Drift Reduction for Monocular Visual Odometry of Intelligent Vehicles Using Feedforward Neural Networks

Wagih, Hassan; Osman, Mostafa; Awad, Mohammed I. ; Hammad, Sherif

DOI
10.1109/ITSC55140.2022.9921796

Publication date
2022

Document Version
Final published version

Published in
Proceedings of the IEEE 25th International Conference on Intelligent Transportation Systems (ITSC 2022)

2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)
October 8-12, 2022, Macau, China

Drift Reduction for Monocular Visual Odometry of Intelligent Vehicles using Feedforward Neural Networks

Hassan Wagih[1*], Mostafa Osman[2*], Mohammed I. Awad[1], and Sherif Hammad[1]

Abstract— **In this paper, an approach for reducing the drift in monocular visual odometry algorithms is proposed based on a feedforward neural network. A visual odometry algorithm computes the incremental motion of the vehicle between the successive camera frames, then integrates these increments to determine the pose of the vehicle. The proposed neural network reduces the errors in the pose estimation of the vehicle which results from the inaccuracies in features detection and matching, camera intrinsic parameters, and so on. These inaccuracies are propagated to the motion estimation of the vehicle causing larger amounts of estimation errors. The drift reducing neural network identifies such errors based on the motion of features in the successive camera frames leading to more accurate incremental motion estimates. The proposed drift reducing neural network is trained and validated using the KITTI dataset and the results show the efficacy of the proposed approach in reducing the errors in the incremental orientation estimation, thus reducing the overall error in the pose estimation.**

I. Introduction

The interest in developing self-driving vehicles capable of navigating their environments without human intervention has significantly grown during the last decade. One of the main modules of a self-driving vehicle is the localization module [1]. A self-driving vehicle has to be able to estimate its position and orientation accurately in order to navigate its environment successfully. To achieve accurate localization, several exteroceptive sensors are usually used such as Global Positioning Systems (GPS) [2], Cameras [3], Light Detection And Ranging (LiDAR) [4], and so on.

Cameras can be utilized for localization using Visual Odometry (VO) algorithms [5], [6]. A VO algorithm estimates the incremental motion between successive frames captured by the camera. The position and orientation of the vehicle are then estimated by integrating these motion increments at each timestep. VO algorithms can be classified based on the type of camera into monocular [7], stereo [8], and RGB-D [9] visual odometries [10]. It can also be classified based on the motion estimation approach as feature-based and direct VO. Feature-based VO relies on detecting and matching visual features in the successive frames. Such features are then used to solve the motion estimation problem.

A monocular visual odometry algorithm uses the captured frames from only one camera to estimate the motion of the vehicle. Such technique uses Structure from Motion (SFM) to estimate the incremental motion [11], [12]. However, since the algorithm uses only one camera, the incremental motion of the vehicle can only be estimated up to an unobservable scale which can be determined using another sensor such as a wheel encoder or an Inertial Measurement Unit (IMU).

Since VO algorithms rely on integration in estimating the position and orientation, individual errors in each incremental motion lead to the accumulation of error in every time step. Although the drift error in VO algorithms cannot be eliminated, several works have tried to reduce such drift. In [13], a new descriptor called the Syntheitc BAsis (SYBA) descriptor is used to reduce the falsely matched features between the successive frames. In order to achieve this, a sliding window approach is developed where the features detected in a given frame is not only matched to the prior frame but to a window of previously captured frames. In [14], the drift in the estimation of a VO algorithm is reduced using a Bayesian Convolutional Neural Network that infers the direction of the sun. The sun direction then provides a global estimate of the orientation estimate which consequently reduces the drift in the visual odometry output.

Another approach to reduce the drift in the estimation of VO is through using sensor fusion techniques where the estimate from the VO algorithm is fused with other estimates from other sensors to reduce the error. Sensor fusion relies on probabilistic estimation approaches such as Extended Kalman Filter [15], Moving Horizon Estimation [16] and pose graph optimization [17], and so on. In such approaches, the uncertainty in the position and orientation estimates need to be quantified using the covariance matrix of the measurements in order to enhance the accuracy of the estimation. In [18], [19], a covariance estimation approach was developed utilizing a sensor not suffering from drift to estimate the covariance of the drift suffering odometry.

In this paper, a machine learning approach is proposed for reducing the drift in the incremental orientation estimates of a monocular VO algorithm which in turn results in the overall enhancement of the integrated position and orientation accuracy. By developing a neural network that can correlate the errors in the motion increments estimated by the VO algorithm with the motion of the features in the successive frames, the error in such increments can be reduced significantly. By using an accurate Realtime Kinematic GPS (RTK-GPS) during the training phase, the neural network can be trained to estimate the error in the visual odometry increments which can be then used to reduce the motion error in realtime during the operation of the

[1] Mechatronics Department, Ain Shams University, Cairo 11566, Egypt {Hassan.Wageh, Mohammed.awad}@eng.asu.edu.eg, Sherif.hammad@garraio.com
[2] Mechanical, Materials, and Maritime Engineering, Delft University of Technology, Delft, The Netherlands M.E.A.Osman@tudelft.nl
[*] The first two authors contributed equally to this work.

2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC) | 978-1-6654-6880-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/ITSC55140.2022.9921796

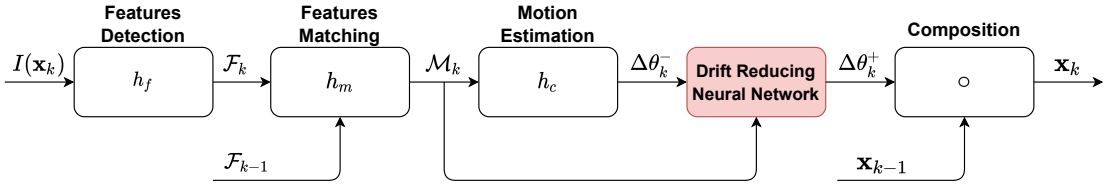

Fig. 1. The overall feature-based visual odometry pipeline with the proposed drift reducing neural network. The image I at the position and orientation $\mathbf{x}_k$ is passed to a features detection algorithm h_f followed by a matching algorithm h_m which matches the detected features to the features from the previous frame. The matched features are then used to estimate the motion increment using a motion estimation algorithm h_c. The statistical moments of the features motion in the camera frames are then passed to the NN along with the motion increment from the VO algorithm. Finally, the NN outputs a more accurate motion increment which is then used to calculate the current position and orientation of the vehicle through composition

vehicle. The proposed approach is validated using sequences from KITTI dataset with its ground-truth data [20].

The remainder of the paper is organized as follows, Section II contains the mathematical formulation of the drift error in the visual odometry followed by our proposed approach for reducing the drift in Section III. In Section IV the experimental work executed to evaluate the proposed method is explained and the results are shown in Section V. Finally, the conclusion and future work are stated in Section VI.

II. Error Modeling in Visual Odometry

As mentioned earlier, a visual odometry estimate the position and orientation of a vehicle by integrating the incremental motion. For feature-based VO algorithm, the first step is detecting the features in the frame, this can be modeled as

$$\mathcal{F}_k = \underbrace{h_{\mathrm{f}}(\mathbf{x}_k)}_{\hat{\mathcal{F}}_k} + \mathbf{w}_k, \tag{1}$$

where $\mathbf{x}_k = [\mathbf{p}_k^\top, \boldsymbol{\theta}_k^\top] \in \mathbb{R}^3 \times \mathbb{M}$ is the state vector of the vehicle in the k-th time-step, which is composed of the position $\mathbf{p} \in \mathbb{R}^3$ and the orientation $\boldsymbol{\theta} \in \mathbb{M}$ of the vehicle using an orientation representation (e.g. quaternions or Euler angles). $\mathcal{F}_k := \{f_k^i\}_{i=1}^{N_f} \in \mathbb{I}$ is the detected features set and $f := [u, v]^\top$ denotes a detected feature in the image. $h_{\mathrm{f}} : \mathbb{R}^3 \times \mathbb{M} \to \mathbb{I}$ is the model of the feature detection, and finally $\mathbf{w}_k$ is a noise term that models the errors in the feature detection process due to the noise affecting the camera.

After the detection of features in the k-th frame, they are matched with the features detected in the $(k-1)$-th frame. Using (1) and applying Taylor expansion, this step can be modeled as

$$\begin{aligned}\mathcal{M}_k &= h_{\mathrm{m}}(\mathcal{F}_{k-1}, \mathcal{F}_k) + \varepsilon_k^{\mathrm{m}} \\ &\approx \underbrace{h_{\mathrm{m}}(\hat{\mathcal{F}}_{k-1}, \hat{\mathcal{F}}_k)}_{\hat{\mathcal{M}}_k} + \underbrace{\frac{\partial h_{\mathrm{m}}}{\partial \hat{\mathcal{F}}_{k-1}} \mathbf{w}_{k-1} + \frac{\partial h_{\mathrm{m}}}{\partial \hat{\mathcal{F}}_k} \mathbf{w}_k}_{\mathbf{w}_k^{\mathrm{m}}} + \varepsilon_k^{\mathrm{m}},\end{aligned} \tag{2}$$

where $\mathcal{M}_k := \{(f_{k-1}^i, f_k^i)\}_{i=1}^{N_f} \in \mathbb{I}^2$ is the set of matched features (here we ignore all unmatched features), $h_{\mathrm{m}} : \mathbb{I} \times \mathbb{I} \to \mathbb{I}^2$ is a mapping that represents the features matching algorithm, $\mathbf{w}_k^{\mathrm{m}}$ is the effect of the white noise $\mathbf{w}_k$ on the matching, and $\varepsilon_k^{\mathrm{m}} := \varepsilon^{\mathrm{m}}(\mathcal{F}_{k-1}, \mathcal{F}_k)$ is another noise term that is correlated to errors in the detection and matching of features in the successive frames. Such errors can result from false feature matching (wrong association), errors in the calibration of the intrinsic parameters of the camera, ... etc [21], [22]. An example of matching errors in the VO pipeline is shown in Fig. 2.

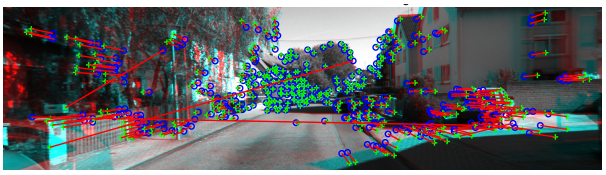

Fig. 2. An example of errors in mismatching of features between the 2180-th frame and the 2181-st frame in KITTI data set sequence 5. The two frames are shown in the figure using red and blue shades. The features in the first frame is represented by the green (+) marks and blue circles in the second frame. Finally, the matched features are joined by red lines.

Finally, depending on the type of camera being used, a motion estimation algorithm can be used to compute the motion increment from the matched feature pairs $\mathcal{M}_k$. This can be modeled similar to (2) as

$$\begin{aligned}\Delta\mathbf{x}_k &= h_{\mathrm{c}}(\hat{\mathcal{M}}_k) + \frac{\partial h_{\mathrm{c}}}{\partial \hat{\mathcal{M}}_k}(\mathbf{w}_k^{\mathrm{m}} + \varepsilon_k^{\mathrm{m}}) \\ &= h_{\mathrm{c}}(\hat{\mathcal{M}}_k) + \mathbf{w}_k^{\mathrm{c}} + \varepsilon_k^{\mathrm{c}},\end{aligned} \tag{3}$$

where $h_{\mathrm{c}} : \mathbb{I}^2 \to \mathbb{R}^3 \times \mathbb{M}$ is the motion estimation function, $\mathbf{w}_k^{\mathrm{c}}$ is the colored noise resulting from the origin white noise in (1), and $\varepsilon_k^{\mathrm{c}} := \varepsilon^{\mathrm{c}}(\mathcal{F}_{k-1}, \mathcal{F}_k)$ is the effect of the noise term $\varepsilon_k^{\mathrm{m}}$ on the motion estimation module.

We aim at reducing the effect of the noise term $\varepsilon_k^{\mathrm{c}}$ in order to increase the accuracy of the motion increments and ultimately reduce the drift in the VO algorithm. Using a Bayesian approach, this problem can be stated as the problem of estimating the distribution

$$p(\Delta\mathbf{x}_k \mid \varepsilon_k^{\mathrm{c}}, \mathcal{M}_k) = \eta\, p(\varepsilon_k^{\mathrm{c}} \mid \Delta\mathbf{x}_k, \mathcal{M}_k) p(\Delta\mathbf{x}_k \mid \mathcal{M}_k). \tag{4}$$

In (4), the prior distribution $p(\Delta\mathbf{x}_k \mid \mathcal{M}_k)$ is the visual odometry algorithm. The distribution $p(\varepsilon_k^{\mathrm{c}} \mid \Delta\mathbf{x}_k, \mathcal{M}_k)$ is the likelihood distribution of the $\varepsilon_k^{\mathrm{c}}$. This distribution is hard to determine in practice because (i) it is affected by many factors which are hard to model, (ii) the detection and matching of features as well as the motion estimation are actually algorithms that has no closed form expressions which makes the propagation of these errors to the motion increments infeasible.

Consequently, we propose the use of a feedforward Neural Network (NN) to estimate the error $\varepsilon_k^{\mathrm{c}}$ using the output of a monocular VO algorithm $\Delta\mathbf{x}_k$ as well as the matched features pairs $\mathcal{M}_k$.

III. Drift Reduction using Neural Networks

In this section, the NN used to estimate the noise component $\varepsilon^{\mathrm{c}_k}$ is described. A monocular VO algorithm estimates the incremental motion of the vehicle up to a scale which can be computed using an external sensor such as a wheel encoder or an IMU. the proposed work is only concerned with estimating the error component for the orientation increment $\Delta\theta$. The overall pipeline of the VO algorithm with the proposed Drift Reducing Neural Network (DRNN) is shown in Fig. 1.

The objective of the proposed DRNN is to estimate a refined incremental orientation $\Delta\theta$ using the output of the monocular VO as well as information about the matched features $\mathcal{M}_k$. As disucssed in the previous section, the error in the matched features result in errors in the estimated motion increments by the VO. Such errors can be detected using the motion of the features in a given frame $I(\mathbf{x}_k)$ with respect to $I(\mathbf{x}_{k-1})$

$$\underbrace{\begin{bmatrix}\Delta u_k^i \\ \Delta v_k^i\end{bmatrix}}_{\Delta f_k^i} = f_k^i - f_{k-1}^i, \tag{5}$$

where Δu and Δv are the change in the position of the feature in the successive frames. Therefore, the DRNN can be trained using the position change of the features.

The space of the matched features in the successive frames is large and many different types of error can occur consequently, training the DRNN using all the matched features would require a very large amount of training samples which would be infeasible during the training phase of the DRNN. Alternatively, we use some of the statistical moments of the set $\{\Delta f_k^i\}_{i=1}^{N_f}$ to train the DRNN to reduce the error. In this paper, the information about the change in the positions is captured using three statistical moments (mean μ_k, variance ν_k, skewness κ_k) in addition to the Root Mean Square (RMS) ρ_k.

As for the orientation increment, the orientation representation used is the unit quaternions. It is well-known that the rotation space $\mathbb{M}$ is not an Euclidean space. However, the tangent space of the rotation space is an Euclidean space ($\mathbb{R}^3$). Consequently, to avoid discontinuities in the cost function of the DRNN, the orientation deviation in the tangent space $\xi := \exp_{\mathrm{q}}(\Delta\theta) \in \mathbb{R}^3$ is used where $\exp_{\mathrm{q}} : \mathbb{M} \to \mathbb{R}^3$ is the exponential map that maps from the quaternion space to the tangent space.

Finally, the input layer of the DRNN is defined as $[(\xi_k^-)^\top, \mu_k, \nu_k, \kappa_k, \rho_k]^\top$, where $\xi_k^- := \exp_{\mathrm{q}}(\Delta\theta_k^-)$ is the orientation deviation of the orientation increment estimated by the VO algorithm. The output layer of the DRNN is the refined orientation increment ξ_k^+. The overall position and orientation of the vehicle can finally be computed as

$$\theta_k^+ = \theta_{k-1}^+ \odot \Delta\theta_k^+ \tag{6}$$

$$\mathbf{p}_k = \mathbf{p}_{k-1} + R(\theta_k^+)\Delta\mathbf{p}_k \tag{7}$$

where $\odot$ denotes the quaternion product, and $R : \mathbb{M} \to SO(3)$ is a mapping from quaternion group to the special orthogonal group. Furthermore, to make sure that the DRNN does not overfit to the training data and is actually capable of reducing the VO errors during operation, a Bayesian regularization approach is used [23].

IV. Experimental Work

In this section we discuss the details of the implemented monocular VO odometry algorithm and the executed experimental work in order to validate our approach. Using Mathworks' Matlab, a feature-based monocular VO algorithm is developed using Speeded Up Robust Features (SURF) detector [24], [25]. The features are then matched using the approach introduced in [26], where only unique features are matched by using forward-backward matching.

The monocular VO implementation estimates the motion $\Delta\mathbf{x}$ using the epipolar constraint between frames

$$\begin{bmatrix}x_{k-1}^i & y_{k-1}^i\end{bmatrix}\mathbf{E}\begin{bmatrix}x_k^i \\ y_k^i\end{bmatrix} = 0, \quad \forall\ 1 \le i \le N_f, \tag{8}$$

where (x^i, y^i) is the position of the i-th feature in the camera frame, and $\mathbf{E} \in \mathbb{R}^{3\times3}$ is the essential matrix for the calibrated camera [12]. The essential matrix $\mathbf{E}$ can be estimated using the five-point algorithm [27]. Furthermore, an outliers rejection approach was applied based on the M-estimator sample consensus (MSAC) algorithm [28]. The essential matrix $\mathbf{E}$ can then be decomposed to the translation (up to a scale) and rotation of the camera between two successive frames [12].

Since the monocular VO algorithm estimates the motion up to a scale, here we compute this scale using the ground-truth data for both the VO with and without the DRNN as we are only concerned with reducing the orientation drift in the VO output. This does not affect the validation of the proposed approach since the scale is computed using the same method for both outputs.

The outcome of the VO is passed to a feed-forward NN with 3 layers. The input layer consists of 11 neurons for the inputs discussed in the previous section. The hidden and output layers consist of 30 and 3 neurons respectively. The activation function used is a sigmoid function and the weights of the NN are optimized using the Levenberg-Marquardt optimization algorithm [29] and the training is done using 1500 training epochs.

Using the KITTI data set, the NN is trained with the first 60% percent of sequences 0, 2, 5, 6, 7, 8, and 9 and is tested on the remaining 40% percent of each sequence. Sequence 4 is used completely for training as it is too short for partitioning. Finally, sequence 10 is completely used for testing the DRNN. The training is accomplished by using the ground-truth data as targets for the NN.

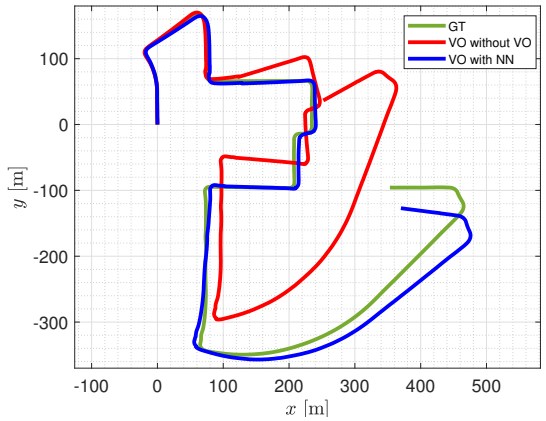

Fig. 3. The computed trajectory by the VO with and without the DRNN along with the ground-truth trajectory of the testing portion of sequence 0 of KITTI dataset.

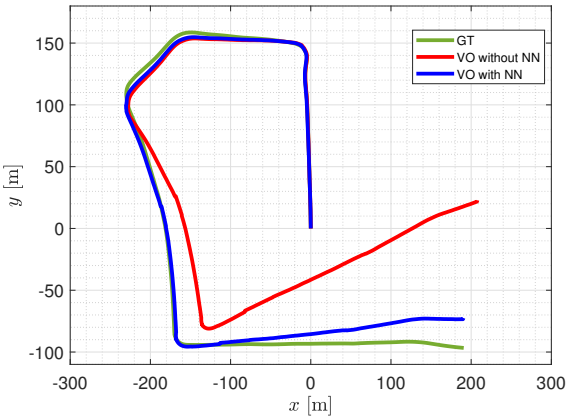

Fig. 5. The computed trajectory by the VO with and without the DRNN along with the ground-truth trajectory of the testing portion of sequence 5 of KITTI dataset.

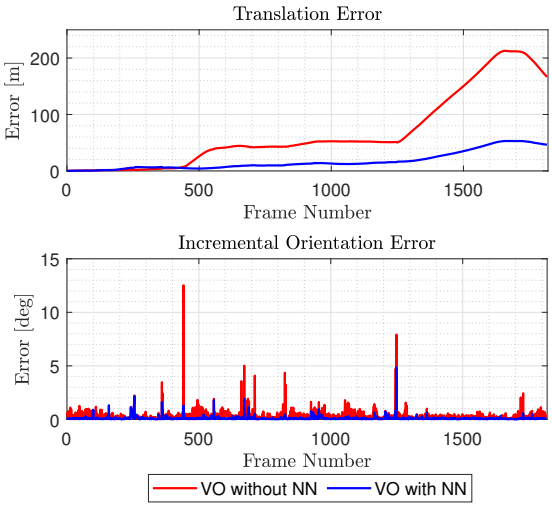

Fig. 4. The translation error in the position estimation of the vehicle in sequence 0 (top) and the incremental orientation error at every frame (bottom).

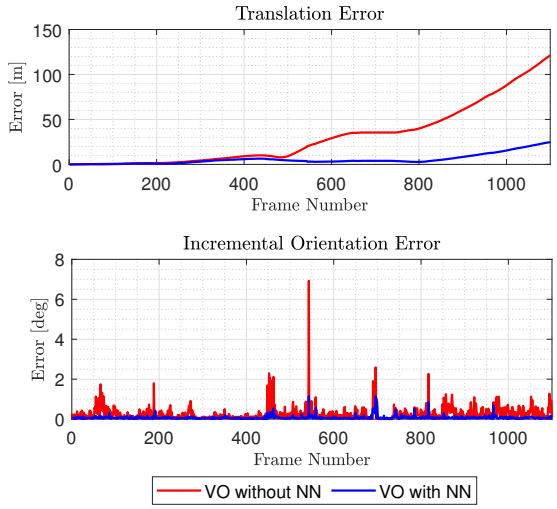

Fig. 6. The translation error in the position estimation of the vehicle in sequence 5 (top) and the incremental orientation error at every frame (bottom).

The performance of the DRNN is validated and its output is compared to that of the monocular VO algorithm without the DRNN. For both outputs, the Root Mean Squared Error (RMSE) in translation and orientation are calculated using the ground-truth data provided for each sequence. The orientation error is computed as

$$R_k^{e,\pm} = R(\theta_k^{GT})^{-1} R(\theta_k^{\pm}) \tag{9}$$

where $R_k^{e,+}, R_k^{e,-} \in SO(3)$ represents the error rotation matrix for the VO with and without the proposed DRNN, θ_k^{GT} represents the ground truth orientation in as a quaternion.The implementation of this work is available in the github repository shown below [1]

V. Results and Discussion

In this section the results of the testing scenarios are discussed as mentioned in Section IV. Fig. 3 shows the estimated trajectory by the VO algorithm with and without the proposed DRNN for the testing part of sequence 0 as well as the ground-truth trajectory. As can be seen in the figure, the computed trajectory using the VO algorithm without the DRNN suffers from significantly more drift compared to the VO algorithm with the DRNN.

Fig. 4 shows the translation error as well as the incremental orientation error for sequence 0. Using the proposed DRNN, the estimation of the incremental orientation change was persistently more accurate over the sequence frames which led to a better overall pose estimation. Although the VO output was relatively accurate at the beginning of the path however, with every turn that the vehicle took, the amount of drift in the estimated orientation increased which led to a root mean translation error of 94.19 m compared to only 23.35 m when using the DRNN which amounts to a 75% improvement. This can be attributed to the enhanced orientation estimation which was enhanced from a RMSE of 15.22 degrees to only 5.03 degrees (66%) when using the

[1] https://github.com/hassanwagih/DriftReducingNeuralNetwork

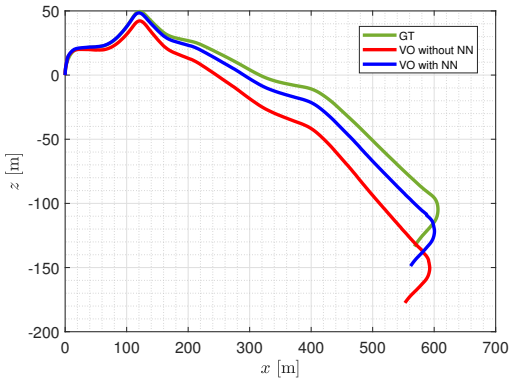

Fig. 7. The computed trajectory by the VO with and without the DRNN along with the ground-truth trajectory of the testing portion of sequence 9 of KITTI dataset.

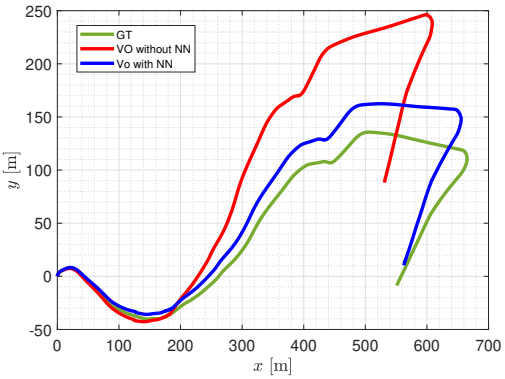

Fig. 9. The computed trajectory by the VO with and without the DRNN along with the ground-truth trajectory of the testing portion of sequence 10 of KITTI dataset.

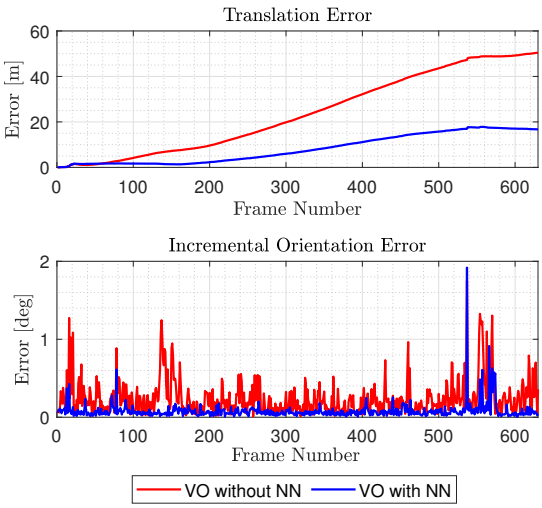

Fig. 8. The translation error in the position estimation of the vehicle in sequence 9 (top) and the incremental orientation error at every frame (bottom).

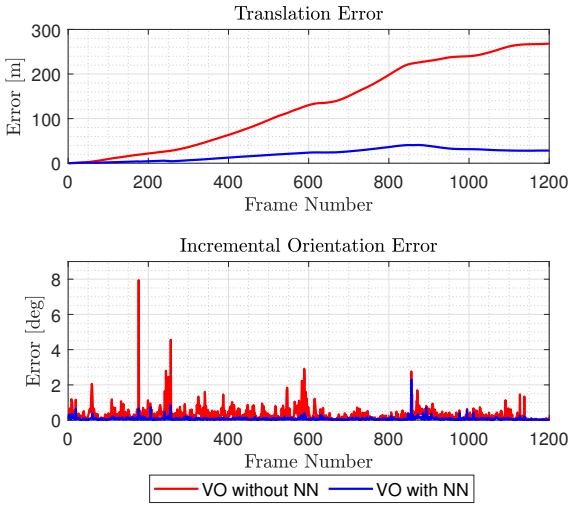

Fig. 10. The translation error in the position estimation of the vehicle in sequence 10 (top) and the incremental orientation error at every frame (bottom).

DRNN.

Fig. 5 shows the results for the testing part of sequence 5. This sequence did not contain as many turns as sequence 0, which resulted in a slower drift compared to sequence 0 and a less error in the overall sequence (shown in Fig. 6). That being said, by the end of the trajectory the VO output still suffered from a larger amount of drift compared to the VO output with the DRNN. In the case of using the DRNN, the RMSE in orientation for sequence 5 was reduced from 12.44 degrees to 3.2 degrees (74% improvement) which resulted in 82% enhancement in the position accuracy. Here, the DRNN output was also persistently better than that of the VO output. Furthermore, notice that the DRNN managed to reduce the incremental orientation error at the error spikes. These spikes can lead to a significant increase in the overall error in the path due to the reliance on integration.

Similarily, Fig. 7 and 8 show the results for sequence 9 testing part of KITTI dataset which are similar to those of sequence 5. In this case, the orientation RMSE was reduced from 7.11 degress to 2.08 degrees (70% improvement) and the translation RMSE was reduced from 29.66 m to 10.34 m (65% improvement).

Fig. 9 and 10 show the estimation results of the complete trajectory of sequence 10 as well as the translation and incremental orientation errors. Even though the sequence was completely used for testing without partitioning, the use of the DRNN resulted in significant enhancement in the translation and orientation estimation accuracy. In addition to the previous tests using the partitioned sequences, this sequences show the practical feasibility of the proposed DRNN. The idea behind the DRNN is that the drift can be reduced through training the NN during the development phase of the vehicle and then using the DRNN during operation to reduce the drift in the VO estimate. For sequence 10, the orientation RMSE was reduced from 29.06 degrees to 7.02 degrees (76% improvement) which then led to a reduction in translation error from 160.56 m to 23.85 m (85% improvement).

Finally, Table I shows the overall results for the used test sequences. The use of the DRNN resulted in a significant

reduction in the translation and orientation errors for all of the testing sequences which shows that through using the proposed approach, the performance of the VO algorithm can be significantly enhanced.

TABLE I
RMSE VALUES FOR THE KITTI DATASET SEQUENCES

Seq. No.	Rotation [deg]		Translation [m]		Distance [m]
	NN	VO	NN	VO	
0	**5.03**	15.22	**23.35**	94.19	1648
2	**8.34**	23.73	**39.65**	93.46	2108
5	**3.2**	12.44	**8.05**	45.36	958
6	**3.79**	15.26	**6.96**	49.88	491
7	**1.83**	3.77	**4.08**	5.44	238
8	**6.88**	16.8	**35.86**	81.72	1348
9	**2.08**	7.11	**10.34**	29.66	706
10	**7.02**	29.06	**23.85**	160.56	919

VI. CONCLUSION AND FUTURE WORK

In this paper, a drift reduction approach for the orientation estimation in VO using a neural network is proposed. The proposed approach estimates the error which is correlated to the errors in the feature detection and matching algorithms and consequently is able to refine the incremental orientation estimation of the VO odometry. Through training the neural network during the development of the vehicle, the performance of the VO can be significantly enhanced using the proposed approach. our proposed DRNN was validated using KITTI dataset, where 7 sequences where partitioned to training and testing sequences for validation. Furthermore, a complete sequence was used for further validation of the algorithm. The results show the efficacy of the proposed DRNN in significantly reducing the drift in the VO output.

In the future work, the proposed approach can be extended to work with different types of cameras such as RGB-D or stereo cameras as well as different VO algorithms such as direct VO. Furthermore, the effect of the proposed approach can also be integrated with visual simultaneous localization and mapping algorithms to achieve more accurate results and avoid significant drift in the pose estimates.

REFERENCES

[1] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to autonomous mobile robots*. MIT press, 2011.

[2] N. M. Drawil, H. M. Amar, and O. A. Basir, "GPS localization accuracy classification: A context-based approach," *IEEE Trans. on Intelligent Transportation Systems*, vol. 14, no. 1, pp. 262–273, 2012.

[3] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," *The Int. Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, 2015.

[4] J. Zhang and S. Singh, "LOAM: Lidar odometry and mapping in real-time." in *Robotics: Science and Systems*, vol. 2, no. 9. Berkeley, CA, 2014, pp. 1–9.

[5] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *Robotics & Automation Magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[6] F. Fraundorfer and D. Scaramuzza, "Visual odometry: Part ii: Matching, robustness, optimization, and applications," *Robotics & Automation Magazine*, vol. 19, no. 2, pp. 78–90, 2012.

[7] M. Sabry, A. Al-Kaff, A. Hussein, and S. Abdennadher, "Ground vehicle monocular visual odometry," in *Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 3587–3592.

[8] H. Fan and S. Zhang, "Stereo odometry based on careful frame selection," in *Int. Symposium on Computational Intelligence and Design (ISCID)*, vol. 2. IEEE, 2017, pp. 177–180.

[9] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *IEEE/RSJ Int. Conf. on intelligent robots and systems*. IEEE, 2012, pp. 573–580.

[10] M. O. Aqel, M. H. Marhaban, M. I. Saripan, and N. B. Ismail, "Review of visual odometry: types, approaches, challenges, and applications," *SpringerPlus*, vol. 5, no. 1, pp. 1–26, 2016.

[11] M. Tomono, "3-d localization and mapping using a single camera based on structure-from-motion with automatic baseline selection," in *Proc. of the Int. Conf. on Robotics and Automation*. IEEE, 2005, pp. 3342–3347.

[12] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[13] A. Desai and D.-J. Lee, "Visual odometry drift reduction using syba descriptor and feature transformation," *IEEE Trans. on Intelligent Transportation Systems*, vol. 17, no. 7, pp. 1839–1851, 2016.

[14] V. Peretroukhin, L. Clement, and J. Kelly, "Reducing drift in visual odometry by inferring sun direction using a bayesian convolutional neural network," in *Int. Conf. on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 2035–2042.

[15] S. Thrun, "Probabilistic robotics," *Communications of the ACM*, vol. 45, no. 3, pp. 52–57, 2002.

[16] M. Osman, M. W. Mehrez, M. A. Daoud, A. Hussein, S. Jeon, and W. Melek, "A generic multi-sensor fusion scheme for localization of autonomous platforms using moving horizon estimation," *Trans. of the Institute of Measurement and Control*, vol. 43, no. 15, pp. 3413–3427, 2021.

[17] X. Tao, B. Zhu, S. Xuan, J. Zhao, H. Jiang, J. Du, and W. Deng, "A multi-sensor fusion positioning strategy for intelligent vehicles using global pose graph optimization," *Trans. on Vehicular Technology*, 2021.

[18] M. Osman, A. Hussein, A. Al-Kaff, F. García, and D. Cao, "A novel online approach for drift covariance estimation of odometries used in intelligent vehicle localization," *Sensors*, vol. 19, no. 23, p. 5178, 2019.

[19] M. Osman, A. Hussein, A. Al-Kaff, F. García, and J. M. Armingol, "Online adaptive covariance estimation approach for multiple odometry sensors fusion," in *Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 355–360.

[20] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The Int. Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

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

[22] R. Sagawa and Y. Yagi, "Accurate calibration of intrinsic camera parameters by observing parallel light pairs," in *Int. Conf. on Robotics and Automation*, 2008, pp. 1390–1397.

[23] F. D. Foresee and M. T. Hagan, "Gauss-newton approximation to bayesian learning," *Proc. of the Int. Conf. on Neural Networks (ICNN'97)*, vol. 3, pp. 1930–1935 vol.3, 1997.

[24] H. Benseddik, O. Djekoune, and M. Belhocine, "SIFT and SURF performance evaluation for mobile robot-monocular visual odometry," *Journal of Image and Graphics*, vol. 2, pp. 70–76, 01 2014.

[25] Y. Wang, J. Fan, C. Qian, and L. Guo, "Ego-motion estimation using sparse surf flow in monocular vision systems," *Int. J. of Advanced Robotic Systems*, vol. 13, no. 6, 2016.

[26] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *Proc. of the Int. Conf. on Computer Vision Theory and Applications*, vol. 1. SCITEPRESS, 2009, pp. 331–340.

[27] D. Nist, "An efficient solution to the five-point relative pose problem," *Trans. on Pattern Analysis and Machine Intelligence*, vol. 26, no. 06, pp. 756–777, jun 2004.

[28] P. H. Torr and A. Zisserman, "MLESAC: A new robust estimator with application to estimating image geometry," *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 138–156, 2000.

[29] M. Hagan and M. Menhaj, "Training feedforward networks with the marquardt algorithm," *Trans. on Neural Networks*, vol. 5, no. 6, pp. 989–993, 1994.