}{\operatorname{argmin}}\, \mathbb{E}\left(\mathcal{R}(\rho_{\mathbf{x}}(\widetilde{\mathbf{V}}_c, \widetilde{\mathbf{\Omega}}_c))\right) \end{aligned} \tag{14}$$

The lower-level problem (second row) of Eq. (14) enforces the cheirality constraint to obtain pose $\widetilde{\mathbf{P}}_r$, which then is used to compute the normal flow error in the upper-level loss function (first row). The upper-level loss enforces the consistency between the normal flow from *NFlowNet* and that computed using the motion parameters $\widetilde{\mathbf{P}}_c$ with the implicit depth term expressed by the motion parameters $\widetilde{\mathbf{P}}_r$.

In practice, we back-propagate through the upper-level to refine poses $\widetilde{\mathbf{P}}_{\mathbf{c}}$ using supervision from $\widetilde{\mathbf{P}}_{\mathbf{r}}$ through the lower-level. Using implicit differentiation, all that is computed from the lower layer is the gradient, and this step is agnostic to the optimizer used. Specifically, we derive $\frac{\partial \widetilde{\mathbf{P}}_r}{\partial \widetilde{\mathbf{P}}_c}$, which is computed from the product of second order derivatives. (The interested reader is referred to [18], eq. (15) for details.) It is important to note that we rely on the generalizability of *NFlowNet*, hence it is not fine-tuned.

6. Experiments

6.1. Implementation Details

6.1.1 Datasets

We use eight environments from the TartanAir [46] dataset (`amusement`, `oldtown`, `neighborhood`, `soulcity`, `japanesealley`, `office`, `office2`, `seasidetown`) for training and two environments (`abandonedfactory` and `hospital`) for testing our *NFlownet* network. For odometry evaluation, we use the Tartan challenge test data [46]. We also conduct extensive experiments on the KITTI Odometry [17] and the TUM-RGBD [42] datasets to evaluate the robustness and generalization performance of our proposed system.

6.1.2 Networks and Optimization Layer

For the *NFlowNet* we use an encoder-decoder architecture based on EVPropNet [37] to directly regress sparse normal flow. The encoder contains residual blocks with convolutional layers and the decoder contains residual blocks with transpose convolutional layers. We choose the number of residual and transposed residual blocks as 2 and the expansion factor (factor by which the number of neurons are increased after every block) as 2. We backpropagate the gradients using a mean squared loss computed between

groundtruth and predicted normal flow as given in Eq. 4. We used the Adam optimizer to train our network with a learning rate of 10^{-4} and batch size of 8 for 400 epochs.

Our *PoseNet* architecture is inspired from [44] and uses the VGG-16 encoder for the CNN stage [13] and two LSTM layers each with 250 hidden units for the recurrent layer-stage. We initially train this model with a subset of the TartanAir data for 30 epochs to obtain a coarse estimate to initialise the Cheirality Layer. We use the Adam optimizer and set a fixed learning rate of 10^{-5}. We consider sequences of six consecutive image frames and a batch size of eight while training. During test time, we use only two consecutive image frames to estimate the relative camera pose between images I_t and I_{t+1}.

For the optimization layer we use the L-BFGS [10] solver. The line search function was set to strong Wolfe [48], the number of iterations were set to 100, and the gradient norms were clipped to 100. We initialised the optimizer with coarse predictions provided by our *PoseNet*. Our overall system is implemented in Python 3.7 and PyTorch 1.9.

6.1.3 Training and Testing Procedure

The whole training schedule consists of three stages. First, we only train *NFlowNet* and *PoseNet* in a supervised manner using the training strategies mentioned above. Then we freeze *NFlowNet* and jointly train the *PoseNet* with the cheirality layer in an self-supervised fashion via the refinement loss. The self supervised training is carried out for 120 epochs using four Nvidia P6000 GPUs. For the cheirality layer, the stopping criteria is when the objective function is below 10^{-20} or the number of iterations exceeds 300.

During testing, our final pose predictions are obtained by passing *PoseNet* priors through the cheirality layer along with the predictions from *NFlowNet*. Due to our self-supervised refinement training, the prior *PoseNet* estimates help in faster convergence of the cheirality optimization process (takes less than 5 iterations). Overall, NFlowNet takes 15 ms and the coarse PoseNet takes 8 ms. The Cheirality layer takes an average of 8 ms per iteration to refine the estimates. The inference time is obtained for an image resolution of 320×640 using a Nvidia 2070 MaxQ GPU.

6.1.4 Evaluation Metrics

To evaluate our *NFlowNet* and to compare against other optical flow networks, we project the optical flow obtained from the state-of-the-art approaches on the gradient direction, and we measure the pixel error. As normal flow is defined only along the gradient direction, we utilize the Projection Endpoint Error (PEE), an analog to the Average Endpoint Error (AEE) error metric proposed in [19]. The PEE between the projected optical flow ($\widetilde{\mathbf{u}}$) and the groundtruth ($\hat{\mathbf{n}}$) is defined as:

$$\text{PEE} = \left\| \hat{\mathbf{n}} - \frac{\nabla I}{||\nabla I||_2^2} \cdot \widetilde{\mathbf{u}} \right\| \tag{15}$$

To evaluate the regressed relative poses from our model, we use Absolute Trajectory Error and Relative Pose Error metrics.

6.2. Analysis of Experimental Results

6.2.1 Accuracy of Normal Flow

In the first case study, we quantitatively evaluate *NFlowNet*. We compare our network with optical flow methods of various flavours: (a) supervised (PWC-Net [43], LiteFlowNet [21]), and (b) self-supervised (SelFlow [30]).

In Table 1, we present a quantitative evaluation of normal flow. We trained our *NFlowNet* and fine-tuned optical flow networks with TartanAir (8 environments). We demonstrate the PEE error on the first four sequences of the environments `abandonedfactory` and `hospital`. *NFlowNet* performs better than the optical flow networks by up to 6×. By learning normal flow, we constrain the network to focus on prominent features (edges, textures) rather than dense correspondences in textureless regions. Through this "attention-like" mechanism, *NFlowNet* performs better than its optical flow counterparts with upto 46× smaller model size.

Table 1. PEE (Projection Endpoint Error) ↓ of different state-of-the-art methods as compared to of our *NFlowNet*.

Method	abandonfactory				hospital				Num.
	000	001	002	003	000	001	002	003	param.(M) ↓
LiteFlownet [21]	2.56	1.82	1.93	2.15	3.17	2.68	2.45	1.93	5.37
PWC-Net [43]	1.23	0.95	1.64	1.48	2.35	2.92	2.28	1.47	8.75
UnFlow [31]	1.35	1.15	1.75	1.56	2.21	1.76	1.83	1.07	126.90
SelFlow [30]	**0.67**	0.73	**0.52**	0.64	1.91	0.51	0.73	**0.65**	5.11
NFlowNet (Ours)	0.72	**0.54**	0.57	**0.63**	**0.82**	**0.44**	**0.57**	0.71	**2.72**

6.2.2 Comparison of Pose Estimation

In this section we compare our *DiffPoseNet* framework with various state-of-the-art camera pose estimation approaches. These approaches can be broadly be classified into: (a) pure deep learning models, (b) pure geometric constraint based models, and (c) hybrid deep learning models that incorporate some form of geometric constraints in the learning pipeline.

We present the Absolute Trajectory Error (ATE) of our model on the TartanAir challenge data in the `MH000-007` sequences and compare it to the results of TartanVO and ORB-SLAM in Table 2. We outperform both methods in this experiment by up to 3.4×.

We present the ATE for the TUM-RGBD sequences (`360`, `desk`, `desk2`, `rpy`, `xyz`) in Table 3. We achieve

competitive results, because TUM RGBD is difficult for monocular vision methods due to rolling shutter, motion blur and large rotation. This is where pure geometric constraints show a massive advantage. We believe our work can further be improved by compensating for rolling shutter in the differential layer and we see this as a potential future research direction. It is important to stress that our method was trained on the TartanAir dataset and tested directly on other datasets to highlight cross-dataset generalization without any fine-tuning or re-training.

We also present the relative pose errors, specifically, the average translational RMSE drift (t_{rel} in %) and average rotational RMSE drift (r_{rel} in °/100m) for comparison in the KITTI dataset sequences `06`, `07`, `09` and `10` (See Fig. 3). The error metrics are computed on a trajectory of length of 100–800 m. In Table 4 we compare our model (*DiffPoseNet*) with (a) pure deep learning models (TartanVO [45], GeoNet [50], UnDeepVO [29], DeepVO [44], Wang et al. [47]), (b) pure geometric constraint based approaches (ORB-SLAM [32], VISO2-M [41]) and with (c) BiLevelOpt [25], a method implementing the epipolar constraint via a differentiable layer. The test sequences were selected such that they do not overlap with the training sets in the deep learning models used for comparison. Note that, similar to TartanVO, our training is performed only on TartanAir and do not perform any fine-tuning or re-training on KITTI dataset. Regardless, we perform competitive to other approaches that are trained/fine-tuned on similar data, thus demonstrating our cross-dataset generalization.

Table 4 also presents our ablation study, comparing different configurations in our pipeline on KITTI. 'Ours (no-SS)' denotes the results of the coarse PoseNet fine-tuned on KITTI groundtruth poses. 'Ours (no-CL)' denotes DiffPosenet without cheirality layer refinement during inference time, and 'Ours (CL 1-iter)' denotes DiffPoseNet with the refinement restricted to only one iteration of the cheirality layer. 'Ours (OF)' denotes the results of DiffPoseNet where the normal flow is computed by projecting optical flow on image gradients, instead of NFlowNet. We noticed that 'Ours' and 'Ours (OF)' are capable of outperforming learning approaches, particularly when there are rotation errors, which is notable because learning methods have shown higher rotation errors compared to geometric methods.

We infer from the above results (also see Fig. 4) that deep learning models usually outperform classical geometric constraint based methods in translation errors (t_{rel}), which can be attributed to the scale drift issue. This sometimes might be solved by performing expensive global bundle adjustment and loop closure. However, we are not using a loop closing procedure in the models in our experiments. Models with differentiable optimizer layers, like *DiffPoseNet* (Ours) and BilevelOpt [25], achieve the best of both worlds, with lower relative rotation errors competitive with geometric methods like ORB-SLAM.

Table 2. ATE (m) ↓ on the `MH` sequences of TartanAir [46] dataset.

Methods	`000`	`001`	`002`	`003`	`004`	`005`	`006`	`007`
ORB-SLAM [32]	**1.30**	**0.04**	2.37	2.45	-	-	21.47	2.73
TartanVO [45]	4.88	0.26	2.00	0.94	1.07	3.19	**1.00**	2.04
Ours	2.56	0.31	**1.57**	**0.72**	**0.82**	**1.83**	1.32	**1.24**

Table 3. ATE (m) ↓ on the TUM-RGBD [42] benchmark.

Methods	`360`	`desk`	`desk2`	`rpy`	`xyz`
ORB-SLAM2 [33]	-	**0.016**	0.078	-	**0.004**
DeepTAM [54]	**0.116**	0.078	0.055	0.052	0.054
TartanVO [45]	0.178	0.125	0.122	**0.049**	0.062
Ours	0.121	0.101	**0.053**	0.056	0.048

Table 4. Relative Pose Error (t_{rel} and r_{rel}) ↓ results of various Pose estimation methods on KITTI [17] dataset. Note that **bold** represents the best result and underline represents the second best.

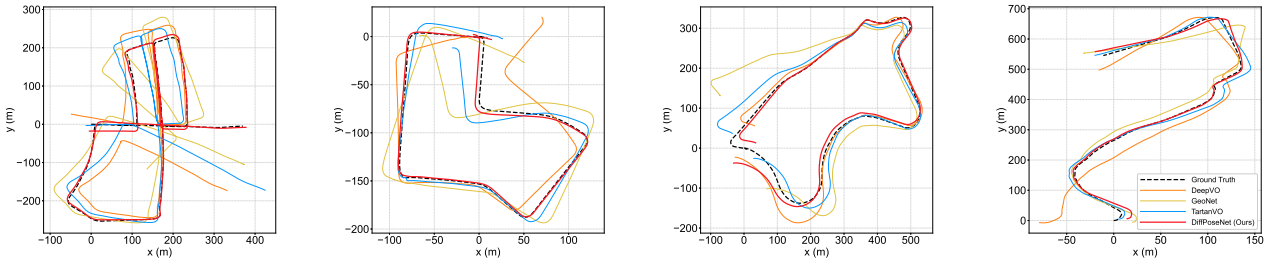
Method	`06`		`07`		`09`		`10`	
	t_{rel}	r_{rel}	t_{rel}	r_{rel}	t_{rel}	r_{rel}	t_{rel}	r_{rel}
DeepVO [44]	5.42	5.82	3.91	4.60	-	-	8.11	8.83
Wang et al. [47]	-	-	-	-	8.04	1.51	6.23	0.97
UnDeepVO [29]	6.20	1.98	**3.15**	2.48	-	-	10.63	4.65
GeoNet [50]	9.28	4.34	8.27	5.93	26.93	9.54	20.73	9.04
TartanVO [45]	4.72	2.95	4.32	3.41	6.03	3.11	6.89	2.73
BiLevelOpt [25]	-	-	-	-	4.36	0.69	4.04	1.37
ORB-SLAM [32]	18.68	**0.26**	10.96	**0.37**	15.3	**0.26**	**3.71**	**0.3**
VISO2-M [41]	7.34	6.14	23.61	19.11	4.04	1.43	25.2	3.84
Ours (no-SS)	5.23	3.15	4.83	3.92	7.12	4.31	8.33	3.74
Ours (no-CL)	4.23	2.43	4.28	2.76	5.13	3.18	5.89	2.98
Ours (CL-1-iter.)	3.19	2.03	4.13	2.53	4.72	1.71	4.82	2.57
Ours (OF)	3.03	2.08	3.89	2.13	4.24	0.72	4.12	1.56
Ours	**2.94**	1.76	4.06	2.35	**4.02**	0.51	3.95	1.23

6.2.3 Robustness of Pose Estimation

For most camera pose estimation approaches the performance of the algorithm is also governed by external factors such as lighting, weather, and sensor noise. These external factors often lead to errors in motion fields and cause pose estimation to fail or diverge. In this case study we present a robustness analysis by injecting noise into the motion fields.

We study the robustness of the normal flow based cheirality layer against the epipolar layer [25]. We artificially inject errors in the normal flow and optical flow and evaluate our framework's performance under these conditions. The error is modelled as additive uniform noise $\mathcal{U}(\epsilon)$, where $[-\epsilon, \epsilon]$ is the bound on noise. We induce this noise to both normal flow and optical flow only on the gradient direction where normal flow is well-defined to make the comparison fair.

Fig. 5 shows the relative pose errors, t_{rel} and r_{rel}, over ϵ values $\{0, 5, 10, 15, 20\}\%$. Here, BiLevelOpt refers to the pose estimation using the epipolar constraint layer [25] with optical flow as input. "Cheirality-OF" refers to the pose estimation via the cheirality layer using normal flow obtained



(a) Seq. 05 (b) Seq. 07 (c) Seq. 09 (d) Seq. 10

Figure 3. Qualitative comparison of trajectories between our *DiffPoseNet* and other state-of-the-art approaches on the KITTI dataset.

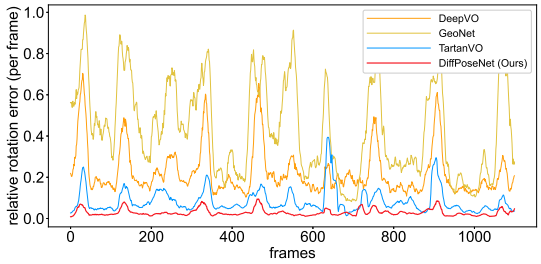


Figure 4. Comparison between our model and pure learning based VO on relative rotation error (in °/frame) from KITTI - 07.

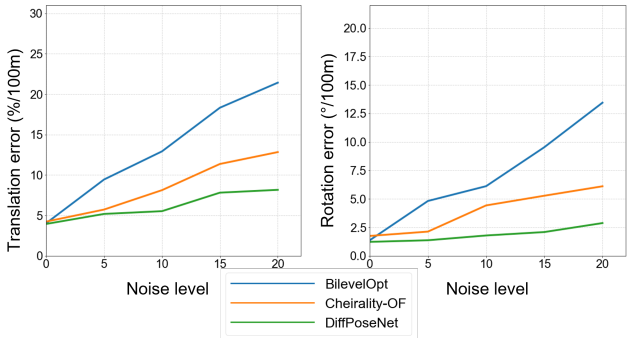


Figure 5. Robustness evaluation of normal flow representation for pose estimation on KITTI-10.

by projecting SelFlow optical flow predictions on the gradient directions (we choose SelFlow as it has the best performance among all optical flow methods used in this paper). Observe that, our *DiffPoseNet* is more resilient to noise and fails "more gracefully" compared to other methods. We owe this robustness to the lack of strong constraints used in our approach as compared to other methods that either rely on strong features or strong photometric consistency. We believe that carefully crafted optimization problems can lead to robust pose estimation neural networks that generalize well to novel datasets while being robust to noise, a capability rarely seen in most state-of-the-art methods [3, 36].

7. Conclusion and Future Work

In this work, we combined the best of both worlds from classical direct camera pose estimation approaches and deep learning taking advantage of differentiable programming concepts. Specifically, we addressed the problem of estimating relative camera pose using a sequence of images. To achieve this, we introduced the *DiffPoseNet* framework. As part of this framework, we introduced a normal flow network called *NFlowNet*, that predicts accurate motion fields under challenging scenarios and is more resilient to noise and bias. Furthermore, we proposed a differentiable cheirality layer that when coupled with *NFlowNet* can estimate robust and accurate relative camera poses.

A comprehensive qualitative and quantitative evaluation was provided on the challenging datasets: TartanAir, TUM-RGBD and KITTI. We demonstrated the efficacy, accuracy and robustness of our method under noisy scenarios and cross-dataset generalization without any fine-tuning and/or re-training. Our approach outperforms the previous state-of-the-art approach. Particularly, NFlowNet can output accurate motion fields at up to 6× the speed with up to 46× smaller model size. We believe this will open a new direction for camera pose estimation problems.

The currently pretrained posenet takes the role of "initialization" in the optimization pipeline. Since we are proposing DiffPoseNet as a generic framework, we will explore replacing posenet with better pose estimators in future work. Furthermore, the current approach focuses on static scenes (with no moving objects) only, and we plan to extend the pipeline to include segmentation of moving objects.

8. Acknowledgement

This work was supported in parts by the Office of Naval Research under Grant N00014-17-1-2622 and the National Science Foundation under Grant BCS 1824198 and OISE 2020624 respectively. Chethan M. Parameshwara was supported by William Hodos Dissertation Assistantship.

References

[1] Yiannis Aloimonos and Zoran Duric. Estimating the heading direction using normal flow. *International Journal of Computer Vision*, 13(1):33–56, 1994.

[2] Brandon Amos and J Zico Kolter. Optnet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning*, pages 136–145. PMLR, 2017.

[3] Adithya Prem Anand, H Gokul, Harish Srinivasan, Pranav Vijay, and Vineeth Vijayaraghavan. Adversarial patch defense for optical flow networks in video action recognition. In *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1289–1296, 2020.

[4] Francisco Barranco, Cornelia Fermüller, Yiannis Aloimonos, and Eduardo Ros. Joint direct estimation of 3d geometry and 3d motion using spatio temporal gradients. *Pattern Recognition*, 113:107759, 2021.

[5] Michael J Black and Paul Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding*, 63(1):75–104, 1996.

[6] Tomáš Brodskỳ and Cornelia Fermüller. Self-calibration from image derivatives. *International Journal of Computer Vision*, 48(2):91–114, 2002.

[7] Tomas Brodsky, Cornelia Fermüller, and Yiannis Aloimonos. Shape from video. In *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, volume 2, pages 2146–2146. IEEE Computer Society, 1999.

[8] Thomas Brox, Andrés Bruhn, Nils Papenberg, and Joachim Weickert. High accuracy optical flow estimation based on a theory for warping. In *European Conference on Computer Vision*, pages 25–36. Springer, 2004.

[9] Thomas Brox and Jitendra Malik. Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):500–513, 2010.

[10] Richard H. Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.

[11] Dylan Campbell, Liu Liu, and Stephen Gould. Solving the blind perspective-n-point problem end-to-end with robust differentiable geometric optimization. In *European Conference on Computer Vision*, pages 244–261. Springer, 2020.

[12] Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. Monoslam: Real-time single camera slam. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1052–1067, 2007.

[13] 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.

[14] Cornelia Fermüller. Passive navigation as a pattern recognition problem. *International journal of computer vision*, 14(2):147–158, 1995.

[15] Cornelia Fermüller, Robert Pless, and Yiannis Aloimonos. The ouchi illusion as an artifact of biased flow estimation. *Vision Research*, 40(1):77–95, 2000.

[16] Cornelia Fermüller, David Shulman, and Yiannis Aloimonos. The statistics of optical flow. *Computer Vision and Image Understanding*, 82(1):1–32, 2001.

[17] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[18] Stephen Gould, Richard Hartley, and Dylan John Campbell. Deep declarative networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[19] Bas J. Pijnacker Hordijk, Kirk Y. W. Scheper, and Guido C.H.E. de Croon. Vertical landing for micro air vehicles using event-based optical flow. *J. Field Robotics*, 35:69–90, 2018.

[20] Berthold KP Horn and EJ Weldon. Direct methods for recovering motion. *International Journal of Computer Vision*, 2(1):51–76, 1988.

[21] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. LiteFlowNet: A Lightweight Convolutional Neural Network for Optical Flow Estimation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8981–8989, 2018.

[22] 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.

[23] Hui Ji and Cornelia Fermuller. A 3d shape constraint on video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(6):1018–1023, 2006.

[24] Hui Ji and Cornelia Fermüller. Better flow estimation from color images. *EURASIP Journal on Advances in Signal Processing*, 2007:1–9, 2007.

[25] Shihao Jiang, Dylan Campbell, Miaomiao Liu, Stephen Gould, and Richard Hartley. Joint unsupervised learning of optical flow and egomotion with bi-level optimization. In *2020 International Conference on 3D Vision (3DV)*, pages 682–691. IEEE, 2020.

[26] Rico Jonschkowski, Austin Stone, Jonathan T Barron, Ariel Gordon, Kurt Konolige, and Anelia Angelova. What matters in unsupervised optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 557–572. Springer, 2020.

[27] Alex Kendall and Roberto Cipolla. Geometric loss functions for camera pose regression with deep learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6555–6564, 2017.

[28] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2938–2946, 2015.

[29] Ruihao Li, Sen Wang, Zhiqiang Long, and Dongbing Gu. Undeepvo: Monocular visual odometry through unsupervised deep learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7286–7291, 2018.

[30] Pengpeng Liu, Michael R. Lyu, Irwin King, and Jia Xu. Selflow: Self-supervised learning of optical flow. In *CVPR*, 2019.

[31] Simon Meister, Junhwa Hur, and Stefan Roth. UnFlow: Unsupervised learning of optical flow with a bidirectional census loss. In *AAAI*, New Orleans, Louisiana, Feb. 2018.

[32] Raúl Mur-Artal, J. M. M. Montiel, and Juan D. Tardós. Orb-slam: A versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[33] Raúl Mur-Artal and Juan D. Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, 2017.

[34] S Negahdaripour. Direct methods for structure from motion. *Ph. D Thesis, Mechanical Engineering, MIT*, 1986.

[35] 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.

[36] Anurag Ranjan, Joel Janai, Andreas Geiger, and Michael Black. Attacking optical flow. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2404–2413, 2019.

[37] Nitin J Sanket, Chahat Deep Singh, Chethan M Parameshwara, Cornelia Fermüller, Guido CHE de Croon, and Yiannis Aloimonos. EVPropNet: Detecting Drones By Finding Propellers For Mid-Air Landing And Following. In *Robotics: Science and systems (RSS) conference 2021*. Robotics: Science and Systems, 2021.

[38] Davide Scaramuzza and Friedrich Fraundorfer. Visual odometry [tutorial]. *IEEE robotics & Automation Magazine*, 18(4):80–92, 2011.

[39] César Silva and José Santos-Victor. Robust egomotion estimation from the normal flow using search subspaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(9):1026–1034, 1997.

[40] David Sinclair, Andrew Blake, and David Murray. Robust estimation of egomotion from normal flow. *International Journal of Computer Vision*, 13(1):57–69, 1994.

[41] Shiyu Song, Manmohan Chandraker, and Clark C. Guest. High accuracy monocular sfm and scale correction for autonomous driving. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(4):730–743, 2016.

[42] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[43] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *CVPR*, 2018.

[44] Sen Wang, Ronald Clark, Hongkai Wen, and Niki Trigoni. Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2043–2050. IEEE, 2017.

[45] Wenshan Wang, Yaoyu Hu, and Sebastian Scherer. Tartanvo: A generalizable learning-based vo. *arXiv preprint arXiv:2011.00359*, 2020.

[46] Wenshan Wang, Delong Zhu, Xiangwei Wang, Yaoyu Hu, Yuheng Qiu, Chen Wang, Yafei Hu, Ashish Kapoor, and Sebastian Scherer. Tartanair: A dataset to push the limits of visual slam. 2020.

[47] Xiangwei Wang, Daniel Maturana, Shichao Yang, Wenshan Wang, Qijun Chen, and Sebastian Scherer. Improving learning-based ego-motion estimation with homomorphism-based losses and drift correction. In *Proceedings of (IROS) IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 970 – 976, November 2019.

[48] Philip Wolfe. Convergence conditions for ascent methods. *SIAM Review*, 11(2):226–235, 1969.

[49] Bernhard P. Wrobel. Multiple view geometry in computer vision. *Künstliche Intell.*, 15:41, 2001.

[50] Zhichao Yin and Jianping Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1983–1992, 2018.

[51] Ding Yuan, Miao Liu, and Hong Zhang. Direct ego-motion estimation using normal flows. In *2013 2nd IAPR Asian Conference on Pattern Recognition*, pages 310–314. IEEE, 2013.

[52] 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.

[53] W. Zhao, S. Liu, Y. Shu, and Y. Liu. Towards better generalization: Joint depth-pose learning without posenet. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9148–9158, Los Alamitos, CA, USA, jun 2020. IEEE Computer Society.

[54] H. Zhou, B. Ummenhofer, and T. Brox. Deeptam: Deep tracking and mapping. In *European Conference on Computer Vision (ECCV)*, 2018.

[55] Ciyou Zhu, Richard H Byrd, Peihuang Lu, and Jorge Nocedal. L-bfgs-b: a limited memory fortran code for solving bound constrained optimization problems. *Technique Report Rep*, 1994.