Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 1930881, 10 pages
https://doi.org/10.1155/2021/1930881

Research Article

Visual Odometry for Self-Driving with Multihypothesis and Network Prediction

Yonggang Chen, Yanghan Mei, and Songfeng Wan

Department of Mechanical and Electrical Engineering, Dongguan Polytechnic, Dongguan 523808, China

Correspondence should be addressed to Yonggang Chen; yonggang44@163.com

Received 8 July 2021; Revised 15 August 2021; Accepted 16 August 2021; Published 26 August 2021

Academic Editor: Yang Li

Robustness in visual odometry (VO) systems is critical, as it determines reliable performance in various scenarios and challenging environments. Especially with the development of data-driven technology, such as deep learning, the combination of data-driven VO and traditional model-based VO has achieved accurate tracking performance. However, the existence of local optimums in the model-based cost function still limits the robustness. In this study, we introduce a novel framework with a particle filter (PF) in the optimization process, where the PF is constructed by deep neural network (DNN) prediction. We propose constructing the PF by motion prediction classification and its uncertainty based on the characteristic of on-road driving motion. At the same time, an interval DNN prediction strategy is introduced to improve the real-time performance. Experimental results show that our framework obtains better tracking accuracy and robustness than the existing works, while the time consumption is maintained.

1. Introduction

As a fundamental problem of robot navigation, visual odometry (VO) has been studied for decades. Due to the mature studies of camera modeling and perception modeling [1], the model-based method has drawn lots of attention in the last twenty years. Some representative studies have been published [2, 3], and many mile-stone results in terms of tracking accuracy have been achieved.

With the development of deep learning in the last decades, especially the development of open image dataset [4] and graphics processing unit (GPU), being different from the model-based method, the data-driven approach is gaining popularity [5, 6]. Particularly, the data-driven end-to-end visual odometry methods [7, 8] are more straightforward and simpler than the model-based methods, while the accuracy and generalization are still inferior to the model-based methods. To improve the accuracy and the robustness of VO systems, the combinations of model-based methods and data-driven methods have been proposed [9]. Such combinations incorporate the prior knowledge information provided by a DNN into a model-based optimization framework [10].

Regarding the application in self-driving, VO plays an essential role in the navigation tasks for visual localization and mapping [11]. With the complicated and challenging situations of city traffic, the combinations of data-driven methods and model-based methods are the way to handle these challenges [12]. However, the robustness of the VO and visual simultaneous localization and mapping (VSLAM) systems in the various environments of self-driving applications is still an open problem.

To overcome the robustness problem in the application of self-driving, we propose to cooperate the DNN-based image classification network with a particle filter (PF) in a model-based pose estimation framework, as shown in Figure 1. With the generalization from the image classification network due to its high-level representation and the multiple hypotheses introduced by the DNN-enabled PF in pose optimization, the robustness and accuracy of visual pose estimation are improved. The

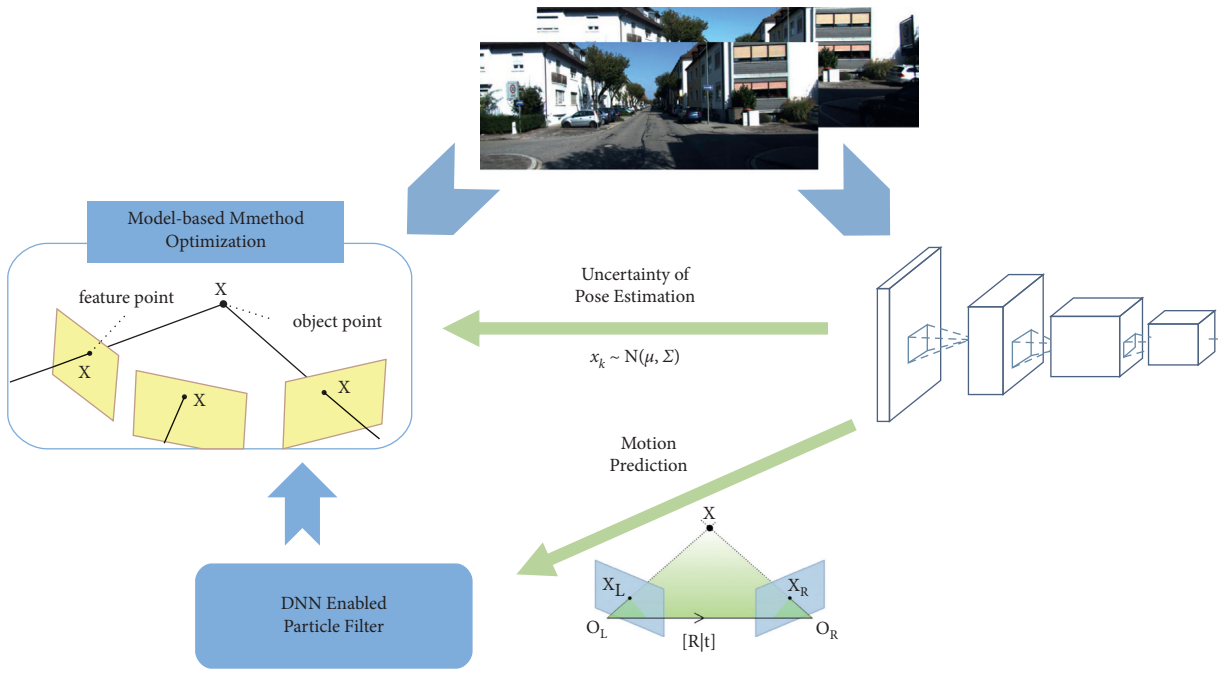


Figure 1: The framework of our method. With the prior knowledge from DNN prediction, we leverage the DNN motion classification for particle filter establishment. With the particle filter, the multihypothesis pose optimization can be performed for robust visual tracking.

simulation and experimental results demonstrate the superiority of our proposed method.

The contribution of our study is twofold:

(1) We propose a DNN-enabled PF for self-driving pose estimation, where the DNN-based image classification network is used for generalization

(2) A multihypothesis pose estimation of a model-based framework system is introduced to overcome the local optimum during optimization

2. Related Work and Motivation

Our proposed work intersects the topics of DNN application in VO and on-road self-driving. Applying DNN to VO and VSLAM systems (Figure 2) is popular in recent years due to the significant progress in image processing and pattern recognition by DNN. As for the end-to-end estimation of transformation in VO application, DeepVO [7] and UndeepVO [8] are the typical examples of supervised and unsupervised training methods, respectively, which leverage the DNN for pairwise image processing. Due to the multiple level perception in DNN, especially the convolution operation, the obtained information from various layers is rich, and the tracking robustness can be enhanced. However, the localization accuracy of those data-driven methods is still lower than the model-based VO and VSLAM systems, such as DSO [2], ORB-SLAM [13], and SVO [3].

To improve the accuracy of end-to-end data-driven VO pose estimation, researchers propose to combine DNN prediction and model-based pose optimization. In CNN-SVO [14], the output from DNN is utilized as the prior knowledge for optimization, which leads to accurate camera tracking performance due to the feasible DNN prediction. Similarly, CNN-SLAM [15] incorporates the depth prediction from DNN into a mature model-based VSLAM system [16], resulting in a dense structure and improved camera motion estimation. The study in [9] takes the prediction of pose transformation from a DNN as the initialization of localization directly for more robust tracking, while the advanced performance is only achieved in the environments similar to the training data. The study in [17] uses both the depth and pose DNN prediction for optimization initialization, where the depth, pose, and uncertainty are generated from the network simultaneously.

As for the VO and VSLAM system in the application of self-driving, thanks to the progress being made with deep learning, visual-based automated driving is advancing rapidly. There are two main paradigms in this area [11]:

(1) The mediated perception approach, which semantically reasons the scene and determines the driving decision based on it [18].

(2) The behavior reflex approach that learns the driving decision end-to-end [19]. The reliable and fast mapping and localization are needed in almost all driving scenarios. Due to the high resolution of cameras compared with other sensors like RADAR and LiDAR, situations that require detailed knowledge about the environment are dedicated to the applications of VSLAM and VO [20]. Therefore, VO and VSLAM systems play an essential role in self-driving. Especially with the development of deep learning, applying data-driving methods to self-driving has achieved much progress in recent years [21, 22]. However, to work in challenging situations,

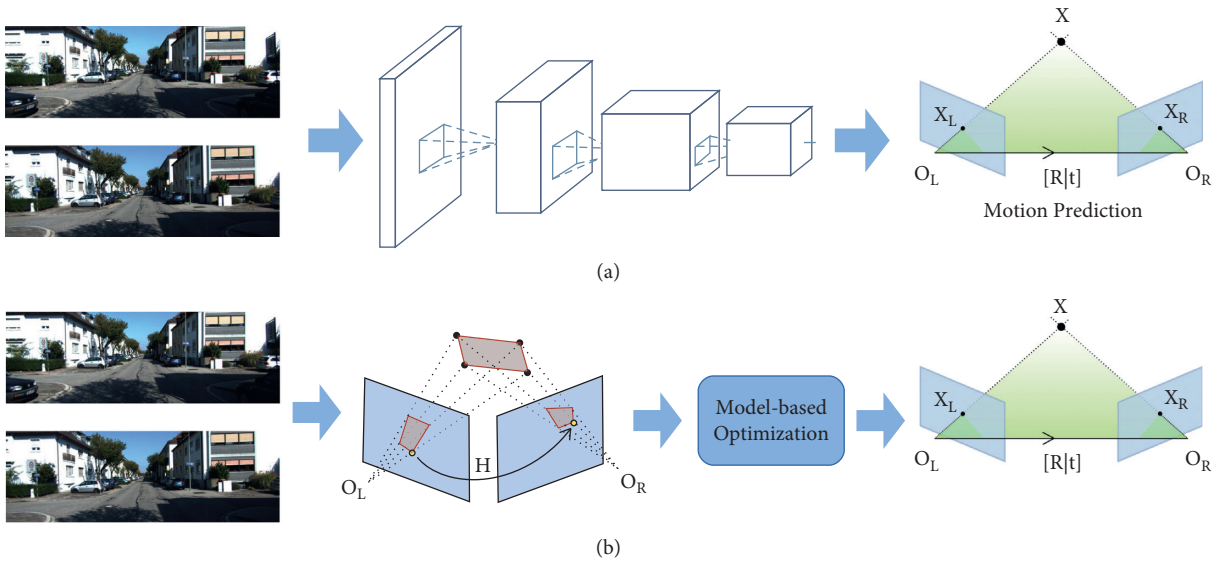


Figure 2: Pipeline of the model-based method and data-driven method. Both methods have the same input and output. The model-based method involves solving transformation using feature detection, data association, and optimization, whereas the data-driven method uses a DNN to predict the motion directly. (a) Model-based visual tracking. (b) Data-driven visual tracking.

including motion blurry, large perspective-changing, and illumination-changing, the robustness of the existing VO and VSLAM systems still cannot well satisfy the requirements.

To handle the robustness problem, especially the usage of DNN output in challenging environments and maintaining the tracking accuracy, we propose a system framework with DNN-enabled PF in pose estimation optimization.

Instead of estimating the motion from the DNN prediction directly in the existing work [7, 8], we use the image classification results for more general performance. Due to the uncertainty representation in image classification, the generality of the image classification network has been verified [23, 24]. Also, for the robust pose estimation, instead of the single hypothesis in the related work [17], a multihypothesis back-end optimization strategy is designed to overcome the local optimum in optimization, where the feasibility of the multiple hypotheses is guaranteed by the DNN motion prediction.

3. Methodology

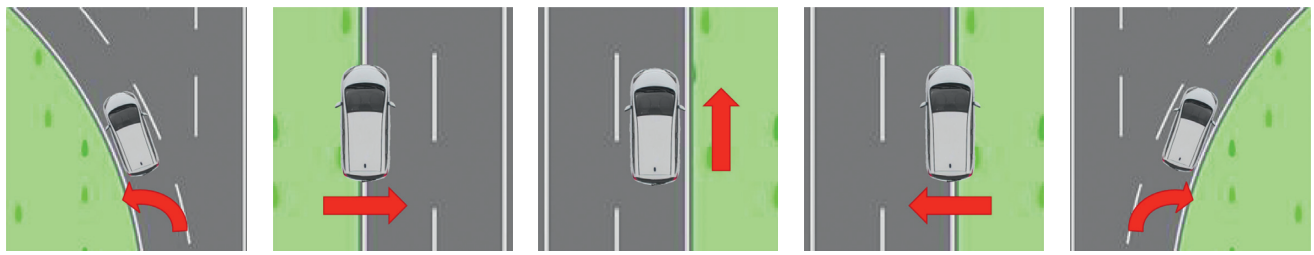
In our method, the robustness improvement is realized by the multihypothesis pose optimization with the PF constructed by DNN image classification. Three parts are involved in the methodology: motion prediction by DNN-based image classification, the PF construction using the motion prediction, and the pose optimization with PF multihypothesis.

3.1. Prior Knowledge from DNN Image Classification. We use the DNN introduced in [25] to provide the motion label prediction, which is regarded as the prior knowledge. Since the motion of the self-driving car is limited, the DNN classification result of the captured image pair is feasible due to its limited motion types: planar rotation and translation. Instead of obtaining the transformation prediction from DNN directly (which is performed in the end-to-end style), according to the motion types, the input image is classified into six types for motion prediction: go-forward, right-side, left-side, no-rotation, turn-left, and turn-right, as shown in Figure 3.

With these six labels, the coefficients in the particle filter of multihypothesis optimization can be determined. Also, it is worth noting that obtaining the training data with motion type classification labels is much easier than collecting the dataset with exact transformation information, because the requirement of ground truth transformation accuracy, as well as the dimension of ground truth label data, is relaxed. Therefore, the cost of the training process is also lower than the method with exact DNN motion transformation prediction.

According to the discussion above, we define three coefficients of the objective function as follows, where r_l, r_r, r_s, m_l, m_r, and m_s are the prediction from the DNN representing the image classification probability, representing the predicted probability of turning left, turning right, no turning, left-side, right-side, and go-straight, respectively:

$$\begin{aligned} C_t^r &= \frac{r_l - r_r}{r_l + r_r + r_s}, \\ C_t^z &= \frac{m_s}{n}, \\ C_t^x &= \frac{m_l - m_r}{n}, \\ n &= \sqrt{m_s^2 + (m_l - m_r)^2}. \end{aligned} \tag{1}$$



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

Figure 3: Illustration of the six types of motion labels in the pretrained DNN and the corresponding motion for adjustment shown as the blue arrow. While go-forward and no-rotation are independent, they are probabilistically correlated, i.e., both go-forward and no-rotation are very likely to have high probabilities. (a) Label turn-left, (b) label left-side, (c) label go-forward, (d) label right-side, and (e) label turn-right and no-rotation.

Since the sum of m_l, m_r, and m_s is 1, while the normalization of translation in monocular transformation is conducted by $L2$-norm, the normalization operation is executed by n here, wherein m_l and m_r are in the same DoF. The coordinate system is shown in Figure 4. The on-road self-driving car is assumed to move in the X-Z plane and rotate about the Y-axis. With the defined coordinate system and the planar motion assumption, the motion prediction transformation matrix v_t can be built according to the obtained C_t^r, C_t^x, and C_t^y as shown in formula (2), and v_t will be utilized in the particle filter establishment.

$$v_t = \begin{bmatrix} \cos(C_r \cdot \theta) & 0 & \sin(C_r \cdot \theta) & C_t^x \cdot \alpha \\ 0 & 1 & 0 & 0 \\ -\sin(C_r \cdot \theta) & 0 & \cos(C_r \cdot \theta) & C_t^z \cdot \alpha \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2}$$

where the two unknown variables are the rotation factor θ and the translation factor α.

In addition to the motion prediction, the corresponding uncertainty u_i predicted by the network is also recorded for the PF construction, which shows the confidence of motion prediction results and indicates the uncertainty for covariance matrix constructing in PF.

3.2. DNN-Enabled Particle Filter

3.2.1. Preliminaries of the Particle Filter. The basic idea of PF is that the belief is represented by a set of samples (also called particles), and the samples are drawn according to the posterior distribution over the prediction. In other words, rather than approximating posteriors in a parametric form, as is in the case for the Kalman filter and the extended Kalman filter, PF represents the posteriors by weighted particles which approximate the desired distribution.

Also, PF algorithm applies a Bayesian iteration to update particle states involving the prediction and update stages. The Bayesian iterative formula is as follows:

Figure 4: The coordinate system in our method.

$$\text{pre}(x_t) = \int p(x_t|u_t, x_{t-1}) \cdot \text{bel}(x_{t-1}) \text{dx}_{t-1}, \tag{3}$$

$$\text{bel}(x_t) = \eta \cdot p(z_t|x_t) \cdot \text{pre}(x_t). \tag{4}$$

As formula (3) shows, $\text{bel}(x_t)$ is the belief obtained from prediction $\text{pre}(x_t)$ by iteration. The kinematic model $p(x_t|u_t, x_{t-1})$ is set to predict the state, and $p(z_t|x_t)$ is denoted as an observation model to update the pose.

Recursively using the prediction and update, the algorithm constantly updates the particle set. The form of the nonlinear kinematic model and the observation model in diversification makes the algorithm extremely robust and suitable for robot localization in almost any field. Moreover, the accuracy of the PF depends to a large extent on the size of the particle set. The larger the size is set, the higher the accuracy can be achieved. If and only if the size of the particle set is huge, the particle distribution is close to the actual distribution unlimitedly.

3.2.2. DNN-Enabled Particle Filter. Based on the original PF mentioned above, combining the motion prediction of DNN, we propose a DNN-enabled PF algorithm, where the initialization is realized by DNN prediction C_t^x, C_t^z, and C_r, and uncertainty u_i, and the resampling process takes the observation from DNN into consideration.

In the initialization, the mean of particles is set in the origin, and the initial covariance is obtained by the prediction uncertainty. Given the motion prediction uncertainty ζ, the covariance matrix Σ_t is defined as

$$\Sigma_t = \mathrm{diag}\left(C_t^x \cdot u_i, \gamma, C_t^z \cdot u_i, \gamma, C_t^r \cdot u_i, \gamma\right), \tag{5}$$

where C_t^x, C_t^z, and C_t^r are the motion prediction results from network. u_i is the corresponding uncertainty of the motion prediction, which is obtained from the Softmax layer of the leveraged network, and Υ is a given value for weight balance.

The mean is given by the last iteration. Therefore, the initialization distribution of PF can be denoted as

$$x_t^i \sim N_t\left(x_t, \Sigma_t\right), \tag{6}$$

where x_t^i is the i th particle at time t and N_t is the Gaussian distribution for the particle sampling.

After initialization, in the motion update process, all the particles are updated according to the motion model and predicted velocity. The velocity is predicted by the visual tracking process with the most weighted particle, which is realized by visual tracking. After each motion update and weight calculation, we calculate $\mathrm{N_{eff}}$ to evaluate the necessary for particle resampling, which is defined as

$$N_{\mathrm{eff}} = \frac{1}{\sum_i \left(w_t^i\right)^2}, \tag{7}$$

where x_t^i is the weight of particle i at time t, which is an indicator of $\mathrm{bel}\left(x_t\right)$. The particle weight is obtained from the residual of optimization Γ_t by the corresponding pose in optimization initialization, which is indicated as follows:

$$w_t^i = \frac{\alpha}{\Gamma_t}, \tag{8}$$

where β is a given value for particle weight generation.

In the resample process, the resample distribution is determined by the motion model and the observation from DNN. In detail, we adopt the resample distribution in [23] for weighted sampling, where the DNN prediction is considered. Given the motion prediction at time t, the most weighted particle pose at time $t-1$ is denoted as T_{t-1}, which is assigned to $x(t-1)^w$. With the transformation estimation v_t generated from C_t^x, C_t^z, and C_t^r, the resample distribution is written as

$$\begin{aligned} &x_t^i \sim N\left(x_t^w, \Sigma_t\right), \\ &\text{s.t. } D\left(x_t^i, f\left(T_{t-1}, v_t\right)\right) < R, \end{aligned} \tag{9}$$

where r is a given threshold to limit the sampling within a range around the network prediction and $f(*)$ is the motion model function, which is written as $p(x_t \mid u_t, x(t-1))$ during the Bayes iteration.

In our proposed PF algorithm, DNN prediction is only leveraged in the resampling and initialization process. Since resampling is not required after every motion update, the DNN prediction can be executed in an interval-style. With such an interval strategy, the real-time performance can be improved for online visual tracking.

3.3. Pose-Graph Optimization with the Particle Filter. With the constructed PF with DNN prediction, the pose estimation in the optimization is conducted with multihypothesis. We give the initial estimation of optimization according to the particle pose and execute the optimization in a parallel way. The optimization function is written according to the definition of reprojection error:

$$T[R|t] = \arg\min_{R,t} \sum_j \left\| p^i - \pi\left(R \cdot P^i + t\right) \right\|_2, \tag{10}$$

where p^i is the feature point in the image plane and P_i is the corresponding 3D projected point. $T[R|t] \in \mathrm{SE}(3)$ is the result of pose optimization, and $\pi(*)$ is the projection function according to the estimated pose.

During the optimization, the initial estimation T_0 is given by the pose of particle filter mentioned above. After optimization, the residual is assigned to the corresponding particle acting as the particle weight. In the optimization, only the pose-graph is involved, instead of the complete graph including map points and pose, whose variable dimension is much smaller to save time consumption. Also, we run the pose-graph optimization within a sliding window, which only considers the neighbor variables that are much related to the current pose estimation. The size of the sliding window is set according to the covisibility graph.

4. Experiments of the VO System

4.1. Experimental Setup. We use the pretrained DNN [26] in this experiment, which has been trained for motion type classification of one image input. The example of the training dataset is shown in Figure 5. Also, we fine-tune the pretrained network on the KITTI dataset, which is in the city self-driving environment. Both sequence 03 and sequence 04 are utilized, and they are not included in the tests. To evaluate our proposed method, we use the following evaluation metrics: pose estimation accuracy by visual localization, the number of optimization iterations, and time consumption in second. The practicality of our method will be evaluated by the runtime performance.

For visual tracking and pose optimization, we use ORB-SLAM3 as the base of our implementation, and the optimization is performed with the DNN-enabled PF algorithm. We build the vision part of this implementation upon the OpenCV library. The Levenberg–Marquardt (LM) solver [27] in the Eigen library is selected as the optimization solver, which is effective in solving nonlinear optimization problems. We run all the experiments on a laptop computer with an Intel Core i7 (8 cores and 16 threads) CPU, 16 GB RAM memory, and RTX-2060 GPU.

4.2. Experimental Results

4.2.1. Visual Tracking Precision. By using the prediction from the DNN classification result, we estimate the trajectories on the KITTI dataset, and the accuracy is indicated by absolutely trajectory Root Mean Square Absolute Trajectory Error (RMS ATE) [28]. The results are listed in Table 1,

Figure 5: Examples of the pretrain dataset and fine-tuned training dataset, which show the different illumination and scene. The upper row is the examples of pretraining data in the forest environment, while the lower row is the fine-tuned training dataset in the self-driving scenario. Also, the tests are conducted on the city environment similar to the fine-tune dataset.

Table 1: Comparisons of RMSE (m) between our method, ORB-SLAM, DSO, and DDVO.

System	00	01	02	05	06	07	08	09	10
Ours	62.28	27.05	14.71	51.87	20.49	73.67	52.32	66.11	18.03
Ours-PF only	66.40	31.44	15.78	57.36	21.45	76.49	55.00	71.66	26.32
ORB-SLAM	66.20	51.18	16.02	56.29	57.71	72.85	53.89	60.44	18.41
DSO	126.70	113.57	27.99	120.17	74.29	80.36	71.69	80.80	30.22
DDVO	—	—	16.70	92.69	21.92	80.59	63.91	62.72	20.31

where the unit of RMS ATE is m. In terms of our proposed method, two groups of the experiment are conducted: the complete proposed method with DNN prediction and the DNN-enabled PF in pose estimation optimization; the multihypothesis optimization by PF only without DNN prediction. These two groups are set up to show the advance of multihypothesis optimization and the DNN prediction in particle sampling.

Since our system is built based on ORB-SLAM, the accuracy of the original ORB-SLAM is shown. As the competitors, some representative work, such as ORB-SLAM [13], DDVO [9], and DSO [2], are included in the comparisons. Note that the loop-closure detection and global optimization in ORB-SLAM are disabled since most of the VO systems do not perform the global optimization.

The comparison results can be seen in Table 1; we generate the ORB-SLAM and DSO experimental results ourselves and extract the DDVO results from the paper [9]. ORB-SLAM and DSO are the representative systems of indirect method and direct method of the model-based style, and DDVO is the VO system leveraging DNN prediction in both environment construction and pose estimation. Our method can obtain better accuracy in most of the sequences. Some examples of the results are shown in Figures 6 and 7, including the estimated trajectories compared with the ground truth, as well as the error curves.

4.2.2. Convergence and Time Consumption. Based on the results above, the real-time performance of our system process example is shown in Figure 8. The time consumption of multihypothesis pose optimization is verified here. Since

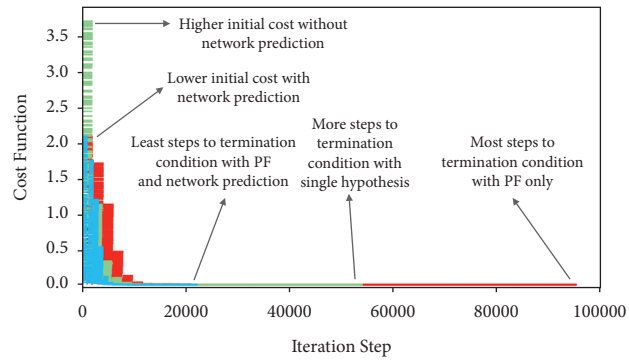


Figure 6: Cost and iteration times. With the same termination condition and step size, the least iteration time is achieved by the optimization with PF and DNN prediction. The efficiency of optimization convergence is improved by the particle sampling with DNN prediction.

the DNN prediction is executed in an interval-style, we count the time consumption of DNN prediction during the whole process. Also, we provide the result of our method with PF only, whose sampling distribution does not consider the prediction from the neural network, and the corresponding iteration process is similar to the PF algorithm introduced in [29].

Also, we provide the real-time performance analysis of the whole system in Table 2. Three groups of experiments are conducted: the traditional optimization with single initialization, called "single hypothesis"; the optimization with multiple hypotheses from PF, while the PF is constructed from the motion model and random

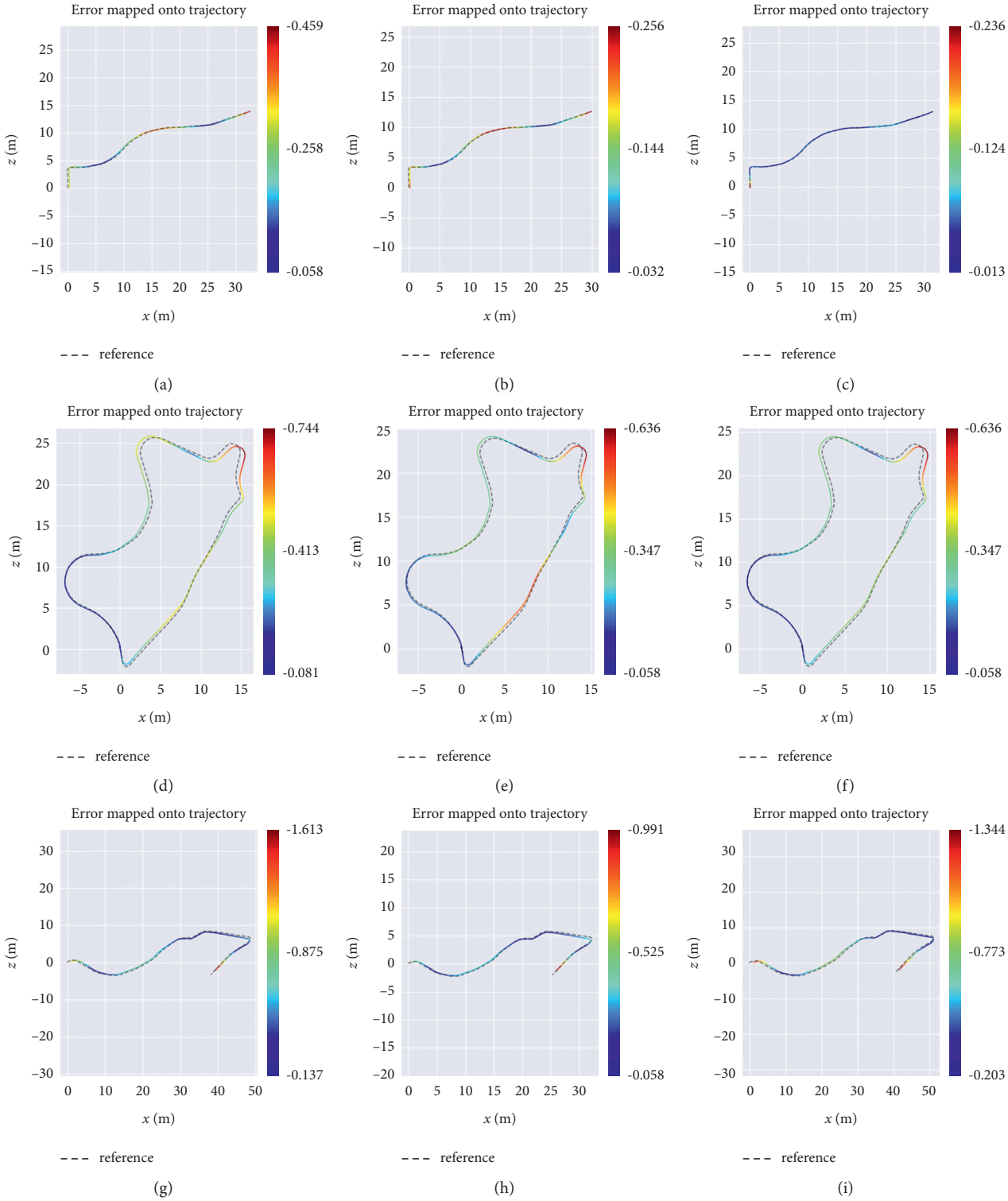


Figure 7: Examples of experimental results on sequences 03, 09, and 10. The trajectories of original system, our method with PF only, and the complete proposed method are shown. The trajectories of (a) the original system on sequence 03, (b) our method with PF only on sequence 03, (c) the complete proposed method on sequence 03, (d) the original system on sequence 09, (e) our method with PF only on sequence 09, (f) the complete proposed method on sequence 09, (g) the original system on sequence 10, (h) our method with PF only on sequence 10, and (i) the complete proposed method on sequence 10.

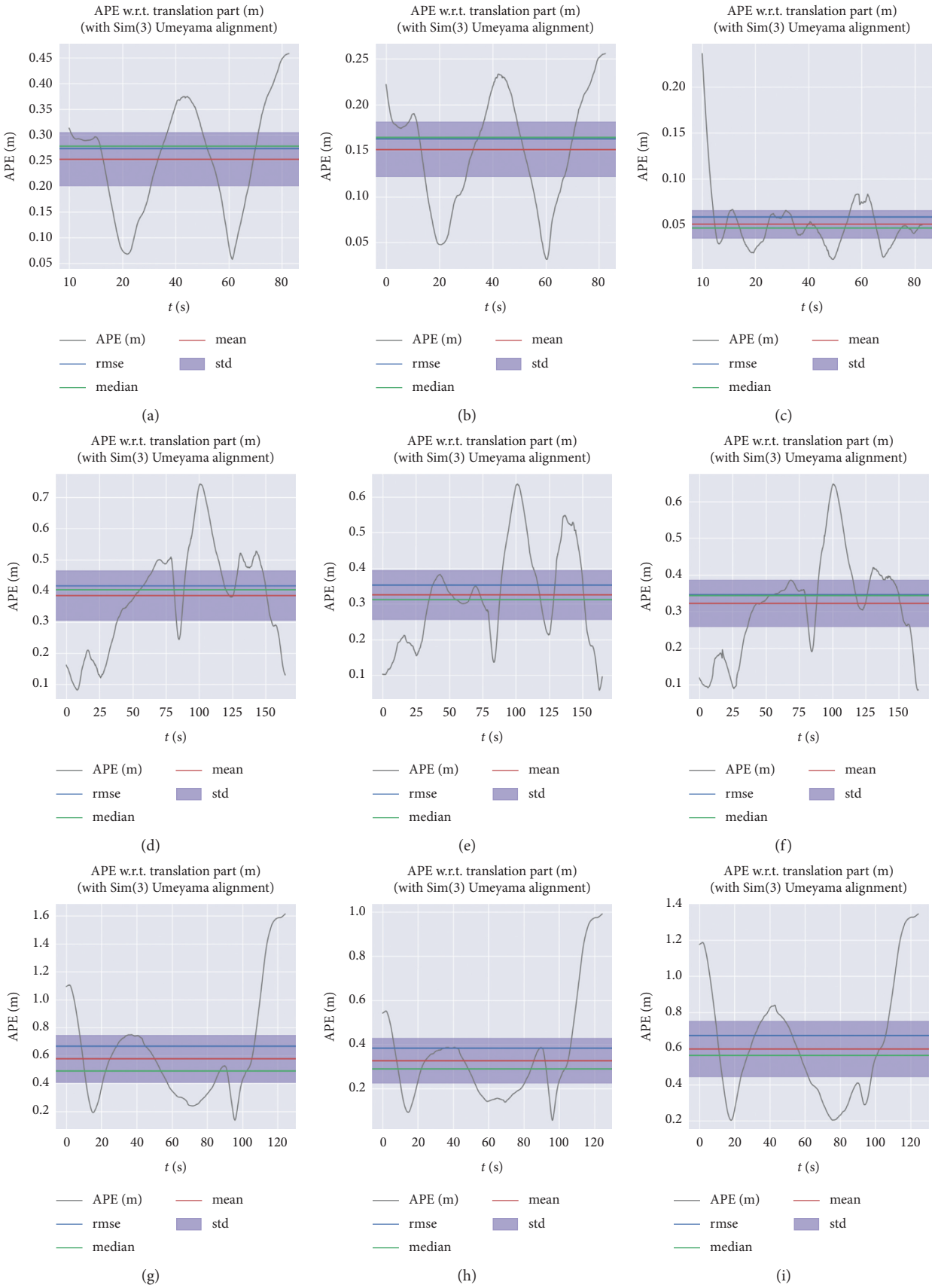


Figure 8: Error curve of experimental results on sequences 03, 09 and 10. The error curve of original system, our method with PF only, and the complete proposed method are shown. The error curve of (a) the original system on sequence 03, (b) our method with PF only on sequence 03, (c) the complete proposed method on sequence 03, (d) the original system on sequence 09, (e) our method with PF only on sequence 09, (f) the complete proposed method on sequence 09, (g) the original system on sequence 10, (h) our method with PF only on sequence 10, and (i) the complete proposed method on sequence 10.

Table 2: Comparison of time consumption(s) on KITTI datasets.

Group name	00	01	02	05	06	07	08	09	10
Singe hypothesis	575.76	83.86	697.46	407.94	148.63	106.52	686.82	208.13	140.55
Multiple hypothesis	808.47	83.65	967.51	486.92	147.05	166.02	758.12	268.45	186.68
Multiple hypothesis with DNN	626.33	91.07	788.39	436.19	163.23	148.10	710.93	238.55	176.15

initialization, called "multiple hypothesis"; the optimization with DNN-enabled PF, where the PF is established with the DNN prediction.

In addition to the time consumption, to show the real-time performance without the consideration of computer configuration, we provide the convergence performance of the DNN-enabled particle filter. Since the time of iteration is related to the efficiency of particle sampling, the advance of particle filter sampling considering the DNN observation can be shown by the convergence performance. Since the feasibility of the sampled particle is enhanced by the DNN prediction, our method is able to converge with a smaller number of iterations than others.

Based on the visual tracking result above, an example of the convergence performance in the multihypothesis optimization process is shown in Figure 6, in which the times of particle filter iteration are shown, as well as the value of cost function. With the same input, the multihypothesis optimization with DNN prediction can satisfy the termination condition with the less iteration steps. Also, the prediction from DNN can provide an appropriate initialization for pose estimation, which is beneficial to the optimization convergence and achieves the fastest convergence. The average framerate of the proposed method on KITTI dataset is 8.1 Hz, whose runtime performance is able to meet the requirement of most applications.

4.3. Discussion. Here we present the analysis according to the results in Tables 1 and 2. Regarding the comparisons in terms of tracking accuracy, our proposed method has better performance than the popular systems and has the smaller tracking error indicated by RMS ATE; the examples of visual tracking trajectory are shown in Figures 6 and 7.

Compared with the popular VSLAM systems, such as ORB-SLAM, DSO, and DDVO, the proposed method with multihypothesis optimization initialization can converge to the global optimal solution with a higher probability and less iteration. As is shown in the tables, the multiple hypothesis optimization with PF only can obtain better tracking accuracy than the existing systems, while its error is still higher than the trajectory estimation with neural network prediction. Because the prediction from the neural network gives high efficiency to the sampled particles, the convergence performance can be improved by the sampling distribution with appropriate observation and prediction.

As for the real-time performance of optimization, the proposed method with neural network prediction is less time-consuming than the one without network prediction. Because the neural network prediction provides a feasible distribution for the particle sampling, the time consumption for particle convergence is reduced. Also, the optimization initialization by DNN-enabled PF is able to provide a lower cost for fast iteration convergence, as is shown in the convergence analysis shown in Figure 8. The optimization with multihypothesis only needs the most iteration steps because the feasibility of PF samples is not guaranteed. Also, comparing with the existing methods that consider a single hypothesis of initialization during optimization, our method still spends more time. However, regarding the application of visual tracking, the average frame rate of our proposed method can still satisfy most of the applications, and the practicality of our proposed method is verified.

5. Conclusion

In this paper, we propose a method to demonstrate the robustness and accuracy introduced by the multihypothesis pose estimation with the proposed DNN-enabled PF. The image classification DNN is used in our method, which provides the motion label prediction for optimization initialization. Besides, we introduce the DNN-enabled PF for improving the particle distribution, where both the motion prediction and prediction uncertainty are considered. Instead of estimating the motion from the DNN prediction directly, we use the high-level representation for more general performance. Also, for the robust pose estimation, a multihypothesis back-end optimization strategy is designed to overcome the local optimum in optimization. The scalability of our method is guaranteed by the improved generalization, which can meet the requirement of many applications.

With the robustness introduced by our method, the higher accuracy of visual tracking than existing work is achieved. The experiment result built based on ORB-SLAM shows the advance of our proposed multiple hypothesis optimization and the DNN-enabled particle filter, where the average accuracy is improved by 13.3%. In the future, the extension of our work is the application to unman-drones and other VO or VSLAM systems that require a higher degree of freedom and more generalized performance. Also, the given parameters can be tuned by the learning method in our future work.

Data Availability

Because of the confidentiality of the college, the data cannot be made public.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] H. Strasdat, J. M. M. Montiel, and A. J. Davison, "Visual SLAM: why filter?" *Image and Vision Computing*, vol. 30, no. 2, pp. 65–77, 2012.

[2] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 4, 2017.

[3] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "SVO: semidirect visual odometry for monocular and multicamera systems," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 249–265, 2017.

[4] J. Deng, W. Dong, R. Socher, L. J. Li, and F. F. Li, "Imagenet: a largescale hierarchical image database," in *Proceedings of the 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009)*, Miami, Florida, USA, June 2009.

[5] C. Cadena, L. Carlone, H. Carrillo et al., "Past, present, and future of simultaneous localization and mapping: toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[6] M. Zhang, J. Li, Y. Li, and R. Xu, "Deep learning for short-term voltage stability assessment of power systems," *IEEE Access*, vol. 9, pp. 29711–29718, 2021.

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

[8] R. Li, S. Wang, Z. Long, and D. Gu, "Undeepvo: monocular visual odometry through unsupervised deep learning," in *Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7286–7291, IEEE, Brisbane, Australia, May 2018.

[9] C. Zhao, Y. Tang, and Q. Sun, "Deep direct visual odometry," 2019, https://arxiv.org/abs/1912.05101.

[10] A. Handa, M. Bloesch, V. Pătrăucean, S. Stent, J. McCormac, and A. Davison, "gvnn: neural network library for geometric computer vision," in *Proceedings of the European Conference on Computer Vision*, pp. 67–82, Springer, Amsterdam, The Netherlands, October 2016.

[11] S. Milz, G. Arbeiter, C. Witt, B. Abdallah, and S. Yogamani, "Visual slam for automated driving: exploring the applications of deep learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 247–257, Salt Lake City, UT, USA, June 2018.

[12] A. Forechi, T. Oliveira-Santos, C. Badue, and A. F. De Souza, "Visual global localization with a hybrid WNN-CNN approach," in *Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–9, IEEE, Rio de Janeiro, Brazil, July 2018.

[13] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[14] S. Y. Loo, A. J. Amiri, S. Mashohor, S. H. Tang, and H. Zhang, "CNN-SVO: improving the mapping in semi-direct visual odometry using singleimage depth prediction," in *Proceedings of the 2019 International Conference on Robotics and Automation (ICRA)*, pp. 5218–5223, IEEE, Montreal, Canada, May 2019.

[15] K. Tateno, F. Tombari, I. Laina, and N. Navab, "CNN-SLAM: real-time dense monocular slam with learned depth prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6243–6252, Honolulu, HI, USA, July 2017.

[16] J. Engel, T. Sch¨ops, and D. Cremers, "LSD-SLAM: large-scale direct monocular slam," in *Proceedings of the European Conference on Computer Vision*, pp. 834–849, Springer, Zurich, Switzerland, September 2014.

[17] N. Yang, L. von Stumberg, R. Wang, and D. Cremers, "D3VO: deep depth, deep pose and deep uncertainty for monocular visual odometry," 2020, https://arxiv.org/abs/2003.01060.

[18] A. Geiger, M. Lauer, C. Wojek, C. Stiller, and R. Urtasun, "3D traffific scene understanding from movable platforms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 5, pp. 1012–1025, 2013.

[19] F. Codevilla, M. Miiller, A. L´opez, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," in *Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1–9, IEEE, Brisbane, Australia, May 2018.

[20] C. Zhao, L. Sun, Z. Yan, G. Neumann, T. Duckett, and R. Stolkin, "Learning Kalman network: a deep monocular visual odometry for on-road driving," *Robotics and Autonomous Systems*, vol. 121, Article ID 103234, 2019.

[21] M. Bojarski, D. Del Testa, D. Dworakowski et al., "End to end learning for self-driving cars," 2016, https://arxiv.org/abs/1604.07316.

[22] H. Xu, Y. Gao, F. Yu, and T. Darrell, "End-to-end learning of driving models from large-scale video datasets," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2174–2182, Brisbane, Australia, July 2017.

[23] T. Iwata and Z. Ghahramani, "Improving output uncertainty estimation and generalization in deep learning via neural network Gaussian pro-cesses," 2017, http://https//arxiv.org/abs/1707.05922.

[24] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: off-policy maximum entropy deep reinforcement learning with a stochasticactor," in *Proceedings of the International Conference on Machine Learning*, pp. 1861–1870, PMLR, Vienna, Austria, June 2018.

[25] N. Smolyanskiy, A. Kamenev, J. Smith, and S. Birchfifield, "Toward low-flflying autonomous mav trail navigation using deep neural networks for environmental awareness," in *Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4241–4247, IEEE, Vancouver, Canada, September 2017.

[26] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[27] A. Ranganathan, "The levenberg-marquardt algorithm," *Tutoral on LM algorithm*, vol. 11, no. 1, pp. 101–110, 2004.

[28] E. Mueggler, H. Rebecq, G. Gallego, T. Delbruck, and D. Scaramuzza, "The event-camera dataset and simulator: event-based data for pose estimation, visual odometry, and SLAM," *The International Journal of Robotics Research*, vol. 36, no. 2, pp. 142–149, 2017.

[29] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte carlo localization for mobile robots," in *Proceedings of the IEEE International Conference on Robotics & Automation*, Detroit, MI, USA, May 2002.