Improving Learning-based Ego-motion Estimation with Homomorphism-based Losses and Drift Correction

Xiangwei Wang, Daniel Maturana, Shichao Yang, Wenshan Wang, Qijun Chen and Sebastian Scherer

***Abstract*— Visual odometry is an essential problem for mobile robots. Traditional methods for solving VO mostly utilize geometric optimization. While capable of achieving high accuracy, these methods require accurate sensor calibration and complicated parameter tuning to work well in practice. With the rise of deep learning, there has been increased interest in the end-to-end, learning-based methods for VO, which have the potential to improve robustness. However, learning-based methods for VO so far are less accurate than geometric methods. We argue that one of the main issues is that the current ego-motion estimation task is different from other problems where deep learning has been successful such as object detection. We define a novel cost function for learning-based VO considering the mathematical properties of the group homomorphism. In addition to the standard L2 loss, we incorporate losses based on the identity, inverse and closure properties of SE(3) rigid motion. Furthermore, we propose to reduce the VO drift by estimating the drivable regions using semantic segmentation and incorporate this information into a pose graph optimization. Experiments on KITTI datasets show that the novel cost function can improve ego-motion estimation compared to the state-of-the-art and the drivable region-based correction further reduces the VO drift.**

I. Introduction

Visual odometry (VO) takes a sequence of consecutive images as input and estimates the motion of a moving robot (ego-motion). This problem was first defined by Nister *et al.* [1] in 2004, but related research can be traced back almost 40 years [2]. It is one of the main methods used in robot localization and state estimation. Most of the existing methods estimate the ego-motion $(\mathbf{R}, \mathbf{t})$ by geometric optimization of reprojection error [3] or photometric error [4]. These approaches require accurate sensor calibration and manual parameter tuning to work in different environments.

Recently, learning-based methods which directly estimate the ego-motion have become active research areas. Convolutional Neural Networks (CNNs) [5], either supervised or unsupervised, are most widely used to learn a mapping function from image pairs to ego-motion. In supervised methods [6], the ground-truth ego-motion is directly used to compute a training loss, while unsupervised methods [7] usually estimate both ego-motion and image depth to formulate an image reprojection-based loss.

This work was completed when Xiangwei Wang (xiangwew@cs.cmu.edu) was a visiting Ph.D student in Carnegie Mellon University. Daniel Maturana, Shichao Yang, Wenshan Wang and Sebastian Scherer are with the Robotics Institute, Carnegie Mellon University and Qijun Chen is with the Department of Control Science and Engineering, Tongji University. This work was supported in part by Yamaha Motor Co., Ltd., National Natural Science Foundation of China (Key Program) (Grant No. 61573260, 61733013) and China Scholarship Council.

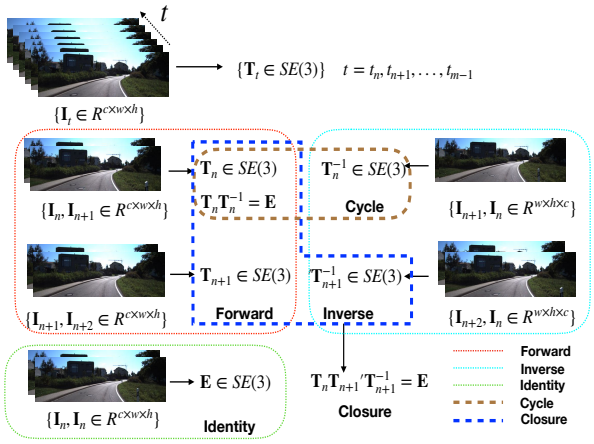


Fig. 1. Our training losses. We take the mathematical properties of homomorphism mapping into consideration and formulate three losses: identity loss, inverse loss and closure loss.

Most learning-based methods model the VO problem as a regression from pairs of images to rigid motions. Currently, the accuracy of these methods is worse compared to geometry-based approaches. We argue that this is partly because these approaches do not fully model the structure of the VO problem. In particular, they do not explicitly model certain properties that the VO mapping should have, such as the fact that for a given pair of images, and the same pair of images in reverse order, the estimated motions should be the inverse of each other. This makes the problem ill-defined.

In this paper, we propose to redefine the task by incorporating the intuition that the mapping should respect the mathematical *group* property of the rigid motion, and moreover, for operations on the input image pair, the output motion should change in a way that is consistent with the nature of the operation as shown in Fig. 1. That is, the mapping should have the properties of a *homomorphism*. Here, we focus on preserving the group properties of *closure*, *identity element*, and *inverse element*. For example, when reversing an image pair, the requirement that the predicted motion should be the inverse corresponds to preserving the *inverse element* property. We implement this idea by incorporating novel losses into the training process of an ego-motion estimation network, as detailed in Sec. III. While we focus on the supervised setting, our method can be used for unsupervised settings as well.

Another problem with frame-to-frame approaches is that they are subject to long-term drifting; as an additional contribution, we propose a novel method to address this issue

in driving scenarios, based on the idea that the estimated poses should follow a plausible path relative to the road on which the vehicle is travelling; we implement this using image-based semantic segmentation and graph optimization.

In this paper, we study 1) whether the CNNs can learn the ego-motion estimation; 2) how to define the suitable loss function for CNN training based on the properties of homomorphisms; 3) how to reduce the drift of the estimated VO; and 4) the empirical performance of our proposed improvements with real data. Thus, our contributions are as follows:

1) We empirically show that the CNNs with an *L2* loss do not learn a mapping that respects the properties of ego-motion.
2) We propose novel loss functions for CNNs based ego-motion estimation considering the mathematical properties of the homomorphism mapping.
3) We propose a pose graph optimization method to improve the ego-motion result considering the vehicle's drivable regions.
4) We show that with our proposed improvements, we can outperform the state of the art on learning-based monocular VO in the well-known KITTI benchmark.

The rest of this paper is organized as follows: Sec. II reviews the related work. Sec. III describes our approach, and Sec. IV evaluates our approach on the KITTI dataset. We conclude and suggest future work in Sec. V.

II. Related Work

The first learning-based ego-motion estimation method is proposed by Roberts *et al.* [8], who formulated the problem as a classification task solved by K-Nearest Neighbors. However, ego-motion estimation is not a classification but a regression problem in nature. Guizilini *et al.* [9] presented a semi-parametric method based on Gaussian Processes and optical flow input to solve the regression problem. Costante *et al.* [10] utilized optical-flow to predict the ego-motion with deep CNNs. Pillai *et al.* [11] further combined it with a GPS/INS system. For the methods based on optical flow, their performance depends on the accuracy of the input and they discard much information in the raw images. Costante *et al.* [12] proposed to jointly estimated a low dimensional representation of optical flow and ego-motion to make the ego-motion estimation more robust to optical flow change. Wang *et al.* [6], [13] used raw images as the input and also considered sequential information by using recurrent CNNs. This method also exploited some properties of ego-motion, while we directly use the mathematical constraints for the loss functions. Iyer *et al.* [14] proposed a self-supervised method where the ego-motion estimation is learned from a geometry-based method, and which enforces geometric consistency of the trajectory. Our homomorphism loss is similar to this method, but we consider the identity, inverse and closure properties of the homomorphism mapping, whereas this method considers only the closure property.

Meanwhile, many unsupervised methods are proposed to make the training process easier. Zhou *et al.* [7] proposed an unsupervised method to predict image depth as well as ego-motion using two networks, then compute the re-projected image residual as the loss function. Mahiourian *et al.* [15] added an additional 3D geometric loss based on [7] and achieved better performance. Li *et al.* [16] adopted binocular images to obtain more supervised information during training but used only monocular images during testing. Zhan *et al.* [17] also used binocular images to train the network, and besides, they utilized a dense feature reconstruction loss as supervisory information in addition to image reconstruction loss, in order to relax the Lambertian scene assumption. Yin *et al.* [18] considered dynamic objects in the environment and combined ResFlowNet with ego-motion and depth estimation networks to get better performance on all tasks. Parisotto *et al.* [19] proposed a learning-based SLAM method which not only estimated the ego-motion but also considered the global optimization. In the field of drift correction, Peretroukhin *et al.* [20] proposed a pose correction method to correct the ego-motion estimated by a geometry-based method using graph optimization method. While our method is to correct a learning-based method by semantic information.

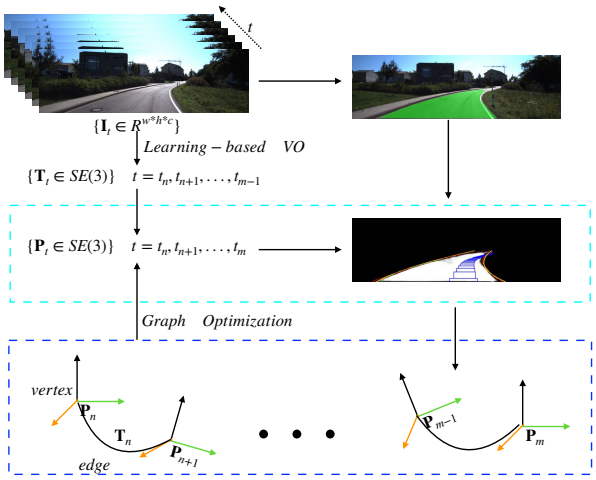


Fig. 2. System structure. The estimated ego-motion and road segmentation information are combined to improve the ego-motion accuracy by graph optimization.

III. Method

In this section, we first introduce and motivate our proposal for training ego-estimation network based on the properties of homomorphisms. We then present our drift correction method. An overview of our framework is described in Algorithm 1 and Fig. 2.

A. Learning-based Ego-motion Estimation as a Homomorphism

Learning-based methods formulate ego-motion estimation as a regression problem by modeling the many-to-one mapping from two consecutive images to ego-motion

$$\mathbf{F} : \mathbb{R}^{2c \times w \times h} \rightarrow se(3) \tag{1}$$

Algorithm 1: Ego-motion Estimation and Correction

Input: Supervised training dataset: monocular image pairs and ego-motion $\{\hat{\mathbf{I}}_{\mathbf{i}}^{i+1}, \hat{\mathbf{T}}_i^{i+1}\}$ $i = 0, ..., n$; image pairs $\{\mathbf{I}_j^{j+1}\}$, $i = 0, ..., m$ for testing

Output: Corresponding ego-motion with $\{\mathbf{I}_j^{j+1}\}$

Train the neural network with the training dataset to fit the function Eq. 2 with the loss function Eq. 12

for $t_i = 0$ *to* m **do**

  Estimate the ego-motion $\mathbf{T}_{t_i}$ with the trained neural network as described in Section III-A.3

  Detect the road region in image $\mathbf{I}_{t_i}$ by semantic segmentation of the road

  Fit the road edges with second order polynomial equations $\mathbf{f}_l$ and $\mathbf{f}_r$

  if $t_i > minimum\_sequence\_length(t_l)$ **then**

    Optimize the egomotion $\mathbf{T}_j^{j+1}$ $j = t_i - t_l, ..., t_i$ as described in Eq. 20

  end

end

return $\mathbf{T}_j^{j+1}$ $j = 0, ..., m$

The domain of the mapping has dimensionality $2c \times w \times h$, where c is the number of channels in each image and w, h are the width and height of the input image respectively.

However, the mapping is ill-defined as the constraints of ego-motion are not considered. Instead, we propose to redefine the learning-based ego-motion estimation formulation:

> Learning-based ego-motion estimation is the process of training a model to find the mapping from consecutive images to the ego-motion, if and only if the mapping is a homomorphism.

1) Mathematical Losses: As described in Section III-A, the problem is formulated as a regression problem. The mapping can be denoted as the function

$$\{\mathbf{t}, \mathbf{r}\} = \mathbf{F}(\mathbf{I}_m^n; P) \tag{2}$$

where $\mathbf{F}$ is the neural network with parameters P, $\mathbf{I}_m^n = \{\mathbf{I}_m, \mathbf{I}_n\}$ are the two consecutive images, $\{\mathbf{t}, \mathbf{r}\} \in se(3)$ are translation and rotation vectors respectively. We consider consecutive image pairs as a general set $\{\mathbf{I}\}$, and define an operation $\otimes$ as

$$\mathbf{I}_m^n = \mathbf{I}_m^t \otimes \mathbf{I}_t^n \tag{3}$$

Then the general set $\{\mathbf{I}\}$, together with the defined operation is a general group $\{\mathbf{I}, \otimes\}$. We can know that the general group satisfies the four conditions: closure, associativity, identity and invertibility. We use the term *general group* instead of *group* because the operation $\otimes$ can only be used on consecutive elements in the set. We denote the identity element of the group as $\mathbf{I}_{iden} = \{\mathbf{I}_n, \mathbf{I}_n\} = \mathbf{I}_n^n$ and the inverse element of $\mathbf{I}_m^n$ as $-\mathbf{I}_m^n = \{I_n, I_m\} = \mathbf{I}_n^m$.

Taking the properties of ego-motion estimation into consideration, the mapping from $\{\mathbf{I}, \otimes\}$ to ego-motion group $SE(3)$ is a homomorphism

$$\exp(\mathbf{F}(\mathbf{I}_m^n \otimes \mathbf{I}_n^t)) = \exp(\mathbf{F}(\mathbf{I}_m^n)) * \exp(\mathbf{F}(\mathbf{I}_n^t)) \tag{4}$$

where $\mathbf{T} = \exp(\{\mathbf{t}, \mathbf{r}\})$ is the Lie group function which maps a motion vector $\{\mathbf{t}, \mathbf{r}\} \in se(3)$ to a motion matrix $\mathbf{T} \in SE(3)$ [21].

We can get three constraints based on the homomorphism:

$$\mathbf{F}(I_{iden}) = \{\mathbf{0}, \mathbf{0}\} \tag{5}$$

$$\exp(\mathbf{F}(-\mathbf{I}_m^n)) * \exp(\mathbf{F}(\mathbf{I}_m^n)) = \mathbf{E} \tag{6}$$

$$\exp(\mathbf{F}(\mathbf{I}_m^n)) * \exp(\mathbf{F}(\mathbf{I}_n^t)) = \exp(\mathbf{F}(\mathbf{I}_m^t)) \tag{7}$$

where $\mathbf{E}$ is the identity matrix in $SE(3)$.

Taking the above properties into consideration, we can formulate three corresponding losses: identity loss, inverse loss and closure loss. We first introduce the commonly used L2 loss [6]

$$\mathscr{L}_2 = \sum_{i=1}^{n} \|\mathbf{F}(I_i^{i+1}) - \{\mathbf{t}_{\mathbf{g}_i}, \mathbf{r}_{\mathbf{g}_i}\}\|_2 \cdot \boldsymbol{\omega} \tag{8}$$

where $\{\mathbf{t}_\mathbf{g}, \mathbf{r}_\mathbf{g}\}$ are the ground truth translation vector and $\boldsymbol{\omega}$ is the weights vector.

The identity loss in Eq. 9 indicates that for two identical images, the output translation and rotation should be zero.

$$\mathscr{L}_{iden} = \sum_{i=1}^{n} \|\mathbf{F}(I_i^i)\|_2 \cdot \boldsymbol{\omega} \tag{9}$$

The inverse loss encodes the constraint that when the order of the two input images are reversed, the output ego-motion should be accordingly inverse:

$$\mathscr{L}_{inv} = \sum_{i=1}^{n-1} \|\log(\exp(\mathbf{F}(I_i^{i+1})) * \exp(\mathbf{F}_{i+1}^i))\|_2 \cdot \boldsymbol{\omega} \tag{10}$$

where $\{\mathbf{t}, \mathbf{r}\} = \log(\mathbf{T})$ is the function which maps a motion matrix $\mathbf{T} \in SE(3)$ to a motion vector $\{\mathbf{t}, \mathbf{r}\} \in se(3)$.

The closure loss represents that given a triplet of consecutive images, the composition of motions corresponding to the first two images and the last two images should be consistent with the motion between the first and last image:

$$\mathscr{L}_{clo} = \sum_{i=1}^{n-2} \|\log(l_{i,i+1,i+2})\|_2 \cdot \boldsymbol{\omega} \tag{11}$$

$$l_{i,i+1,i+2} = \exp(\mathbf{F}(I_i^{i+1})) * \exp(\mathbf{F}(I_{i+1}^{i+2})) * \exp(\mathbf{F}(I_i^{i+2}))^{-1}$$

The total loss is thus

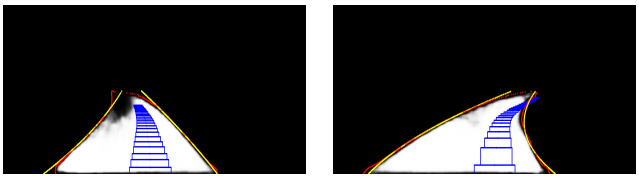
$$\mathscr{L} = \mathscr{L}_2 + \mathscr{L}_{iden} + \mathscr{L}_{inv} + \mathscr{L}_{clo} \tag{12}$$

We simplely set the weight of each loss to 1 though better performance may be achieved by adjusting the weight parameters.

2) Learning Setup: The network structure is adapted from PoseNet in SFM Learner [7]. We modify the convolutional layer to use dilated convolutions, to increase the receptive field. The input is the downsampled grayscale images. The output is normalized motion vector

$$\{\mathbf{r}, \mathbf{t}\} = (\{\mathbf{r}, \mathbf{t}\} - \{\bar{\mathbf{r}}, \bar{\mathbf{t}}\})/\sigma(\{\mathbf{r}, \mathbf{t}\}) \tag{13}$$

where $\{\bar{\mathbf{r}}, \bar{\mathbf{t}}\}$ and $\sigma(\{\mathbf{r}, \mathbf{t}\})$ are the mean values and standard errors of a motion vector. The weight parameters $\boldsymbol{\omega}$ of the different losses are simply set as $(1, 1, 1, 1, 1, 1, 1)$.



(a) Correct Estimated Ego-motion (b) Wrong Estimated ego-motion

Fig. 3. We projected a sequence of ego-position calculated by estimated ego-motion onto the beginning image of the sequence. This figure shows different conditions when the ego-motion is calculated correct (Fig. 3(a)) or wrong (Fig. 3(b)).

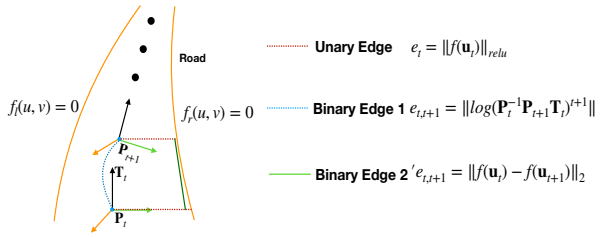


Fig. 4. This figure presents the definition of the graph for ego-motion correction. The vertices are the poses $\mathbf{P}_t$, and we define three kinds of edge to correct the ego-motion.

3) Estimation Method: In the training phase, we add the relative loss function (Eq. 12) to make the estimated mapping maintain the homomorphism properties. This not only improves the accuracy of the estimation, but also provides an indicator which shows how much the estimated result can be trusted, as follows. We build a pose graph where each edge e_{ij} encodes a predicted ego-motion $\mathbf{T}_i^j$ between the poses $\mathbf{P}_i$ and $\mathbf{P}_j$ of two images:

$$e_{ij} = \|\mathbf{P}_i \mathbf{T}_i^j \mathbf{P}_j^{-1}\|_2 \tag{14}$$

$$\mathbf{P}_t, \mathbf{P}_{t-1}, \mathbf{P}_{t-2} = \text{argmin} \sum_{i,j \in t,t-1,t-2} e_{ij} \tag{15}$$

Here we use the same information matrix for each estimated pose, as we do not have prior information. For each timestamp, the final poses and ego-motion are inferred by graph optimization, and the final loss can be used as a confidence measure in the accuracy of our estimations.

B. Ego-motion Path Correction

The ego-motion estimation in the last section is a completely open loop estimation whose performance is heavily dependent on the quality of the training procedure, and may have drift in the long term. Therefore, we propose to combine drivable region detection and ego-motion estimation to reduce the drift as post-processing. These two tasks are connected by the robot's localization: one predicts reasonable locations of the robot in the future, and the other calculates the current location. By incorporating temporal context, the two approaches can be combined to boost accuracy.

As we have estimated the ego-motions $\{\mathbf{T}_t^{t+1} \in SE(3)\}\, t = t_n, t_{n+1}, ..., t_m$, we can calculate the robot poses at each time by $\mathbf{P}_{t+1} = \mathbf{P}_t * \mathbf{T}_t^{t+1}$.

TABLE I
TESTING RESULTS OF DIFFERENT TRAINING METHODS

sequence	Forward Training	Cycle Training	Group Training
09 R(deg/m)	0.0289	0.0177	**0.0151**
09 t(%)	16.52	8.62	**8.04**

As shown in Fig. 3, we project the sub segment of each estimated ego-path $\{\mathbf{P}_t \in SE(3)\}\, t = t_n, t_n+1, ..., t_m$ onto each image $\mathbf{I}_{t=n}$ where the position of the robot is visible by

$$\mathbf{u}_t = \mathbf{K}\mathbf{P}_n^{-1}\mathbf{P}_t \tag{16}$$

where the $\mathbf{u}_t$ are the 2D positions in image space of each robot pose. At the same time, we detect the drivable road region on the pixel level by semantic segmentation with the pre-trained MultiNet architecture [22]. Then, the left and right edges of the road were fit with second-order polynomial curves $\mathbf{f}_l$ and $\mathbf{f}_r$, such that for each $\mathbf{f}$, $\mathbf{f}(\mathbf{u}) = 0$ for points on the curve, $\mathbf{f}(\mathbf{u}) < 0$ for points to the left of the curve, and $\mathbf{f}(\mathbf{u}) > 0$ for points to the right of the curve. This is depicted in Fig. 3.

We assume that in the usual case, the vehicle will stay on the road. This method can correct the drift when the estimated ego-motion drift make the robot off-road. The problem is formulated as a pose graph optimization, as shown in Fig. 4. The graph vertices represent the robot poses $\{\mathbf{P_t}\}\, t = t_n, ..., t_m$, and we define three kinds of edges: one unary edge and two kinds of binary edges. The unary edges encode the constraint that the projected path points should be on the road region:

$$\mathbf{e}_t = \mathscr{L}_{relu}(-\mathbf{f}_l(\mathbf{u}_t))) + \mathscr{L}_{relu}(\mathbf{f}_r(\mathbf{u}_t))) \tag{17}$$

Here we apply a ReLU nonlinearity, as the loss should be zero when the vehicle is on the road for the unary edge. When realization, the width of the robot should also be considered to decide whether it is off road.

The first binary edge prevents the optimized poses from drifting too far from the original poses:

$$\mathbf{e}_{t,t+1} = \|log(\mathbf{P}_{t+1}^{-1}\mathbf{P}_t\mathbf{T}_t^{t+1})\|_2 \tag{18}$$

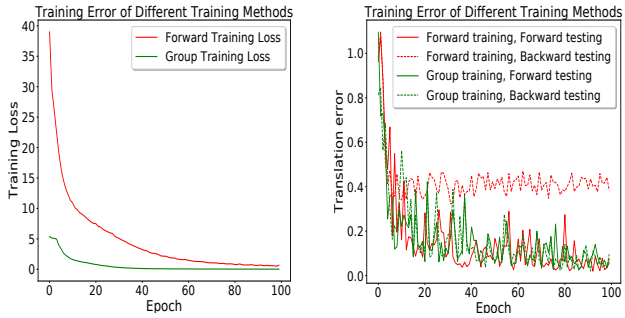
The other binary edge encodes the notion that the vehicle's pose should stay at approximately the similar distance from the road edges for consequent timestamps:

$$\mathbf{e}'_{t,t+1} = \|\mathbf{f}_l(\mathbf{u}_t) - \mathbf{f}_l(\mathbf{u}_{t+1})\|_2 + \|\mathbf{f}_r(\mathbf{u}_t) - \mathbf{f}_r(\mathbf{u}_{t+1})\|_2 \tag{19}$$

We can get the corrected ego-motion after optimization over poses by

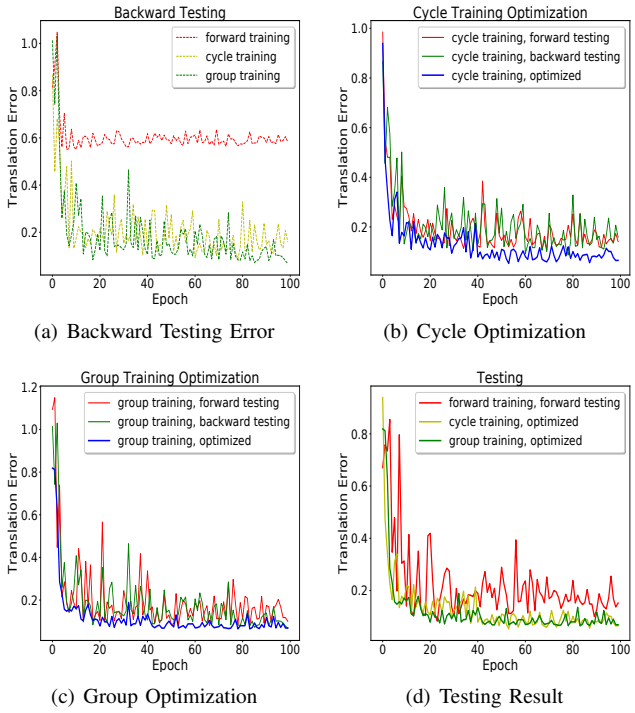
$$\{\mathbf{P}_t\}\, _{t=t_n,...,t_m} = \underset{\mathbf{P}_t}{\text{argmin}} \sum_{t=t_n}^{t_m} \mathbf{e}_t + \mathbf{e}_{t,t+1} + \mathbf{e}'_{t,t+1} \tag{20}$$

where the minimization is performed by g2o [23] with the Levenberg-Marquadt algorithm.



(a) Evaluated by Loss Function (b) Evaluated by KITTI Metric [24]

Fig. 5. Training loss of forward training and group traing are shown in Fig. 5(a), they both converge. Fig. 5(b) shows the evaluation result by KITTI evaluation metrics on training dataset. Group training means training with the proposed loss Eq. 12 and forward training represents training with *L2* loss Eq. 8. Backward testing means testing the trained model by reversed image pairs.



(a) Backward Testing Error (b) Cycle Optimization

(c) Group Optimization (d) Testing Result

Fig. 6. This figure shows the testing result of different training patterns. Fig. 6(a) shows the backward testing. Fig. 6(b) and Fig. 6(c) show the optimized ego-motion result by Eq. 15. The comparison of final testing is shown in Fig. 6(d).

IV. Experiments

Four experiments were conducted to show the performance of the proposed method. First, we verified whether CNNs are sufficient for learning ego-motion estimation, and whether the *L2* loss alone can learn the ego-motion mapping. This is an implicit assumption for most learning-based ego-motion estimation research. Second, we verified the efficiency of the proposed learning objective by evaluating on the testing dataset with different learned models. Third, the result is compared with other state-of-the-art methods. Finally, an experiment was performed to verify whether our drivable region-based method can further improve ego-motion estimation.

We evaluated the methods on the KITTI dataset [24], based on the metric of average rotation and translation error of each fixed distance segment [24]. The training data is split into multiple 3-frame sequences, as our model calculates the closure loss based on three consecutive images. The data is shuffled for training.

We implemented our method mostly in Python using the PyTorch deep learning framework [25]. The learning rate was set to 0.01 and the model was trained for 100 epochs. We did not tune and reduce the learning rate during training. The drivable region optimization is performed in C++ using g2o [23], via a pybind11-based Python wrapper. When testing on a laptop with Intel Core i7 and an NVIDIA GeForce GTX 1060, the method takes approximately 6.3 ms and 423 MB GPU memory to estimate the ego-motion of each frame. Road region-based correction takes about 70 ms for each optimization.

A. Assumption Verification

Most current CNN prediction methods only utilize the *L2* loss to learn the ego-motion mapping, ignoring other constraints described in Section III-A.3. So in our first experiment, we verify whether CNN with *L2* loss is enough to solve the problem; under the definition described in Sec III-A and whether neural networks are able to converge with the homomorphism loss.

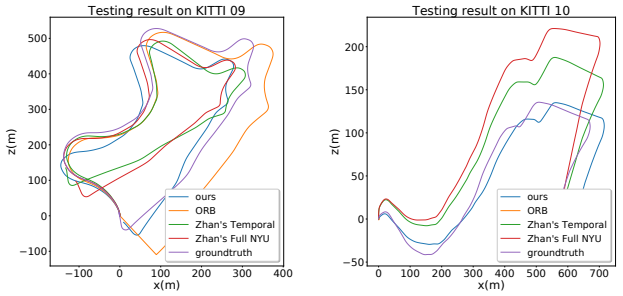
We first trained the network with the KITTI sequence 00-08 using the *L2* loss. As shown in Fig. 5(a), the training process converged. However, the backward testing error on the training dataset (shown in Fig. 5(b)) did not decrease. This means that the learned mapping with *L2* loss is not a correct ego-motion estimation mapping, as it does not maintain the homomorphism property and unable to extend to the inverse ego-motion when the input image pairs are reversed. That means that the neural networks are not able to know the mapping is homomorphism if we did not show networks the backward data or homomorphism based loss.

Second, we must show that the neural networks are able to converge with the homomorphism loss, because the convergence is a necessary condition to use learning-based method for ego-motion estimation. We train the same network on the same dataset with the proposed training loss function (Eq. 12). As shown in Fig. 5, the training process converged. This shows that the CNNs can fit the homomorphism mapping from image pair to ego-motion.

As a conclusion, CNNs can fit the homomorphism ego-motion mapping; we now evaluate how the homomorphism loss can improve performance.

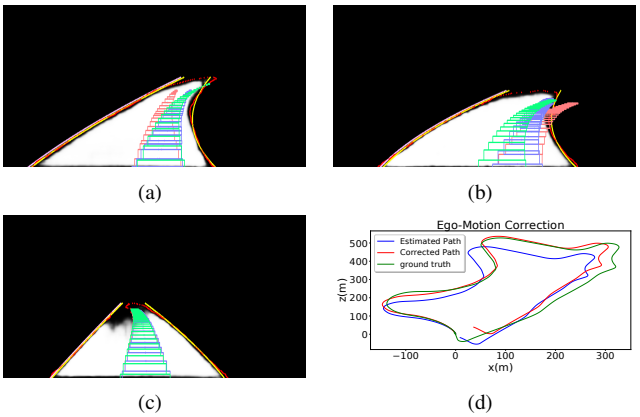
B. Loss function Efficiency

To show the influence of the proposed loss on the performance of learning-based visual odometry, we trained the same neural networks discussed in Sec. III-A.3 and used the same training and test split (KITTI sequences 00-08 for training, sequence 09 for testing). We compared the testing



(a) 09 (b) 10

Fig. 7. We compare our path on KITTI 09 and 10 with the ground truth and other methods. It shows that our path is much closer to the ground truth compared to the baselines.



(a) (b) (c) (d)

Fig. 8. Pose correction results. The green boxes are the ground truth positions, the red boxes are the estimated positions, and the blue boxes are the positions corrected by graph optimization. Fig. 8(a) is the correction when the estimated pose too far away from the road edge. Fig. 8(b) is the correction when the estimated poses are outside of the road. There is almost no correction on Fig. 8(c) because the estimated poses are roughly correct. Fig. 8(d) is the corrected path. We note the ground truth is only used for visualization, not optimization.

performance of the models trained under three conditions: 1) *forward training*: only $L2$ loss (Eq. 8), 2) *cycle training*: $L2$ loss and inverse loss (Eq. 10), and 3) *group training*: $L2$ loss, inverse loss, identity loss (Eq. 9), and closure loss (Eq. 11). We tested the trained models on KITTI 09, with pairs presented in both forward and backward order. The error curves are shown in Fig. 6. For each model, we average the testing error with the parameters from the last 30 training epochs. The average errors are shown in Table I.

We can see that the models trained under the forward condition only generalized well in forward testing (Fig. 6(d)) and did not generalize well to inverse testing (Fig. 6(a)). The models that used cycle or group training worked well on both forward and backward testing. We also find that the optimized testing results (by Eq. 15) of the cycle model (Fig. 6(b)) and group model (Fig. 6(c)) performed better than the unoptimized counterparts. Furthermore, the cycle-trained model and group-trained model achieved better forward-testing performance than the forward-trained model (Fig. 6(d)), which shows that the proposed training method can achieve better generalization for ego-motion estimation. Similarly, group training obtained better performance than cycle training, as shown in Table I.

C. Comparison with Other Methods

We compare the performance of our method with other learning-based ego-motion estimation methods. The model was trained on KITTI VO dataset sequence 00-08, and tested on sequence 09 and 10, the same as [7], [17] and [18]. Results are evaluated with the evaluation metric from [24]. Table II and Fig. 7 summarize the results. To make it clear, the results in Table II are the result without drivable region based correction. The model is trained with ground truth in absolute scale and tested with the dataset in similar distribution, so our results do not need scale alignment.

As there is no absolute scale information in the unsupervised methods (Zhou et al [7] and GeoNet [18]), their outputs are scaled to align with the ground truth. From Table II, we can see that our method outperforms their methods. While unsupervised, Zhan et al. [17] can get the absolute scale by using binocular images for training. The performance of our approach is better by about 40%. As we can see, while unsupervised methods offer the benefit of not requiring ground truth information, their performance is comparatively worse.

DeepVO [6] is the state-of-the-art supervised method, which combines CNNs and RNNs. To make the results comparable, we train our network with the same dataset as DeepVO [6] (KITTI 00, 02, 08, 09) and test on KITTI 10. We note that this is a smaller training dataset, which degrades our performance slightly compared to the setting in our previous experiment. Here, the translation and rotation error of our approach are 8.08% and 0.0123 (deg/m), which outperforms DeepVO, especially for rotation estimation. This shows the benefits of our homomorphism-based losses, even without using sequential information with an RNN.

Our performance is also better than LIBVISO2 [27] and competitive with ORB-SLAM [3], two popular geometry-based methods.

D. Drivable Region Based Correction

We show the performance of the correction method on KITTI 09 in Fig. 8. We see that when the estimated ego-motion has drift, and the drift moves the projected pose outside of the road region (Fig. 8(b)) or the path does not maintain a constant distance to the road edge (Fig. 8(a)), our method can correct the poses. When the estimated ego-motion is correct (Fig. 8(c)), the correction method does not change the poses. As shown in Fig. 8(d), the correction method improves the performance of ego-motion estimation. The translation error is reduced from 8.04% to 6.92%.

V. Conclusion

In this paper, we redefined the learning-based ego-motion estimation problem and proposed a novel objective for training the ego-motion estimation network. Our proposed inverse loss, identity loss, and closure loss are used in the training stage to put the functional mapping under the constraint

TABLE II

COMPARISON OF TRANSLATION AND ROTATION ERRORS FOR OUR METHOD VERSUS OTHER VISUAL ODOMETRY METHODS ON THE KITTI BENCHMARK.

Seq	Zhan et al. (from [17])		DeepVO (from [6])		Zhou et al. (from [7])		GeoNet (from [18])		LIBVISO2 (from [26])		ORBSLAM (from [3])		Our Method	
	Trans (%)	Rot (deg/m)	Trans (%)	Rot (deg/m)	Trans (%)	Rot (deg/m)	Trans (%)	Rot (deg/m)	Trans (%)	Rot (deg/m)	Trans (%)	Rot (deg/m)	Trans (%)	Rot (deg/m)
09	11.92	0.0360	-	-	17.84	0.0678	26.93	0.0954	4.04	0.0143	15.30	0.0026	8.04	0.0151
10	12.62	0.0343	8.11	0.0883	37.91	0.1778	24.69	0.0843	25.20	0.0388	3.68	0.0048	6.23	0.0097
Avg	12.27	0.0351	8.11	0.0883	28.88	0.1228	25.81	0.0899	14.62	0.0266	9.49	0.0037	**7.14**	0.0124

of group homomorphism. Experiments on KITTI showed that the novel objective not only made the testing result more accurate when the input is reversed, but also achieved better performance overall conditions. Moreover, our training method is also able to provide a confidence index for the estimated ego-motion, which is essential for future fusion and other tasks. We also explore correction of the estimated ego-motion with optimization based on the intuition that the estimated poses should be consistent with semantically-segmented road observations, and our experiments show that the correction can improve the estimated ego-motion.

In future work, our homomorphism-based losses could be combined with more expressive CNN architectures incorporating sequential information, such as RNNs, or in unsupervised scenarios. In addition, we would like to adapt our ego-motion drift correction method to use other inputs than road segmentation, which is only applicable in urban driving; attractive options include trajectory prediction and path planning methods.

REFERENCES

[1] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 1. Ieee, 2004.

[2] F. Fraundorfer and D. Scaramuzza, "Visual odometry: Part i: The first 30 years and fundamentals," *IEEE Robotics and Automation Magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[3] R. Mur-Artal, J. Montiel, and J. D. Tardós, "Orb-slam: a versatile and accurate monocular slam system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[4] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Mar. 2018.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[6] S. Wang, R. Clark, H. Wen, and N. Trigoni, "Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 2043–2050.

[7] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *CVPR*, vol. 2, no. 6, 2017, p. 7.

[8] R. Roberts, H. Nguyen, N. Krishnamurthi, and T. Balch, "Memory-based learning for visual odometry," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*. IEEE, 2008, pp. 47–52.

[9] V. Guizilini and F. Ramos, "Semi-parametric models for visual odometry," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 3482–3489.

[10] G. Costante, M. Mancini, P. Valigi, and T. A. Ciarfuglia, "Exploring representation learning with cnns for frame-to-frame ego-motion estimation," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 18–25, 2016.

[11] S. Pillai and J. J. Leonard, "Towards visual ego-motion learning in robots," *arXiv preprint arXiv:1705.10279*, 2017.

[12] G. Costante and T. A. Ciarfuglia, "Ls-vo: Learning dense optical subspace for robust visual odometry estimation," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1735–1742, 2018.

[13] S. Wang, R. Clark, H. Wen, and N. Trigoni, "End-to-end, sequence-to-sequence probabilistic visual odometry through deep neural networks," *The International Journal of Robotics Research*, vol. 37, no. 4-5, pp. 513–542, 2018.

[14] G. Iyer, J. Krishna Murthy, G. Gupta, M. Krishna, and L. Paull, "Geometric consistency for self-supervised end-to-end visual odometry," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 267–275.

[15] R. Mahjourian, M. Wicke, and A. Angelova, "Unsupervised learning of depth and ego-motion from monocular video using 3D geometric constraints," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5667–5675.

[16] R. Li, S. Wang, Z. Long, and D. Gu, "Undeepvo: Monocular visual odometry through unsupervised deep learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 7286–7291.

[17] H. Zhan, R. Garg, C. S. Weerasekera, K. Li, H. Agarwal, and I. Reid, "Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 340–349.

[18] Z. Yin and J. Shi, "Geonet: Unsupervised learning of dense depth, optical flow and camera pose," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 2018.

[19] E. Parisotto, D. Singh Chaplot, J. Zhang, and R. Salakhutdinov, "Global pose estimation with an attention-based recurrent network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 237–246.

[20] V. Peretroukhin and J. Kelly, "Dpc-net: Deep pose correction for visual localization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2424–2431, 2018.

[21] A. L. Onishchik, E. Vinberg, and V. Minachin, *Lie groups and Lie algebras*. Springer, 1993.

[22] M. Teichmann, M. Weber, M. Zoellner, R. Cipolla, and R. Urtasun, "Multinet: Real-time joint semantic reasoning for autonomous driving," *arXiv preprint arXiv:1612.07695*, 2016.

[23] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g 2 o: A general framework for graph optimization," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 3607–3613.

[24] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, Conference Proceedings, pp. 3354–3361.

[25] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS-W*, 2017.

[26] S. Song, M. Chandraker, and C. Guest, "High accuracy monocular SFM and scale correction for autonomous driving," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, pp. 1–1, 2015.

[27] A. Geiger, J. Ziegler, and C. Stiller, "Stereoscan: Dense 3d reconstruction in real-time," in *Intelligent Vehicles Symposium (IV)*, 2011.