Robust Visual Odometry Using Position-Aware Flow and Geometric Bundle Adjustment

Yijun Cao, Xianshi Zhang, Fuya Luo, Peng Peng, and Yongjie Li, Senior Member, IEEE

Abstract—In this paper, an essential problem of robust visual odometry (VO) is approached by incorporating geometry-based methods into deep-learning architecture in a self-supervised manner. Generally, pure geometry-based algorithms are not as robust as deep learning in feature-point extraction and matching, but perform well in ego-motion estimation because of their well-established geometric theory. In this work, a novel optical flow network (PANet) built on a position-aware mechanism is proposed first. Then, a novel system that jointly estimates depth, optical flow, and ego-motion without a typical network to learning ego-motion is proposed. The key component of the proposed system is an improved bundle adjustment module containing multiple sampling, initialization of ego-motion, dynamic damping factor adjustment, and Jacobian matrix weighting. In addition, a novel relative photometric loss function is advanced to improve the depth estimation accuracy. The experiments show that the proposed system not only outperforms other state-of-the-art methods in terms of depth, flow, and VO estimation among self-supervised learning-based methods on KITTI dataset, but also significantly improves robustness compared with geometry-based, learning-based and hybrid VO systems. Further experiments show that our model achieves outstanding generalization ability and performance in challenging indoor (TMU-RGBD) and outdoor (KAIST) scenes.

Index Terms—visual odometry, self-supervise learning, optical flow, monocular depth estimation, joint learning, robustness.

I. INTRODUCTION

Simultaneous localization and mapping (SLAM) or visual odometry (VO) for estimating depth and relative camera poses from image sequences is a fundamental problem in the fields of computer vision and robots, involving numerous applications, including robotics navigation [1] and augmented reality [2]. Conventional VO algorithms [3], [4] estimate camera motion via multi-view geometry, such as bundle adjustment (BA) and epipolar geometry. However, a pure geometry-based VO system is accurate only under some conditions, including sufficient illumination and texture to establish correspondence, sufficient overlap between consecutive frames, and static scenes, which means dynamic objects, such as moving cars, must be as few as possible.

With the strong capability of convolutional neural networks (CNNs) [5], [6], learning-based VO models [7], [8] have been proposed to address these issues in a self-supervised manner. In general, they jointly train two networks to separately predict the depth and camera motion based on image reconstruction error. However, deep-learning approaches greatly depend on the distribution of training data and lack of generalization in many real situations, such as rapid camera movement, random noise, and rolling shutter effect. These aspects can greatly affect the VO performance, which has not been fully evaluated in learning-based algorithms. In addition, learning-based VO methods fail to provide sufficient reliability and accuracy compared with those geometry-based methods under geometrically favorable conditions.

Motivated by these observations, researchers have tried to combine the advantages of learning- and geometry-based methods to estimate VO using separate [9], [10] or integrated [11] training strategies. However, most of these schemes ignore the requirement of robustness while improving accuracy. Figure 1 shows three common problems in VO tasks. Note that the geometric noise is used to simulate the rolling shutter effect [4], [12]. It can be seen that under conditions with high noise or fast motion (or low frames per second, FPS), existing VO systems have large performance fluctuations, whether the systems learning- or geometry-based.

For avoiding performance degradation and tracking failure due to noise, we argue that the following two aspects are relatively important. The first one is a robust optical flow estimation. As an input to many geometric methods, such as epipolar geometry, perspective-n-point (PnP) and BA [13], accurate optical flow implies accurate pose estimation, yet traditional algorithms [14] have a significant disadvantage over deep learning for this task, both in terms of accuracy and stability to different lighting conditions. Therefore, based on the existing optical flow network using self-supervised method, a new optical flow network using position-aware mechanism are proposed in this work. The second aspect is a
pose estimation module. Earlier approaches \cite{10,11} tended to combine epipolar geometry and PnP to compute camera motion, however, PnP algorithm need more accurate depth performance, which is hard to obtain only using monocular self-supervised manner. Based on the above analysis, geometric BA algorithm is used in this work as pose estimation module, because i) it relies on a precise optical flow, which is easy to obtain in self-supervised manner, but not on a precise depth; ii) the use of hubel loss \cite{13} in optimizing re-projection error can reduce a part of effect of outlier.

In summary, the contributions of this paper lie in three aspects:

1. A novel position-aware flow network (PANet) is proposed, which works as the position-aware model (PAM) to replace the optical flow estimator of PWC-Net \cite{15}. Inspired by the attention mechanism \cite{16}, the proposed PAM is invoked to predict the probability of optical flow and map it to a fixed optical flow table. It barely increases the number of parameters while improving performance and robustness.

2. The second contribution is a novel learning framework combining geometric BA into a joint depth and optical flow learning model in which the components can benefit from each other. In this way, the BA module jointly computes ego-motion and optimizes inverse depth, which can directly supervise Depth-Net \cite{17} to learn accurate and scale-consistent results.

3. To embed the BA module with the Levenberg-Marquardt (LM) algorithm \cite{18} into the entire system and train them together, several strategies are employed: i) using multiple filtering to ensure the stability of the input pixels; ii) dynamically adjusting the damping factor \( \lambda \) according to the error of each training batch, iii) assigning different weights to the Jacobian matrix of pose and inverse depth; and iv) initializing with PnP. These strategies help achieve a better convergence of the BA module throughout the training process.

Experiments show that the proposed system not only achieves state-of-the-art performance of depth, flow, and VO estimation on the KITTI dataset, but also significantly improves the robustness in many challenging scenes with a self-supervised learning method, as shown in Fig. 1. Note that high-speed movement or low FPS, random photometric noise, and the rolling shutter effect (geometric noise) is simulated by sampling the original sequences with large stride, adding randomly distributed Gaussian noise, and perturbing the locations of pixels in a small neighborhood, respectively. All the synthetic data are unseen during training. It can be seen that the proposed method performs well when tested on changing environments, showing that the proposed method significantly improves robustness.

II. RELATED WORK

Monocular depth estimation. Learning depth from monocular image is a fundamental problem in computer vision, and has important applications in SLAM, 3D reconstruction and scene understanding. Early algorithms used specific handcrafted features (texture variations \cite{19}, gradients, vanishing point \cite{20} etc.) or strong assumptions, such as posterior distribution \cite{19} and scene structure \cite{21}, to predict sparse depth representation.

With the development of CNNs, a variety of models are proposed to learn monocular depth in a supervised manner \cite{22, 23, 24}. However, these methods require labeled ground truth, which are expensive to obtain in natural environments. More recent works have begun to approach the problem in a self-supervised or unsupervised way. A pioneering work is SfMLearner \cite{7}, which learns depth and ego-motion jointly by minimizing photometric loss in an unsupervised manner. This pipeline has inspired a large amount of follow-up works.

To deal with moving objects breaking the assumption of static scenes, some works \cite{25, 26} use consistency of forward-backward optical \cite{11}, depth-optical \cite{27}, or depth-depth \cite{8} flow to mask dynamic objects, while others predict segmentation masks by pre-trained segmentation models \cite{28, 29}. Several methods develop the framework by changing training strategies and adding supplementary constraints \cite{17, 30}, including ICP regularization \cite{31}, epipolar constraint \cite{32}, and collaborative competition \cite{33}.

More recently, several researchers \cite{11} have tried to combine a geometric module into the standard structure-from-motion (SfM) pipeline, and obtained better depth and VO estimations by training with only two frames in a video sequence. Different from those methods, the proposed method embeds geometric BA in the entire system, which forecasts ego-motion and depth simultaneously and improves both accuracy and generalization.

Optical flow estimation. Optical flow techniques attempt to get apparent movement of brightness patterns between continuously images. Classical methods infer optical flow for a pair of images by minimizing photometric consistency and smoothness \cite{34, 35}. Recent approaches \cite{16, 15, 17} see optical flow estimation as a supervised learning problem with CNN. Although these methods produce better performance by incorporating useful components, such as cost volumes and coarse-to-fine warping \cite{15}, supervised methods require accurate optical flow labels which is laborious, and requires approaches as unusual as manually painting scenes with textured fluorescent paint and imaging it under ultraviolet light \cite{38}.

Some synthetic datasets, such as Flying Chairs \cite{39} and Sintels \cite{40}, require careful consideration of scene content, camera motion, lens distortion, and sensor degradation \cite{41}.

Unsupervised methods overcome the need for labels by optimizing photometric consistency and local flow smoothness. Some recent approaches significantly improve the results by omitting occluded regions using forward-backward consistency checks \cite{42} or range-map filtration \cite{16}. Other extensions introduce edge-aware smoothness \cite{16}, data distillation \cite{42}, estimation from multiple frames \cite{43}, and co-training optical flow with depth and ego-motion models \cite{26, 11}.

The proposed self-supervised optical flow estimation model builds on \cite{11} and further improves the optical flow network using position-aware mechanism.

Visual odometry. VO is a long-standing problem that estimates the ego-motion incrementally using visual input. Benefiting from the theory of multi-view geometry, a geometry-based VO system usually consists of two steps. First, the raw
III. METHOD

A. Overview

As shown in Fig. 2, the overall system proposed includes three components: Depth-Net [17], the proposed PANet, and a geometric BA module, which are unified into a novel framework in a self-supervised way. Given a pair of consecutive frames $I_t$ and $I_{t'}$, the Depth-Net and proposed PANet outputs inverse depth $d_t$, and optical flow $F_{t \rightarrow t'}$, respectively. The BA module receives $d_t$ and $F_{t \rightarrow t'}$, and outputs sparse inverse depth $\hat{d}_\text{ba}$ and ego-motion $T_{t \rightarrow t'}$ by optimizing the re-projection error using the LM algorithm.

In the following, we will first describe the network architecture of proposed PANet (Sec. III-B), and then demonstrate the geometric BA module for suitting deep learning architectures (Sec. III-C). Finally, a detailed description of the losses and training strategy are presented in Sec. III-D.

B. PANet

PANet builds upon PWC-Net [15], which is composed of two compact components: a pyramidal feature extractor and optical flow estimator. The feature extractor is a six-stage convolutional architecture, and each stage includes three $3 \times 3$ convolution layers activated by LeakyReLU. From coarse to fine, the optical flow estimator receives concatenated features that consist of cost volume, the feature $f_1^i$ from the first frame $I_t$, and the optical flow $F_{t \rightarrow t'}^i$ from the coarse level, as shown in Fig. 3. Given the feature map $f_1^i$ and warping feature $F_{t \rightarrow t'}^i$, the cost volume layer computes the matching costs for associating a pixel with its corresponding pixels at the next frame in a local region.

Each layer of the optical flow estimator of PWC-Net [15], such as that underlying Fig. 3(a), predicted optical flow straightforward via a multi-layer CNN, while the proposed PAM introduced a priori of location of the optical flow. This prior position is about the possible optical flow of a pixel from the first frame $I_t$ to the second frame $I_{t'}$, i.e. the offset value, and defined as two 2D grids along the $x$ and $y$ axes:

$$G_x = \begin{bmatrix} -k & \cdots & k \\ \vdots & \ddots & \vdots \\ -k & \cdots & k \end{bmatrix}, \quad G_y = \begin{bmatrix} k & \cdots & k \\ \vdots & \ddots & \vdots \\ -k & \cdots & -k \end{bmatrix},$$

(1)
where \( k \) is set to 4, the same as previous work [15]. \( G_x \) and \( G_y \) denote the possible optical flow values of a local region along the x and y directions, respectively. With a priori coordinates, the model needs to predict a four-dimensional (4D) probability volume \( (PV \in \mathbb{R}^{K \times K \times H \times W}) \), which can be viewed as a weight matrix, and stores the probability of the optical flow to fall in the area with a size of \( K \times K \) pixels, where \( K = 2k + 1 \). Predicting a weight matrix (PV) theoretically reduces uncertainty in forecasting compared with directly predicting a large range of optical flows \([-k, k]\). This process may be able to better approximate the optimal value in a self-supervised learning, the relevant experiments can be seen in Table III. Next, the optical flow is obtained by a weighted average of PV over predefined 2D position matrices \( G_x \) and \( G_y \), and formulated as

\[
F_x(i, j) = \frac{1}{K^2} \sum_{u=-k}^{k} \sum_{v=-k}^{k} PV(u, v, i, j)G_x(u, v),
\]
\[
F_y(i, j) = \frac{1}{K^2} \sum_{u=-k}^{k} \sum_{v=-k}^{k} PV(u, v, i, j)G_y(u, v),
\]

where \( F_x(x, y) \) and \( F_y(x, y) \) are the components of the optical flow along the x and y axis, respectively. The position-aware means that the process requires the network to learn an awareness matrix that does weighting on the predefined optical flow positions.

C. Geometric BA module

Correspondence and Sample. Optical flow provides dense correspondence for every pixel. However, not all correspondences can be used correctly due to the errors of optical flow and noise caused by moving objects. Therefore, it is very important to filter noise and unnecessary points before BA. Inspired by previous studies [11, 10], screening was done in three stages to ensure the reliability of the alternative correspondence. Given a forward optical flow \( F_{t \rightarrow t'} \), backward \( F_{t' \rightarrow t} \) optical flows, and flow occlusion mask \( M_o \), which is composed of a forward-backward consistency mask [51] and range-map mask [16], the first filtering condition is forward-backward flow consistency. Specifically, good 2D-2D correspondences are chosen according to top 20% consistency from \( S_o = M_o/[ F_{t \rightarrow t'} - F_{t' \rightarrow t} ] \). The second key operation is the removal of moving objects. After computing a fundamental matrix by using 3,000 best correspondences from the first step, the inlier score is computed as \( S_r = 1(D_e < 0.5)/(1.0 + D_e) \), where \( D_e \) is the distance map representing the distance from pixels to its corresponding epipolar line, and \( 1(\cdot) \) denotes indicator function. The last process is to filter the matches that have minimal ray angles or invalid re-projection [11], and the filter is expressed as a mask, \( M_a \). To improve computational efficiency, the best 3,000 matches from \( M_oS_oS_a \) are randomly sampled.

Geometric BA. After filtering and sampling, several inverse depths \( \hat{d} \) can be obtained and then fed into the geometric BA module. The geometric BA [52] jointly optimizes camera poses \( T \) and 3D scene coordinates by minimizing the re-projection error, and has been the gold standard for SFM in the last two decades. To embed it into the proposed training pipeline, the geometric BA algorithm was simplified into two-view form and only the camera poses \( T \) and inverse depth \( \hat{d} \) of several points were optimized. The objective function is formulated as follows,

\[
\mathcal{X} = \arg \min_{T, \hat{d}} \sum_{i=1}^{N} \omega|\mathcal{E}_i|,
\]

where \( \mathcal{X} = \{ T, d_i | i = 1, 2, ..., N \} \). \( \mathcal{E} \) is the error of geometric distance, and defined as \( \mathcal{E}_i = p_i - P_{depth}(P_{flow}(p_i)) \), where the functions \( P_{depth}(p_i) \) and \( P_{flow}(p_i) \) are the camera perspective projection and photometric correspondence, respectively, that are formulated as in Eq. 6. The weight \( \omega \) is computed with Huber loss [3]. The symbol \( d_i \) is the inverse depth of the \( i \)th pixel.

The general strategy to minimize Eq. 3 is to use the LM algorithm. At each iteration, the LM algorithm solves for an optimal update \( \Delta \mathcal{X}^* \) to the solution by minimizing:

\[
\Delta \mathcal{X}^* = \arg \min || J(\mathcal{X}) \Delta \mathcal{X} + E(\mathcal{X}) || + \lambda || D(\mathcal{X}) \Delta \mathcal{X} ||,
\]

where, \( E(\mathcal{X}) = [E_1, E_2, ..., E_N] \), \( J(\mathcal{X}) = [J_T, J_d] \) is the Jacobian matrix of \( E(\mathcal{X}) \) respect to camera poses \( T \) and inverse depth \( d \), \( D(\mathcal{X}) \) is a non-negative diagonal matrix, typically the square root of the diagonal of the approximated Hessian \( J(\mathcal{X})^T J(\mathcal{X}) \). The non-negative value \( \lambda \) controls the regularization strength.

However, a standard LM algorithm is not suitable in end-to-end self-supervised training, which has two main problems. The first one is fixed parameter initialization on damping factor \( \lambda \) and ego-motion \( T \). On the premise of ensuring as few iterations as possible, \( T \) with large error or a small \( \lambda \) will make the algorithm difficult to converge to the minimum, while a large \( \lambda \) leads to a local optimal solution. To solve this problem, use of PnP algorithm [53] was targeted to do ego-motion \( T \) initialization. Compared with the initial attempt with an identity matrix, the errors of both using PnP and an identity matrix are high in the early stage of model training, while, in the later stages of training, PnP can better approximate the optimal solution. The following equation was also used to initialize \( \lambda \),

\[
\lambda = \lambda_{min} + (\lambda_{max} - \lambda_{min}) \exp(-\frac{1}{\sigma} \sum_{i=1}^{N} \omega|\mathcal{E}_i|),
\]

where the minimum \( \lambda_{min} = 1 \), maximum \( \lambda_{max} = 10^4 \) and \( \sigma = 5 \) are empirically set for training. Damping factor \( \lambda \) controls whether BA prefers to use first- or second-order optimization. When the error is large, Eq. 5 makes \( \lambda \) decrease so that the LM algorithm tends to converge quickly with the second-order method. In contrast, when the error is small, the equation makes \( \lambda \) increase so that LM algorithm tends to increase the convergence accuracy with the first-order method.

The second issue is an empirical discovery that the numerical ratio of the Jacobi matrix of \( T \) and \( d \) will greatly affect the training result of the entire model, especially Depth-Net. In the training phase, if the depth Jacobi matrix \( J_d \) is given a small weight \((J(\mathcal{X}) = [J_T, w_d J_d])\), where \( w_d < 1 \), then the training accuracy of Depth-Net will be essentially improved (see Table I), and, conversely, in the testing phase, if the
small weight of the depth Jacobi matrix is still maintained, the
testing error of the ego-motion will be particularly large (see
Table[V]). It is conjectured that this phenomenon is caused by
the inconsistent range of values of the depth and pose Jacobi
matrixes, and smaller values will give more accurate results
in the joint optimization using Schur-Complement [13]. Based
on the above finding, a small weight \( w_d = 0.1 \) is set for the
depth Jacobi matrix in the training phase, and it is restored to
\( w_d = 1.0 \) during inference phase.

D. Training the system

Self-supervised learning pipeline. The method for finding
corresponding pixels with regard to depth and optical flow is
introduced first. For a pixel \( p_t \) in \( I_t \), the corresponding pixel \( p_{t'} \)
in \( I_{t'} \) can be found either through camera perspective projection
\( P_{depth}(p_t) \) or the photometric correspondence \( P_{flow}(p_t) \),
which should be consistent for static scenes. Formally, the two
relationships can be written as

\[
P_{depth}(p_t) = K T_{t\rightarrow t'} D_t(p_t) K^{-1} p_t,
\]

\[
P_{flow}(p_t) = p_t + F_{t\rightarrow t'}(p_t),
\]

where \( K \) and \( D_t(p_t) = 1/d_t(p_t) \) denote camera intrinsic
and the depth in \( p_t \), respectively. Here, conversion between
homogeneous and non-homogeneous coordinates is omitted.

After computing the corresponding \( p_t \) and \( p_{t'} \), the image \( I'_{t'} \)
can be synthesized using \( I_t \). Then, the self-supervised training
of both depth and optical flow is realized by minimizing the
photometric errors between \( I_t(p_t) \) and the synthetic image
\( I'_{t'}(p_t) \):

\[
L_{self}^\circ(W) = \sum_{p_t} W(p_t) r(I_t(p_t), I'_{t'}(p_t)),
\]

where \( W(p_t) = \delta(p_t) \) indicates the way in which the images are synthesized to train the self-supervised loss
\( L_{self}^\circ(W) \). The \( r(I_t(p_t), I'_{t'}(p_t)) \) is the metric between target image
\( I_t(p_t) \) and synthetic image \( I'_{t'}(p_t) \), and is usually defined as
\( L1 + SSIM \) [54]:

\[
r(I_t, I'_{t'}) = \frac{1}{2} (1 - SSIM(I_t, I'_{t'})) + \alpha |I_t - I'_{t'}|,
\]

where \( \alpha = 0.85 \) is set for the training processes.

Pixel color matching alone is unstable and ambiguous.
Therefore, an edge-aware smoothness term is often applied
for regularization:

\[
L_s(V, \beta, k) = \sum_{p_t} \sum_{r \in S, b} \left| \frac{\partial^k V(p_t)}{\partial r^k} \right| e^{-\beta |\partial_{x,y} V(p_t)|},
\]

where \( V \) represents the type of input and \( \beta \) is the order of the
smoothness gradient.

Training strategies and losses. In practice, it is found that
jointly training all networks does not generate reasonable
outputs. The possible reasons are that i) random initialization
of different modules (Depth-Net and PANet) can lead to
difficulties in converging the network to the same objective
function; ii) self-supervised cost functions are difficult to train
complex networks effectively in the presence of multiple task
objectives. According to previous work [26], [11], the entire
training schedule consists of three stages: (1) stage I: self-
supervised training optical flow, (2) stage II: fixed optical
flow network weights, self-supervised training depth and (3)
stage III: fixed depth network parameters, fine-tuned optical
flow using consistency of depth reprojection and optical flow
mapping.

First, the proposed PANet is trained in a self-supervised
manner, and the loss function \( L_{flow} \) consists of photometric
error and edge-aware smoothness:

\[
L_{flow} = \frac{1}{2} \sum_{F \in \{F_{t\rightarrow t'}, F_{t'\rightarrow t}\}} L^F_{self}(M_o) + \gamma_S L_s(F, 10, 2),
\]

where \( M_o \) is the occlusion mask mentioned in Section [III-C].
Second, the PANet parameters are fixed and Depth-Net
trained using \( L_{depth} \), which consists of relative photometric
error, geometric consistency [8], edge-aware smoothness, and
point-wise depth loss:

\[
L_{depth} = \frac{1}{2} \sum_{d \in \{d_p, d_{t'}\}} \gamma_{d_{rp}} L_{rp}^d + \gamma_{d_{s}} L_{s}(d, 2, 1) + \gamma_{dc} L_{dc}^d + \gamma_{dba} L_{dba}^d,
\]

where geometric consistency \( L_{dc} \) is defined in sc-SfMLearner
[8]. During the experiment, it was found that the conventional
photometric error performs poorly in dark light, which may
be due to the small weighting caused by the small pixel value
in the darkness region. Therefore, a relative photometric error
function was proposed, which is written as

\[
L_{rp}^d = \frac{1}{N} \sum_{p_t} M_d(p_t) \frac{r(I_t, I'_{t'}(p_t))}{I_t(p_t)},
\]

where function \( r(I_t, I'_{t'}) \) is defined in Eq. [8]. \( M_d \) is the
validation mask of depth, which includes the auto-mask, depth
projection mask [17], and inlier score \( S_t \) mentioned in Section
[III-C]. \( N \) is the sum of \( V_t \). The point-wise depth loss is the difference between the predicted inverse depth after
sampling \( \hat{d}(p_t) \) and BA outputted inverse depth \( \hat{d}_{ba}(p_t) \), and
is formulated as follows:

\[
L_{ba} = \frac{1}{N} \sum_{p_t} |\hat{d}(p_t) - \hat{d}_{ba}(p_t)|.
\]

In the last training stage, the Depth-Net was fixed and
PANet was fine-tuned using cross-task loss \( L_{cross} \) [27]:

\[
L_{total} = \gamma_f L_{flow} + \gamma_c L_{cross},
\]

where \( L_{cross} \) is the minimizing of \( P_{depth}(p_t) \) and \( P_{flow}(p_t) \).
The two flow fields should be consistent with each other for
non-occluded and static regions. Minimizing the discrepancy
between the two flow fields allows one to simultaneously
update the depth and flow models:

\[
L_{cross} = \frac{1}{N} \sum_{p_t} S_r(p_t)|P_{depth}(p_t) - P_{flow}(p_t)|.
\]

Note that we fix the Depth-Net in the third stage unlike some
joint learning algorithms [11], [27] due to that the use of cross-
task loss \( L_{cross} \) assumes that the scene is static or that the
optical flow and depth have a similar dynamic target mask,
which is very rare in traffic scenarios.
IV. EXPERIMENTS

Dataset. The KITTI dataset [55] provides videos in 200 street scenes captured by RGB cameras, with sparse depth ground truths captured by laser scanner. For depth and flow evaluation, training was done on KITTI raw [8], [11], [17] and videos were resized to 832 × 256. The depth was evaluated on Eigen’s testing split [56], and the optical flow on the KITTI 2015 training set. For KITTI odometry evaluation, the standard setting [8], [11], [17], which use sequences 00-08 for training and 09-10 for testing, was followed.

The KAIST urban dataset [60] provides light detection and ranging (LiDAR) data and stereo image with various position sensors targeting an urban environment. Compared to KITTI, it has longer mileage and a more complex traffic environment (e.g., metropolis areas, complex buildings and residential areas), and is therefore more challenging. The sequences containing stereo images are urban18-39, and we used urban26-39 to build the dataset, dropping the simple highway sequence urban18-25. For VO evaluation, we use urban26, 29, 30, 32, 34, 37, 38 for training and the others for testing.

TUM-RGBD dataset [61] is a prevalent public benchmark used by many VO/SLAM algorithms [4], [5], [11], [47]. The dataset was collected in indoor environments with various challenging conditions including non-texture, dynamic objects and abrupt motions. The dataset provides both monocular RGB images and depth images, while only the RGB images are used in our experiments. For training our model using this dataset, we follow Xue’s training and testing split [47].

Depth network architectures. For the depth network, the same architecture as [17] was used, which adopts ResNet18 [8] as the encoder. Mirrored exponential disparity (MED) [58] was substituted for a conventional Sigmoid function [17] as the output activated process.

Implementation details. The Adam [62] optimizer was used, the learning rate set to 10⁻⁴, and the batch size to 10. The training epochs from the first to the last stages were 30, 30, and 20. In the first two training stages, the learning rate was decreased to 5 × 10⁻⁵ in the last 10 epochs. The LM iteration was set to 30 in training and inference. The coefficients [γf, γf∗, γdrp, γds, γdc, γdba, γc] are set to [1, 0.1, 0.2, 10⁻³, 0.5, 0.5, 0.002].

A. Depth evaluation

Ablation Study. This part presents the ablation analysis (Table I) used to assess the performance of all components of our method. The definition of the metric can be found in [26]. Several control experiments are generated for evaluation, including:

| Method | Train | AbsRel↓ | SqRel↓ | RMS↓ | RMSLog↓ | < 1.25 ↑ | < 1.25² ↑ | < 1.25³ ↑ |
|--------|-------|---------|--------|------|---------|---------|---------|---------|
| EPC++  [26] | MS    | 0.127   | 0.936  | 5.008 | 0.209   | 0.841   | 0.946   | 0.979   |
| Monodepth2 [17] | MS    | 0.106   | 0.818  | 4.750 | 0.196   | 0.874   | 0.957   | 0.979   |
| D3VO [9] | MS    | 0.099   | 0.763  | 4.488 | 0.185   | 0.885   | 0.958   | 0.979   |
| SuperDepth [57] | S     | 0.112   | 0.875  | 4.958 | 0.207   | 0.852   | 0.947   | 0.977   |
| Monodepth2 [17] | S     | 0.109   | 0.873  | 4.960 | 0.209   | 0.834   | 0.948   | 0.975   |
| FAL [58] | S     | 0.097   | 0.590  | 3.991 | 0.177   | 0.893   | 0.966   | 0.984   |
| SiMLearner [7] | M     | 0.183   | 1.595  | 6.709 | 0.270   | 0.734   | 0.902   | 0.959   |
| Geonet [25] | M     | 0.155   | 1.296  | 5.857 | 0.233   | 0.793   | 0.931   | 0.973   |
| DF-Net [27] | M     | 0.150   | 1.124  | 5.507 | 0.223   | 0.806   | 0.933   | 0.973   |
| CC [33] | M     | 0.140   | 1.070  | 5.326 | 0.217   | 0.826   | 0.941   | 0.975   |
| EPC++ [26] | M     | 0.144   | 1.029  | 5.305 | 0.216   | 0.816   | 0.941   | 0.976   |
| SC-SfMLearner [8] | M     | 0.137   | 1.089  | 5.439 | 0.217   | 0.830   | 0.942   | 0.975   |
| Monodepth2 [17] | M     | 0.115   | 0.882  | 4.701 | 0.190   | 0.879   | 0.961   | 0.982   |
| PackNet [59] | M     | 0.107   | 0.802  | 4.538 | 0.186   | 0.889   | 0.962   | 0.981   |
| Zhao et al. [11] | M     | 0.113   | 0.704  | 4.581 | 0.184   | 0.871   | 0.961   | 0.984   |
| Ours (baseline) | M     | 0.179   | 1.408  | 5.943 | 0.247   | 0.756   | 0.920   | 0.968   |
| Ours (sampling) | M     | 0.128   | 0.844  | 4.638 | 0.192   | 0.849   | 0.958   | 0.984   |
| Ours (L2-smooth) | M     | 0.117   | 0.786  | 4.559 | 0.183   | 0.868   | 0.962   | 0.985   |
| Ours (ωf = 1.0) | M     | 0.230   | 1.607  | 6.775 | 0.304   | 0.624   | 0.938   | 0.973   |
| Ours (ωf, L +) | M     | 0.124   | 0.788  | 4.555 | 0.187   | 0.857   | 0.960   | 0.985   |
| Ours (ωf, ωM) | M     | 0.118   | 0.798  | 4.534 | 0.185   | 0.866   | 0.961   | 0.985   |
| Ours (full) | M     | 0.118   | 0.787  | 4.488 | 0.183   | 0.870   | 0.962   | 0.985   |

![Fig. 4. Qualitative results on KITTI dataset. Images from top to bottom are original image, depth without MED, depth replacing $L_{\text{ref}}$ with raw photometric error, and depth with all components, respectively. Parts selected by red circles indicate the advantages of MED and $L_{\text{ref}}$.](image-url)
Fig. 5. Qualitative results on KITTI dataset. Images from top to bottom are original image, ground truth depth map using linear interpolation, depth from SC-SfMLearner [8], Zhao et al [11] and our model, respectively. Parts selected by red circles indicate the advantages of the proposed model.

TABLE II
QUANTITATIVE RESULTS. BRIGHT AND DARK DENOTE THE EVALUATION RESULTS FOR THE REGIONS OF IMAGE THAT PIXEL VALUE BIGGER AND SMALLER THAN 127, RESPECTIVELY. GAIN IS THE DIFFERENCE OF PERFORMANCES BETWEEN OUR MODEL WITH OR WITHOUT \( L_{rp} \).

| Method           | AbsRel ↓ | SqRel ↓ | RMS ↓  | RMSlog ↓ |
|------------------|----------|---------|--------|----------|
| Ours-bright (w/o \( L_{rp} \)) | 0.127    | 0.901   | 4.314  | 0.182    |
| Ours-bright (full) | 0.122    | 0.956   | 4.297  | 0.179    |
| gain             | 0.005    | 0.055   | 0.017  | 0.003    |
| Ours-dark (w/o \( L_{rp} \)) | 0.129    | 0.838   | 4.705  | 0.191    |
| Ours-dark (full)  | 0.122    | 0.823   | 4.622  | 0.187    |
| gain             | 0.007    | 0.016   | 0.083  | 0.004    |

(1) Ours (baseline): Depth-Net trained with normal photometric error, edge-aware smoothness \( L_s \) and point-wise depth loss \( L_{ba} \), without inlier score \( S_r \), and mask \( M_a \) to filter outlier.

(2) Ours (sampling): based on baseline model, this results added complete sampling strategies, and it gets a great performance boost.

(3) Ours (L2-smooth): training the model with all the components and using L2 smoothness loss \( L_s(d,1,2) \) instead of L1 smoothness loss \( L_s(d,1,1) \). Compared with L1 version, L2 has small performance degradation.

(4) Ours \((w_d = 1.0)\): Depth-Net trained with all the components without a small weight \( w \) set for the depth Jacobi matrix in the training phase (Sec. III-C). It can be seen that this small weight is important when embedding BA modules into a deep learning framework in self-supervised learning. Such a results may be due to the fact that changing \( w_d \) means that the degree of optimization of the LM algorithm for depth and ego-motion make the loss \( L_{rp} \) and \( L_{ba} \) have some conflict in synchronizing updating Depth-Net, and thus falling into local minima. It is noteworthy that this problem is not reported in supervised learning algorithms, such as BA-Net [48], so we speculate that this problem exists only in unsupervised learning without identified labels.

(5) Ours (w/o \( L_{rp} \)): Depth-Net trained with all the components without relative photometric error \( L_{rp} \). Fig. 4 shows the qualitative results of the depth prediction using the proposed method. It can be seen that the relative photometric error \( L_{rp} \) shows advantages in the dark regions. To further illustrate the effect of \( L_{rp} \), we evaluate the performance of the dark (pixels value smaller than 127) and bright (pixels value bigger than 127) areas separately and calculate the magnitude of the respective performance changes. As shown in Table II, it can be seen that training with \( L_{rp} \) is mostly better than the version without \( L_{rp} \), both in bright and dark circumstances. In addition, the gain item shows that performance enhancements of the loss \( L_{rp} \) in dark exceed those in bright regions.

(6) Ours (w/o MED): Depth-Net trained without MED. We can see that MED is better than Sigmoid function as the output activated process.

Comparison with state-of-the-art methods. The depth results are reported and compared with several state-of-the-art (SoTA) methods in Table I. The proposed model achieves comparable performance with the SoTA methods [11], [17], [59]. Note that Depth-Net cannot converge without the loss \( L_{ba} \). This is because using photometric error alone to train the network relies on the accuracy of the ego-motion estimation, which is difficult to obtain from the BA module with few iterations and poor initial depth. Fig. 5 shows the qualitative results of the depth prediction compared with Zhao et al. [11] and SC-SfMLearner [8]. It can be seen that the accuracy in some regions (the first and third columns) is better than Zhao et al. [11]. In addition, the absolute depth accuracy of our model is better than other methods in some areas (the second and fourth columns). This is probably due to that BA loss \( L_{ba} \) can obtain a relative accuracy depth in the later stages of training.

B. Optical flow evaluation

Ablation Study. The ablation study is presented in Table III. The metrics \( Noc \) and \( All \) are endpoint errors with non-occluded and all regions. Fl indicates percentage of erroneous
A MANUSCRIPT 8

Fig. 6. Quantitative results of optical flow robustness on KITTI 2015 [55] optical flow training set. The bottom and top x-coordinate indicate photometric $\delta_p$ and geometric $\delta_g$ noise intensity, respectively. The y-coordinate indicates evaluation metrics which were normalized to [0,1] using the corresponding scale.

Table III

| Method                  | KITTI 2012 | KITTI 2015 |
|-------------------------|------------|------------|
| UnFlow [63]             | 3.29       | 8.10       |
| Geonet [25]             | 8.05       | 10.81      |
| DF-Net [27]             | 3.54       | -          |
| CC [33]                 | -          | 5.66       |
| GLNet [32]              | -          | 4.86       |
| EPC++ [26]              | 3.30       | 3.84       |
| Zhao et al. [11]        | -          | 3.60       |
| PANet (full)            | 1.83       | 2.97       |
| EPC++ [26] (flow only)  | -          | 3.66       |
| Zhao et al. [11] (flow only) | -        | 4.96       |
| Ours w/o PANet (flow only) | 3.53      | 3.74       |
| PANet (flow only)       | **2.54**   | **3.57**   | **6.74**   | **19.87%** |

Table IV

| Method                  | KITTI 2012 | KITTI 2015 |
|-------------------------|------------|------------|
| UnFlow [63]             | 3.29       | 8.10       |
| Geonet [25]             | 8.05       | 10.81      |
| DF-Net [27]             | 3.54       | -          |
| CC [33]                 | -          | 5.66       |
| GLNet [32]              | -          | 4.86       |
| EPC++ [26]              | 3.30       | 3.84       |
| Zhao et al. [11]        | -          | 3.60       |
| PANet (full)            | 1.83       | 2.97       |
| EPC++ [26] (flow only)  | -          | 3.66       |
| Zhao et al. [11] (flow only) | -        | 4.96       |
| Ours w/o PANet (flow only) | 3.53      | 3.74       |
| PANet (flow only)       | **2.54**   | **3.57**   | **6.74**   | **19.87%** |

Fig. 7. Qualitative results on KITTI dataset. Images from top to bottom are original image, ground truth, optical flow estimated by Zhao et al. [11] and PANet, respectively.

In this part, two common cases of visual noise (geometric and photometric noise) are simulated using data synthesis. Note that we did not simulate the low FPS case because the KITTI optical flow dataset only provides test samples for two consecutive frames.

It is first necessary to simulate different noise scenarios. Geometric noise is synthesized the same as in [4]. Specifically, a low-frequency random map was separately generated $N_g: \Omega \rightarrow \mathbb{R}^2$ by up-sampling a $3 \times 3$ grid filled with uniformly distributed random values from $[-\delta_g; \delta_g]$, where we set $\delta_g \in \{1, 2, 3\}$ in the experiments. The original image was perturbed by shifting each pixel by $N_g(x)$. This procedure simulates noise originating from (unmodeled) rolling shutter or inaccurate geometric camera calibration, as shown in Fig. 4. Photometric noise is generated by Gaussian blurring with random strength per pixel. A high-frequency random blurmap $N_p: \Omega \rightarrow \mathbb{R}^2$ was separately created by up-sampling a $300 \times 300$ grid filled with uniformly distributed random values in $[-\delta_p; \delta_p]$, where $\delta_p \in \{1, 2, 3, 4, 5, 6\}$ in the experiments performed. The original image was then perturbed by adding anisotropic blur with standard deviation $N_p(x)$ to pixel $x$.

Fig. 6 shows the robustness evaluate, where in the legend, the symbols $P$ and $G$ denotes photometric and geometric noise. We can see that proposed PANet is substantially better than Zhao et al. [11] under all noise conditions. In addition, the PANet without third stage training outperforms the model without PANet.
Comparison with state-of-the-art methods. Table \( \text{III} \) shows the qualitative results of self-supervised optical flow estimation on the KITTI 2015 training set. Results show that the proposed PANet outperforms other methods on self-supervised optical flow estimation under the same configuration. In addition, the optical flow module can benefit from cross-task loss \( L_{\text{cross}} \) for joint learning. Fig. 7 shows several qualitative results. It can be seen that the proposed PANet outperforms PWC-Net \( \text{II} \) in some region, such as the car in bottom left corner.

C. Odometry evaluation

Setup. To evaluate the performance of our model, the translation error \( t_{\text{err}}(\%) \) and rotation error \( r_{\text{err}}(\degree/100\text{m}) \) over possible sub-sequences of length (100, 200, ..., 800) meters are reported with full trajectories. Since monocular VO lacks real-world scale information, the predicted trajectory is aligned to ground truth by using 7-degree-of-freedom (DoF) (scale + 6DoF) transformation.

Ablation study. The proposed model uses geometry-based algorithm to estimate odometry, thus, in this section we give an ablation analysis aiming at the VO inference. As we can see, the performance is already better than meany learning based methods, such as CC \( \text{III} \) and AdaptVO \( \text{VII} \).

(2) Ours (sampling): The full sampling strategy is used in the base of baseline version. Compared to the baseline, the complete sampling did not significantly improve performance in the case of using P3P to estimate pose.

(3) Ours (baseline + BA): Based on setting of Ours (baseline), geometric BA is used to infer the pose without P3P initialization and dynamic adjustment of damping factor \( \lambda \). We set the initial rotation matrix to be a unit matrix, the initial translation vector to be zeros, and the initial damping factor \( \lambda = 0.1 \), the same as in DSO \( \text{IV} \). The number of iterations is set to 30. Compared with P3P, BA performs better in our model, one reason is that the learned optical flow is much more accurate than the depth.

(4) Ours (sampling + BA): The full sampling strategy is used in the base of Ours (baseline + BA) version. From this result, we can find that the complete sampling strategy improves the performance of pose estimation more than simple sampling (flow occlusion mask \( M_{\text{f}} \)) after using the BA algorithm.

(5) Ours (P3P init): Based on Ours (sampling + BA) version, the method uses P3P as initialization. It can be seen that using P3P gives better results than fixed initialization, especially in terms of rotation error \( r_{\text{err}} (1.05 \text{ vs. } 0.32) \) for sequence 10.

(7) Ours \((w_d = 0.1)\): Based on Ours (full) version, the method sets \( w_d = 0.1 \) rather than 1.0 in inference phase. As we can see, contrary to the depth performance in the training phase (Table \( \text{I} \), setting \( w_d = 0.1 \) during inference leads to a decrease in results.

(6) Ours (full): The method uses a dynamic damping factor \( \lambda \) (see Equation \( 5 \)) instead of a fixed value 0.1 as initialization. As we can see, this component can further improve performance.

Robustness evaluation. Robustness is an important metric in a VO/SLAM system. It ensures that the system will not crash or large errors will not occur in various unconventional scenes. In this work, two common cases of visual noise and low FPS are simulated using data synthesis; see Fig. 1. The proposed method is compared with several representative algorithms, such as direct method DSO \( \text{IV} \), indirect method ORB-SLAM \( \text{III} \), the learning-based VO SC-SfMLearner \( \text{VII} \), and a joint geometry-learning training model \( \text{V} \). The synthesized geometric and photometric noise images are the same as Sec. \( \text{V-B} \). As for Low-FPS case, the original sequence was sampled separately with a big stride \( \delta_s \), where \( \delta_s \in \{2, 3\} \) was set in the experiment. Fig. 2 and Table \( \text{V} \) show the variation of different noise intensities, and qualitative results with \( \delta_g = 3, \delta_p = 6, \text{ and } \delta_s = 3 \), respectively. Figure 10 shows the trajectories in three noisy environments.

According to the analysis of \( \text{IV} \), ORB-SLAM is significantly more robust to geometric noise, while DSO performs better in the presence of strong photometric noise, because ORB-SLAM and DSO model geometric and photometric noise, respectively. The experiments performed in this work (Table \( \text{V} \)) demonstrate consistent results. For geometric noise, as expected, the learning based SC-SfMLearner \( \text{VII} \) shows a
The other learning-based VO methods, which includes a long-term model trained with 97 frames \cite{gao2021pwc} and the algorithms \cite{zhao2021learning} trained online on a test set. Compared with a pure geometry-based algorithm, such as ORB-SLAM \cite{mur2015orb} or DSO \cite{leutenegger2015dsosl}, the proposed model also achieves remarkable advantages in sequence 09.

**Performance on KAIST urban dataset.** To verify the performance of the model in more realistic traffic scenarios, we use KAIST urban dataset \cite{park2019deep} to further evaluate the accuracy of VO. This dataset has long mileage (3-12km), fast light changes, complex environments (including pedestrians, oncoming traffic and urban complexes) and motion closer to real driving.

Since this dataset did not include ground truths for depth and optical flow, we only qualitatively evaluate to give a few examples of depth and optical flow, while quantitatively evaluating the VO performance of the model. Figure 12 shows some visualization results, including original RGB image, correspondence inlier score $S_r$, depth, and optical flow. We can see that inlier score (Sec. IV-C) will filter some outer points, and the strong direct light environment (the third column) can affect optical flow and depth estimation.

Figure 13 and Table VI show the quantitative and qualitative experiments on VO estimation. To evaluate the generalization ability of our model, first, we directly use pre-trained model on the KITTI dataset to evaluate results on the KAIST urban dataset \cite{park2019deep}. As we can see, compared with pure learning method \cite{park2019deep}, the geometric BA module introduced by proposed model makes the model more generalizable due to that the model only learn low-level optical flow feature and are not sensitive to the overall distribution. Pure geometry methods, such as DSO \cite{leutenegger2015dsosl} and ORB-SLAM \cite{mur2015orb}, have failed to track in the tested sequences. In addition, the performance of the most sequences was further improved when the model was trained on KAIST data.

### Table V

**Qualitative VO results on KITTI odometry dataset of synthetic images with geometric noise $\delta_g = 3$, photometric noise $\delta_p = 6$, and low FPS $\delta_x = 3$.**

| Method          | Geometric noise | Photometric noise | Low FPS |
|-----------------|-----------------|-------------------|---------|
|                 | Seq. 09 | Seq. 10 | Seq. 09 | Seq. 10 | Seq. 09 | Seq. 10 | Seq. 09 | Seq. 10 |
| DSO \cite{leutenegger2015dsosl} | 10.55 | 0.59 | 8.10 | 1.30 |
| ORB-SLAM \cite{mur2015orb} | 28.47 | 6.27 | 14.60 | 6.55 |
| SC-SfMLearner \cite{censi2021s} | 41.26 | 21.87 | 47.99 | 24.02 |
| Zhao et al. \cite{zhao2021learning} | 28.47 | 6.27 | 14.60 | 6.55 |
| Ours (sampling) | 5.63 | 1.86 | 6.93 | 3.33 |
| Ours | 7.54 | 1.87 | 5.37 | 2.33 |

**Fig. 8.** The variation curve of VO performance with the number of iterations of the LM algorithm. Each color indicates a different number of sampling points.
Fig. 9. Scatter plot of VO performance under different noise environments and intensities. Each sub-plot is divided into three sub-regions, left-center-right, representing photometric noise, geometric noise and Low FPS respectively. Different color dots indicate different noise intensity, including $\delta_g$, $\delta_p$, and $\delta_s$. Note that the same noise intensity is indicated by the same color.

D. Generalization on indoor dataset

The challenges of the indoor environment are the existence of large texture-less regions, much more complex ego-motion (compared to the outdoor scene of driving along the road), and low dynamic range of optical flow. To further test the generalization ability of proposed model, we evaluate our method on a challenging indoor dataset: TMU-RGBD [61]. First, we evaluate our method on several sequences (Xue’s testing [47] split) of TMU-RGBD using the model pre-trained on KITTI dataset. Table VII shows the results of geometry based DSO [4] and ORB-SLAM [3], self-supervised joint learning method [11] and proposed method using pre-training on KITTI dataset (Ours - KITTI), fine-tuning on TMU-RGBD without BA module (Ours (sampling)), and with BA module (Ours). As we can see, the evaluation results pre-training on KITTI has satisfactory results. Such generalization ability is mainly due to that we learn only some underlying cues, such as optical flow and depth, instead of directly learning high-level visual representations. In most cases, the proposed model fine-tuned on TMU-RGBD dataset is better than Zhao et al. [11], while, in some cases, the proposed model gets some poor results. We visualized some results, including good and failed cases shown in Fig. 14 and find that the results fine-tuned on TMU-RGBD tends to consider the bottom of the image as near (the third row), which is one of the differences between KITTI traffic data and indoor data, however, on the sequence fr2/desk, the desk should be the nearest rather than the bottom floor. In addition, the TMU-RGBD dataset moves very slowly, resulting in insufficient differentiation of the optical flow over the whole image, which makes the re-projection error $L_{rp}$ and BA loss...
TABLE VI

Quantitative results. Comparison of proposed method to existing methods on KAIST urban dataset [60]. The reported results is absolute trajectory error.

| Method       | Training data | urban27 (5.4km) | urban28 (11.47km) | urban31 (11.4 km) | urban33 (7.6 km) | urban35 (3.2km) | urban39 (11.06 km) |
|--------------|---------------|----------------|------------------|------------------|-----------------|----------------|-------------------|
| DSO [4]      | -             | X              | X                | X                | X               | X              | X                 |
| ORB-SLAM [3] | -             | X              | X                | X                | X               | X              | X                 |
| SC-SfMLearner [8] | KITTI       | 312.86         | 412.80           | 958.54           | 480.39          | 276.81         | 358.29            |
| Zhao et al. [11] | KITTI       | 137.64         | 186.23           | 683.83           | X               | 122.33         | 443.09            |
| Ours (sampling) | KITTI       | 332.34         | 315.43           | 980.14           | 356.98          | 101.85         | 284.34            |
| Ours | KITTI | 98.32 | 212.08 | 339.23 | 93.58 | 85.64 | 143.82 |

TABLE VII

Quantitative results. Comparison of proposed method to existing methods on TMU-RGBD dataset [61]. The reported results is absolute trajectory error. Ours - KITTI denotes the result obtained by KITTI pre-training model. The abbreviations nstr., str., ntex, and tex. indicate non-structure, structure, non-texture, texture, respectively. Bolded and underlined numbers is best and second-best metrics except geometric algorithms DSO [4] and ORB-SLAM [3].

| Method | DSO [4] | ORB-SLAM [3] | Zhao et al. [11] | Ours - KITTI | Ours (sampling) | Ours |
|--------|---------|-------------|-----------------|-------------|----------------|------|
| fr2/desk | X       | 0.653       | 1.376           | 1.393       | 1.482          |      |
| fr2/pioneer 360 | X       | X           | 1.408           | 0.202       | 0.144          | 0.154 |
| fr2/pioneer slam3 | 0.737   | X           | 1.430           | 1.665       | 1.375          | 1.842 |
| fr2/large cabinet | X       | X           | 2.165           | 0.353       | 0.440          | 0.385 |
| fr2/sitting static | 0.082   | X           | 0.017           | 0.015       | 0.023          | 0.017 |
| fr3/str. ntex. near loop | X       | X           | 1.305           | 1.382       | 0.775          | 0.774 |
| fr3/str. tex. near loop | 0.093   | 0.057       | 1.294           | 0.735       | 0.533          | 0.575 |
| fr3/str. far | 0.543   | X           | 0.265           | 0.280       | 0.226          | 0.211 |
| fr3/str. tex. far | 0.040   | 0.018       | 0.165           | 0.366       | 0.140          | 0.138 |

**Fig. 10.** Visualization of trajectories results on synthetic KITTI: sequence 09 (left) and sequence 10 (right) plotted deep-learning- and geometry-based methods.

**Fig. 11.** Visualization of trajectories results on KITTI dataset: sequence 09 (left) and sequence 10 (right) against deep-learning-based and geometry-based methods.

**E. Evaluation time**

To evaluate the running time of our method, we record the average and standard deviation of time in millisecond (ms) and summarize the results corresponding to each component in our pipeline. As shown in Table VIII, our method takes 411.46 ms to inferre VO with two $256 \times 832$ images on GTX 3090 24G and Intel Xeon(R) 6230 2.10GHz. Since the algorithm uses both the GPU (deep network) and the CPU (computing the essential matrix and P3P with OpenCV library), we use the `torch.cuda.synchronize()` function to synchronize the execution time of the CPU and GPU, and calculate the mean and

**TABLE VIII**

Evaluation time (ms) for each component. The mean and standard deviation of the 100 replicate experiments were calculated on GTX 3090 24G and Intel Xeon(R) 6230 2.10GHz.

| Component | PANet | Depth-Net | Mask | Sample | BA | Total |
|-----------|-------|-----------|------|--------|----|-------|
|           | 86.11 | 7.01      | 12.62| 181.69 | 124.03 | 411.46 |
|           | (38.03)| (0.81)    | (0.165)| (0.366)| (0.140)| (0.138)|
Fig. 12. Qualitative results on KAIST dataset. Images from top to bottom are original image, inlier score $S_r$, depth, and optical flow, respectively.

Fig. 13. Visualization of trajectories on KASIT urban dataset. The suffix KITTI in legend denotes the estimated trajectories training on KITTI dataset.

The current computation bottlenecks include three aspects: (1) the optical flow needs to be calculated twice. In our model, the forward and backward optical flow need to be computed for subsequent masking and sampling. If an algorithm can be designed to obtain stable sampling points by calculating the forward optical flow only once, this will improve the speed of the model inference. (2) the sampling process. As mentioned in Sec. III-C, multiple sampling was used to ensure the stability of the matching points. To achieve stable sampling, we use the CPU to compute the essential matrix, which leads to additional communication cost between GPU and CPU. (3) BA optimization. The BA algorithm is actually very fast after careful optimization, for example, the photometric BA of DSO [4] can also be calculated in real time on the CPU. The BA module in this paper does not further optimize due to debugging and visualization convenience. In summarize, if we want to run the model in real time, further improvements and optimizations are needed in these three areas.

F. Discussion

VO performance mainly depends on 1) the accuracy of detecting and matching, and 2) follow-up VO estimates. Pure geometry-based algorithms lack sufficient fineness and robustness in pixel detection and matching, while a pure learning-based model performs poorly on the ego-motion estimate due to the lack of a geometry guarantee. Under more difficult conditions, these disadvantages are magnified; see Table VII and Fig. 10 [13]. Therefore, the model that learns depth and optical flow and predicts ego-motion using a geometry-based approach is perhaps one of the directions to take to solve the robustness problem. In the proposed model, the gain on robustness comes from two novel designs: 1) the accuracy and robust optical flow, and 2) the BA module, in which CNN-based optical flow is a more robust visual cue compared with both relative pose estimation for deep-learning methods and conventional feature-point detection and matching. With the
improvements made in the proposed method, the geometric BA module can be directly embedded into the joint learning model and achieves remarkable depth, optical flow, and VO performance.

The joint geometry-learning algorithm has more advantages in generalization ability than the pure learning algorithm, see Table V, VI. This may be due to that the pure learning algorithm needs to learn high-level information, such as ego-motion, while the joint learning algorithm only needs to learn the local low-level image features (optical flow and depth), which vary relatively little across datasets.

However, the method does not perform well for some weakly textured scenes (like Fig. 14 the fourth column), which is a challenge for the current VO algorithm. In addition, learning a good depth is especially difficult for those optical flow with low dynamic range, because the low dynamic range means that the geometric BA module does not learn a large enough range of inverse depth, and under such conditions, the self-supervision losses ($L_{rp}$ and $L_{ba}$) has difficulty learning a good result.

V. CONCLUSIONS

In this paper, a novel optical flow net, named PANet is proposed, which significantly improves performance on self-supervised learning, and robustness under the condition of geometric noise. For better training Depth-Net and predicting ego-motion, a novel system is advanced that combines Depth-Net, Flow-Net, and a geometric BA module for self-supervised training. The experiments show that the system not only improves the performance of depth, optical flow, and VO, but also significantly improves robustness under various challenging conditions, e.g., geometric noise, photometric noise, and low FPS.

ACKNOWLEDGMENT

The authors would like to thank...

REFERENCES

[1] G. N. DeSouza and A. C. Kak, “Vision for mobile robot navigation: A survey,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 24, no. 02, pp. 237–267, 2002.
[2] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, “Dtam: Dense tracking and mapping in real-time,” in ICCV, 2011.
[3] R. Mur-Artal and J. D. Tardós, “ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras,” IEEE Trans. Robot., vol. 33, no. 5, pp. 1255–1262, 2017.
[4] J. Engel, V. Koltun, and D. Cremers, “Direct sparse odometry,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 40, no. 3, pp. 611–625, 2018.
[5] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in CVPR, 2016.
[6] Y. J. Cao, C. Lin, and Y. J. Li, “Learning crisp boundaries using deep refinement network and adaptive weighting loss,” IEEE Trans. Multimedia, vol. 23, pp. 761–771, 2021.

[7] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, “Unsupervised learning of depth and ego-motion from video,” in CVPR, 2017.

[8] J.-W. Bian, Z. Li, N. Wang, H. Zhan, C. Shen, M.-M. Cheng, and I. Reid, “Unsupervised scale-consistent depth and ego-motion learning from monocular video,” in NeurIPS, 2019.

[9] N. Yang, L. Stumberg, R. Wang, and D. Cremers, “D3vo: Deep depth, deep pose and deep uncertainty for monocular visual odometry,” in CVPR, 2020.

[10] H. Zhan, C. S. Weerasekera, J. Bian, and I. Reid, “Visual odometry revisited: What should be learnt?” in ICRA, 2019.

[11] W. Zhao, S. Liu, Y. Shu, and J.-Y. Liu, “Towards better generalization: Joint depth-pose learning without posenet,” in CVPR, 2020.

[12] N. Yang, R. Wang, X. Gao, and D. Cremers, “Challenges in monocular visual odometry: Photometric calibration, motion bias, and rolling shutter effect,” IEEE Robotics and Automation Letters, vol. 3, no. 4, pp. 2878–2885, 2018.

[13] X. Gao, T. Zhang, Y. Liu, and Q. Yan, 14 Lectures on Visual SLAM: From Theory to Practice. Publishing House of Electronics Industry, 2017.

[14] G. Farnebäck, “Two-frame motion estimation based on polynomial expansion,” in Image Analysis, J. Bigun and T. Gustavsson, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 363–370.

[15] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, “PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume,” in CVPR, 2018.

[16] Y. Wang, Y. Yang, Z. Yang, L. Zhao, P. Wang, and W. Xu, “Occlusion aware unsupervised learning of optical flow,” in CVPR, 2018.

[17] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow, “Digging aware unsupervised learning of optical flow,” in CVPR, 2018.

[18] N. Mayer, E. Ilg, P. Fischer, C. Hazirbas, V. Golkov, P. v. d. Smagt, D. Cremers, and T. Brox, “Flownet: Learning optical flow with convolutional networks,” in ICCV, 2015.

[19] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, “A naturalistic open source movie for optical flow evaluation,” in ECCV, 2012.

[20] D. Eigen, P. A. F. Geiger, S. Pillai, A. Raventos, and A. Gaidon, “3d monocular slam,” in ICCV, 2019.

[21] D. Eigen, C. Puhrsch, and R. Fergus, “Depth map prediction from a single image using a multi-scale optical flow network,” in NeurIPS, 2014.

[22] J. Gonzalez and M. Kim, “Forget about the lidar: Self-supervised depth estimation from monocular video,” in CVPR, 2019.

[23] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski, “A database and evaluation methodology for optical flow,” International Journal of Computer Vision, vol. 92, no. 1, pp. 1–31, 2011.

[24] N. Yang, R. Wang, J. Stueckler, and D. Cremers, “Deep virtual stereo,” in CVPR, 2016.

[25] Z. Yin and J. Shi, “Geonet: Unsupervised learning of dense depth, optical flow and camera pose,” in CVPR, 2018.

[26] J. Li, R. Klein, and A. Yao, “A two-streamed network for estimating fine-scale depth maps from single rgb images,” in CVPR, 2017.

[27] D. Xu, E. Ricci, W. Ouyang, X. Wang, and N. Sebe, “Multi-scale continuous crfs as sequential deep networks for monocular depth estimation,” in CVPR, 2017.

[28] J. Li, R. Klein, and A. Yao, “A two-streamed network for estimating fine-scale depth maps from single rgb images,” in CVPR, 2017.

[29] D. Xu, E. Ricci, W. Ouyang, X. Wang, and N. Sebe, “Multi-scale continuous crfs as sequential deep networks for monocular depth estimation,” in CVPR, 2017.

[30] Y. Zou, P. Ji, Q.-H. Tran, J.-B. Huang, and M. Chandraker, “Learning monocular visual odometry via self-supervised long-term modeling,” in ECCV, 2020.

[31] S. Li, W. Xin, Y. Cao, F. Xue, Z. Yan, and H. Zha, “Beyond tracking: Selecting memory and refining poses for deep visual odometry,” in CVPR, 2019.

[32] G. Farnebäck, “Two-frame motion estimation based on polynomial expansion,” in Image Analysis, J. Bigun and T. Gustavsson, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 363–370.

[33] D. Hoiem, A. A. Efros, and M. Hebert, “Recovering surface layout from an image,” International Journal of Computer Vision, vol. 61, no. 1, pp. 151–172, 2004.

[34] A. G. Schwing, S. Fidler, M. Pollefeys, and R. Urtasun, “Box in the box: Joint 3d layout and object reasoning from single images,” in ICCV, 2013.

[35] D. Eigen and R. Fergus, “Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture,” in ICCV, 2015.

[36] J. Li, D. Eigen, and Y. J. Li, “Learning to close the depth gap between raw images and dense maps,” in CVPR, 2017.

[37] D. Xu, E. Ricci, W. Ouyang, X. Wang, and N. Sebe, “Multi-scale continuous crfs as sequential deep networks for monocular depth estimation,” in CVPR, 2017.

[38] Z. Yin and J. Shi, “Geonet: Unsupervised learning of dense depth, optical flow and camera pose,” in CVPR, 2018.

[39] S. Liu, M. Lyu, I. King, and J. Xu, “Selfflow: Self-supervised learning of optical flow,” in CVPR, 2019.

[40] S. Agarwal, N. Snavely, S. M. Seitz, and R. Szeliski, “Bundle adjustment in the large,” in ECCV, 2010.

[41] J. Janai, F. Gümey, A. Ranjan, M. Black, and A. Geiger, “Unsupervised learning of multi-frame optical flow with occlusions,” in ECCV, 2018.

[42] J. Engel, T. Schöps, and D. Cremers, “Lsd-slam: Large-scale direct monocular slam,” in ECCV, 2014.

[43] S. Meister, J. Hur, and S. Roth, “Unflow: Unsupervised learning of optical flow with a bidirectional census loss,” in AAAI, 2018.

[44] N. Mayer, E. Ilg, P. Fischer, C. Hazirbas, D. Cremers, A. Dosovitskiy, and T. Brox, “What makes good synthetic training data for learning disparity and optical flow estimation?” International Journal of Computer Vision, vol. 126, no. 9, pp. 942–960, 2018.

[45] Y. Zou, P. Ji, Q.-H. Tran, J.-B. Huang, and M. Chandraker, “Learning monocular visual odometry via self-supervised long-term modeling,” in ECCV, 2020.

[46] S. Li, W. Xin, Y. Cao, F. Xue, Z. Yan, and H. Zha, “Self-supervised monocular visual odometry with online adaptation,” in CVPR, 2020.

[47] P. Liu, M. Lyu, I. King, and J. Xu, “Selfflow: Self-supervised learning of optical flow,” in CVPR, 2019.

[48] S. Guizilini, R. Ambrus, S. Pillai, A. Raventos, and A. Gaidon, “3d matching for self-supervised monocular depth estimation,” in CVPR, 2019.

[49] J. Gonzalez and M. Kim, “Forget about the lidar: Self-supervised depth estimators with med probability volumes,” in NeurIPS, 2020.

[50] S. Pillai, R. Ambrus, A. Angulo, and C. Stiller, “Spherical self-supervised depth estimation,” in NeurIPS, 2020.

[51] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, “A naturalistic open source movie for optical flow evaluation,” in ECCV, 2012.

[52] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: The kitti dataset,” International Journal of Robotics Research, 2013.

[53] E. Horn, B. K. Horn, and B. G. Schunck, “Determining optical flow,” Artificial Intelligence, vol. 17, no. 1, pp. 185 – 203, 1981.

[54] R. Mahjourian, M. Wicke, and A. Angelova, “Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints,” in CVPR, 2018.

[55] Y. Chen, C. Schmid, and C. Sminchisescu, “Self-supervised learning with geometric constraints in monocular video: Connecting flow, depth, and camera motion,” in CVPR, 2019.

[56] A. Ranjan, V. Jampani, L. Balles, D. Sun, K. Kim, J. Wulff, and M. J. Black, “Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation,” in CVPR, 2019.
Michael Shell Biography text here.