Multi-Camera Sensor Fusion for Visual Odometry using Deep Uncertainty Estimation

Nimet Kaygusuz[1], Oscar Mendez[1], Richard Bowden[1]

Abstract— Visual Odometry (VO) estimation is an important source of information for vehicle state estimation and autonomous driving. Recently, deep learning based approaches have begun to appear in the literature. However, in the context of driving, single sensor based approaches are often prone to failure because of degraded image quality due to environmental factors, camera placement, etc. To address this issue, we propose a deep sensor fusion framework which estimates vehicle motion using both pose and uncertainty estimations from multiple on-board cameras.

We extract spatio-temporal feature representations from a set of consecutive images using a hybrid CNN - RNN model. We then utilise a Mixture Density Network (MDN) to estimate the 6-DoF pose as a mixture of distributions and a fusion module to estimate the final pose using MDN outputs from multi-cameras.

We evaluate our approach on the publicly available, large scale autonomous vehicle dataset, nuScenes. The results show that the proposed fusion approach surpasses the state-of-the-art, and provides robust estimates and accurate trajectories compared to individual camera-based estimations.

I. INTRODUCTION

Perception of the environment via sensor systems is a major challenge in the quest to realise autonomous driving. To ensure safety, accurate and robust systems are required that can operate within any environment. One of the most fundamental components of any autonomous system is ego-motion estimation which can be used for both vehicle control and automated driving.

In recent years, considerable research effort has been invested into ego-motion estimation using on-board sensors [1]. It is common to employ various types of complementary sensors, e.g., IMU, GPS, LiDAR and cameras, to provide accurate motion for vehicle state estimation systems.

Cameras are a popular choice as they are cost effective, already installed on most modern vehicles and we know that vision is key to human driving ability. However, the accuracy of vision-based motion estimation is affected by various internal and external factors. Even if we assume that camera calibration is known and correct, the quality of the captured images are dependent on environmental factors such as occlusions, lighting and weather conditions. Considering that Visual Odometry (VO) works by examining the change in image pixel motion between consecutive frames, robust and distinctive features are crucial for its performance. This means that the quality of estimated motion is dependent upon the quality of images, which in turn is dependent upon the type and structure of the scene.

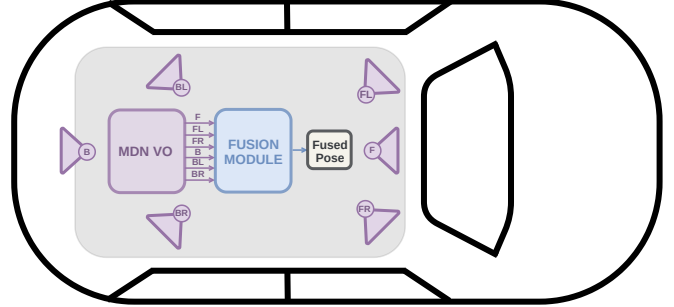[1]All authors with the University of Surrey {n.kaygusuz, o.mendez, r.bowden}@surrey.ac.uk

Fig. 1: An overview of the proposed multi-camera fusion approach.

Robustness can be provided in many ways, but for vehicle state estimation, sensor fusion techniques are often employed. Many successful sensor fusion frameworks employ a Kalman filter (or variant) [2], [3]. Kalman filters are designed to handle the sensor measurement noise and allow the combination of complementary sensors. However, measurement statistics and system dynamics needs to be known for the filter to operate successfully. As image quality is dependent upon environmental factors, fusion of multiple cameras in a VO system requires estimates of sensor reliability or measurement noise and this implies implicit knowledge about what environmental factors lead to degradation.

Motivated by recent developments in deep learning and its applications to autonomous vehicles, we propose a learning based sensor fusion framework which fuses multiple VO estimates from multiple cameras, taking into account the predicted confidence of each sensor.

We utilise a hybrid Convolutional Neural Network (CNN) - Recurrent Neural Network (RNN) architecture that extracts spatio-temporal features from images. We then employ a Mixture Density Network (MDN) which predicts the probability distributions of motion for each camera independently. This approach allows our model to learn the confidence/uncertainty of motion for each camera. Independent motion estimates from multiple arms of the network are then combined in a neural fusion module. An overview of our approach can be seen in Figure 1.

We evaluate the proposed fusion approach on nuScenes [4] which is a large scale, public, autonomous driving dataset. We report quantitative and qualitative VO results. Our experiments show that, overall, the proposed fusion approach can provide more accurate estimations than individual cameras alone. Furthermore, our model surpasses the two state-of-

the-art VO algorithms and baseline Extended Kalman Filter (EKF) based fusion.

The contributions of this work can be summarised as:

- We utilise a deep neural network approach to estimate both vehicle motion and the confidence in that estimation, from multiple monocular cameras.
- We propose a deep fusion framework that can intelligently fuse multiple motion estimates incorporating the pose and uncertainties in prediction.
- We demonstrate state-of-the-art performance on a large-scale autonomous driving dataset and evaluate performance under different weather and lighting conditions.

The rest of this paper is structured as follows: In Section II we discuss relevant related work. We then introduce the proposed learning based fusion model in Section III. We share the experimental setup and report our results in Section IV. Finally, we conclude the paper in Section V by discussing our findings.

II. RELATED WORK

This paper focuses on estimating vehicle motion from multiple on-board cameras. VO can be defined as estimating the motion of a camera by examining the motion of features between consecutive image frames. Traditionally, there are two families of approaches to estimating vehicle motion via vision sensors, namely direct methods and feature based methods. Direct methods [5], [6] track the changes in pixel intensities to estimate the camera motion. Feature based methods [7], [8] use handcrafted features to detect salient patterns and track their displacement on the image plane in order to estimate the camera motion [9]. Even though both methods show promising results, they both have their short comings. Feature based methods are prone to failure in low texture environments or with insufficient lighting, while direct methods tend to fail at high velocities and are sensitive to photometric changes.

More recently, deep learning based models have been applied to the VO task [10], [11], [12]. One of the main advantages of these models is not requiring handcrafted features, due to their ability to learn meaningful feature representations for the task they were trained upon. Compared to classical VO approaches, they are more robust in low illumination and texture-less environments [1]. In [13], Mohanty et al. proposed estimating motion via monocular camera from consecutive frames using CNNs which were trained in a supervised manner. Wang et al. [10], [14] expanded this approach and introduced the use of RNN to model the temporal dependencies between frames. In this work, we employ a similar supervised approach to estimate motion for individual cameras. However, instead of directly regressing a 6-DoF pose, we estimate the motion as a mixture of Gaussians using MDN's. This enables our model to estimate pose and its uncertainty together which form the input of our fusion framework.

One of the main shortcomings of supervised deep learning approaches is their need to be trained on large scale datasets. To address this issue, unsupervised VO methods have been studied. Zhou et al. [15] proposed learning depth maps and camera pose change together with view synthesis. Li et al. [11] utilised stereo images which enabled their network to recover the global scale. More recently, Yang et al. [16] proposed learning depth, pose and photometric uncertainty in an unsupervised manner, where they used photometric uncertainties to optimise the VO estimates. Although unsupervised VO methods have the potential of exploiting unlabelled data, these approaches do not have the necessary robustness for vehicle state estimation.

To improve the robustness of vehicle motion estimation, sensor fusion techniques have been employed [17], [18], [19]. However, most of the sensor fusion studies focus on using VO as a complementary source of information. Parra et al. [17] use VO when GPS fails. Lynen et al. fuse Inertial Measurement Unit (IMU) and vision sensors using a Kalman filter. Chen et al. [19] proposed a fusion framework which selectively fuses a monocular camera and IMU by eliminating the corrupted sensor measurements.

To perform more reliably than any individual sensor, sensor fusion techniques require measurement uncertainty. But building an accurate uncertainty model for sensors is a difficult task [20]. In addition to the sensor imperfections, adverse environmental conditions have a significant effect on sensor output. For example, visual sensors, such as colour cameras, are particularly sensitive to environmental factors such as direct sunlight, low texture areas or poor lighting. Thus, semantic understanding of the scene is required in order to estimate the accuracy of vision sensors. In this work we model the measurement uncertainty using an MDN. By estimating a probability distribution over the motion for each camera, we can robustly fuse multiple sensors together in a single multi-stream neural network to overcome the possible failure of individual sensors.

III. METHODOLOGY

In this section, we introduce the proposed multi-camera fusion approach for estimating VO. Our approach starts by acquiring video streams from multiple cameras that are mounted to the vehicle with different positions & orientations. Given video streams $V^{1:N} = \{I_0^n, ..., I_T^n\}^{1:N}$ from N cameras with $(T + 1)$ number of frames, our model predicts the 6-DoF relative poses $Y = \{y_{(0,1)}, ..., y_{(T-1,T)}\}$ between consecutive time steps.

For each camera, we first extract spatio-temporal feature representations using a hybrid CNN - RNN architecture (Section III-A). We then pass these representations to an MDN module (Section III-B), which predicts a mixture of Gaussian distributions for the vehicle pose. By running multiple streams of VO through different arms of the network simultaneously, we then combine the pose distributions coming from multiple cameras in a deep fusion module (Section III-C). This produces a final pose estimate. An overview of the proposed multi-camera fusion approach is visualised in Figure 2.
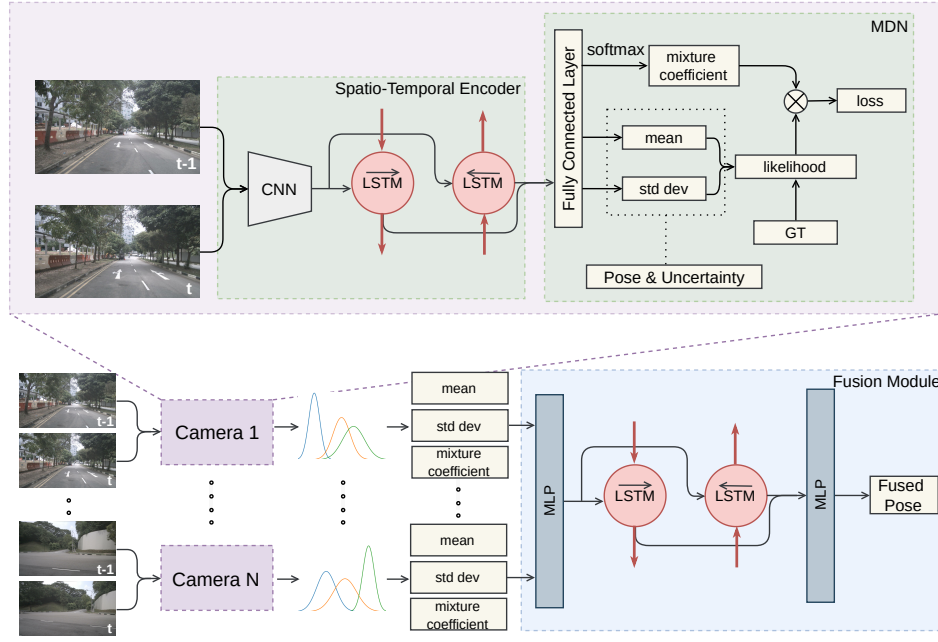
Fig. 2: An overview of the proposed multi-camera sensor fusion for VO. The fusion module uses pose and uncertainty estimations from the MDN module.

A. Spatio-Temporal Encoder

Estimating motion from video requires both spatial and temporal understanding. Temporal modelling can be further divided into short term and long term tracking. While we are interested in short term changes between consecutive frames, i.e. relative pose, we also want to model the long term motion of the vehicle to help produce consistent estimates.

In this work, we model spatial and short-term temporal representations using a CNN backbone similar to those used for optical flow estimation [21], which is a related task. Given a consecutive image pair $[I_{t-1}^n, I_t^n]$ from the nth camera, we extract the motion features, f_t^n as:

$$f_t^n = \mathrm{CNN}([I_{t-1}^n, I_t^n]) \tag{1}$$

where $[.]$ is the concatenation operation over the image colour channels. The CNN network is composed of 9 convolutional layers. After each convolutional layer, we include batch normalisation [22], ReLU [23] and Dropout [24].

To model the longer term vehicle motion, we feed the features from the CNN, $f_{1:T}^n$, to an RNN module. At each time step t, the RNN produces temporally enhanced representations r_t^n as:

$$r_t^n = \mathrm{RNN}(f_t^n, h_{t-1}^n) \tag{2}$$

where f_t^n and h_{t-1}^n are visual features and the previous hidden states at time t for the nth camera, respectively. We initialise the hidden state as all zeros for $t = 1$. In this work we employed a bi-directional Long Short-Term Memory (LSTM) [25] and used a small sliding window of past 5 frames. However, any RNN architecture can be utilised with our approach.

B. Mixture Density Network

Learning based approaches commonly suffer from the problem of regression to the mean, which can be considered as approximating the conditional average of the output. To address this issue, we utilise an MDN, which estimates a mixture of distributions of the 6-DoF relative poses.

We choose to construct our mixture model from Gaussian distributions to model the vehicle's pose. For each camera n, our model estimates a mixture model, $\mathcal{G}^n$, with M components (5 in our experiments) at every time step t, which can be notated as:

$$\mathcal{G}_t^n = \{(\alpha_t^1 \mathcal{N}_t(\mu^1, \sigma^1))^n, ..., (\alpha_t^M \mathcal{N}_t(\mu^M, \sigma^M))^n\} \tag{3}$$

where μ, σ and α represent the mean, standard deviation and mixture coefficients, respectively. This approach enables us to model the variances in the output and provide an estimate of how confident our model is in its estimation.

Given the LSTM outputs of the nth camera, r_t^n, we estimate the conditional density of the 6-DoF pose, $y_{(t-1,t)}$, for the ith mixture component, $\phi_i(y_{(t-1,t)}^n | r_t^n)$, as:

$$\phi\left(y_{(t-1,t)}^n | r_t^n\right) = \frac{1}{(\sigma_i(r_t))^n \sqrt{2\pi}} e^{\left\{-\frac{\|y_{(t-1,t)}^n - (\mu_i(r_t))^n\|^2}{2(\sigma_i(r_t)^2)^n}\right\}} \tag{4}$$

where $(\mu_i(r_t))^n$ and $(\sigma_i(r_t))^n$ denote mean and standard deviation of ith distribution, conditioned on the features, r_t^n. The probability density of the 6-DoF pose, is represented as a linear combination of mixture components as:

$$p\left(y^n_{(t-1,t)}|r^n_t\right) = \sum_{i=1}^{M} (\alpha_i(r_t))^n \phi_i\left(y^n_{(t-1,t)}|r^n_t\right) \quad (5)$$

where $(\alpha_i(r_t))^n$ is the mixture coefficient which represents the probability of the pose being generated from i^{th} component.

Finding the appropriate weights for our model can be achieved by maximising the likelihood $\mathcal{L}^n$. Thus, we train our model by minimising the negative log likelihood of the ground truth being generated by the mixture distribution as:

$$E^n = -\log(\mathcal{L}^n) \quad (6)$$

We train MDN modules independently for each view. Estimating parameters of a mixture of distributions [26] allows our model to predict the variance of the pose which explicitly represents the VO uncertainty.

C. Uncertainty Based Deep Sensor Fusion

The fusion module within the network combines the predictions from individual cameras and estimates a final 6-DoF pose transformation. In a similar fashion to using covariance matrices in a Kalman filter to model measurement uncertainties, we use the output of each camera MDN (including the means, standard deviations and mixture coefficients) as input to the fusion module.

We concatenate the mixture model estimates $\{\mathcal{G}^1_t, ..., \mathcal{G}^N_t\}$ from N cameras, and project them to a latent space using a Multilayer Perceptron (MLP) with dropout and ReLU activation as:

$$p_t = \text{MLP}([\mathcal{G}^1_t, ..., \mathcal{G}^N_t]) \quad (7)$$

where $\mathcal{G}^n_t$ is the mixture model parameters of the nth camera at time t (See Equation 3). To encourage a temporally consistent final pose estimate, we employ an RNN model. We pass MLP outputs $p_{0:T}$ to the RNN layer and extract temporal representation, q_t as:

$$q_t = \text{RNN}(p_t, h_{t-1}) \quad (8)$$

where h_{t-1} is the hidden state from the previous time step. Finally, we employ a final MLP layer to estimate the fused 6-DoF relative pose, z_t as:

$$z_t = \text{MLP}(q_t) \quad (9)$$

We train our fusion module using Mean Squared Error (MSE) loss function. To balance the translation and rotation errors we apply a weighted sum and calculate the error as:

$$\mathcal{E} = \frac{1}{T}\sum_{t=1}^{T} ||y^\rho_{(t-1,t)} - z^\rho_t||^2_2 + \lambda_\varphi ||y^\varphi_{(t-1,t)} - z^\varphi_t||^2_2 \quad (10)$$

where $y^\rho_{(t-1,t)}$ and $y^\varphi_{(t-1,t)}$ denote the translation and rotation (Euler angles) of the ground truth relative poses, respectively. λ_φ is an empirically chosen hyper-parameter to weight the rotation errors, which is set to be 100 in our experiments.

IV. EXPERIMENTS

A. Dataset and Implementation Details

We evaluate our approach on the recently released nuScenes dataset [4], which is a large-scale autonomous driving dataset consisting of 1.4 million camera images containing different types of manoeuvres (velocities ranging from 0 km/h to 63 km/h), weather and lighting conditions. It has six cameras attached at different locations & orientations around the vehicle. The cameras have $70°$ FOV for front and side and $110°$ FOV for rear providing a 360 degree view. This makes nuScenes an ideal dataset for multi-camera fusion. nuScenes has 850 driving sequences with available ground truth poses. Among them we eliminate the static scenarios, where the vehicle does not move in the sequence. It should be noted that the remaining scenes already contain static parts of a trajectory which can be easily learned by the model. We split the sequences for training and test, yielding 676 and 100 sequences in each set, respectively.

We implemented our network using the PyTorch deep learning framework [27]. Adam optimiser [28] is used to train our network with a learning rate of 10^{-3} ($\beta_1 = 0.9, \beta_2 = 0.999$). We utilise plateau learning rate scheduling with a patience of 8 and a decay factor of 0.7. We use pre-trained FlowNet [21] weights to initialise our CNN backbone. We use Xavier initialisation [29] for the remaining parameters. We train our networks on a machine equipped with an NVIDIA Titan X GPU. The training takes approximately 70 epochs. At inference, our model runs at 20 ms per frame on the same machine. Considering nuScenes cameras run at 12 Hz, the proposed approach meets real time requirements. To evaluate the performance of our approach, we use the *evo* python package [30] and report Relative Pose Error (RPE) since it measures the local accuracy of a trajectory, which is a standard way to evaluate VO systems.

B. Comparing Fusion Results with Individual Cameras

In our first set of experiments, we compare the performance of the proposed multi-camera fusion approach against the VO estimations from individual cameras. As mentioned in Section III-B, the MDN module estimates a mixture of distributions over the 6-DoF pose for each camera. Here, we use the mean of the estimated Gaussians as their VO predictions and compare them to our fusion module's estimations (See Section III-C) to see the efficacy of the proposed approach compared to using only single camera estimations.

TABLE I: Relative pose errors of monocular VO and the proposed fusion approach.

Camera Views	RMSE	Max	Mean $\pm$ std
FRONT	0.077	0.531	0.049 $\pm$ 0.058
FRONT LEFT	0.086	0.536	0.057 $\pm$ 0.062
FRONT RIGHT	0.087	0.563	0.056 $\pm$ 0.066
BACK	0.090	0.621	0.054 $\pm$ 0.070
BACK LEFT	0.111	0.599	0.073 $\pm$ 0.082
BACK RIGHT	0.085	0.531	0.056 $\pm$ 0.064
FUSION	**0.045**	**0.171**	**0.035 $\pm$ 0.028**

(a) Daylight / scene-0303
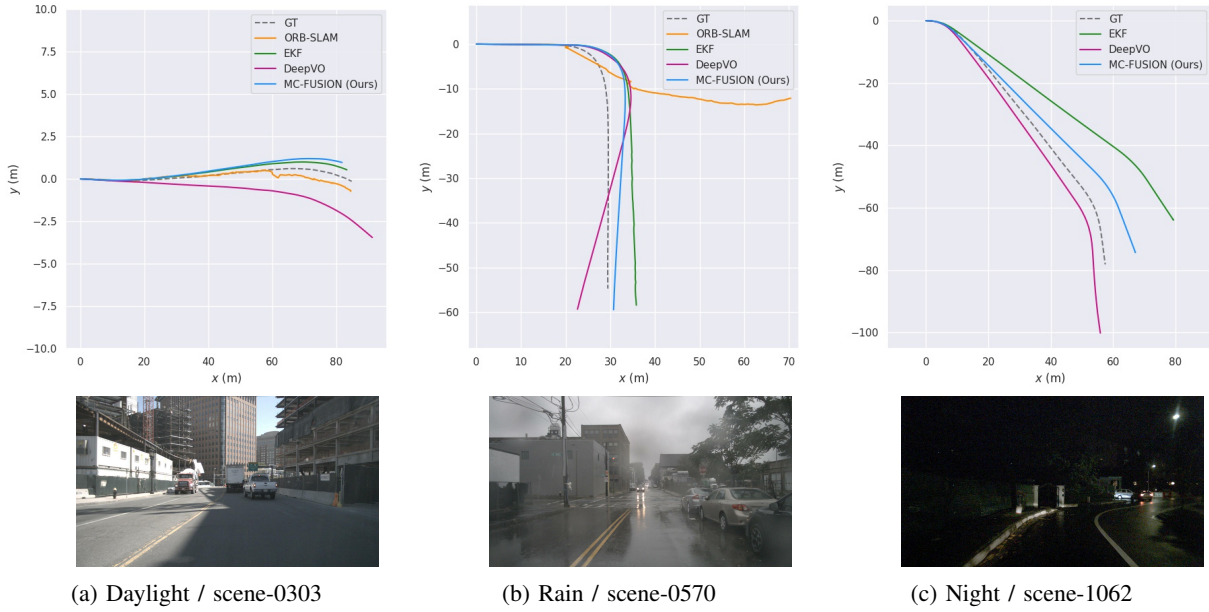
(b) Rain / scene-0570

(c) Night / scene-1062

Fig. 3: Estimated trajectories (top) and sample frames (bottom) from the corresponding nuScenes sequences.

TABLE II: Relative pose errors on the nuScenes test sequences, categorised by weather/lighting conditions. NR represents sequences where an approach fail to produce results.

| | Daylight (64 seq.) | | | Rain (12 seq.) | | | Night (24 seq.) | | |
	RMSE	Max	Mean $\pm$ std	RMSE	Max	Mean $\pm$ std	RMSE	Max	Mean $\pm$ std
ORB-SLAM [8]	0.40	3.28	0.12 $\pm$ 0.38	0.76	10.19	0.14 $\pm$ 0.74	NR	NR	NR
DeepVO [10]	0.09	0.54	0.06 $\pm$ 0.07	0.07	0.39	0.05 $\pm$ 0.05	0.12	0.74	0.09 $\pm$ 0.09
EKF [31]	0.07	0.35	0.05 $\pm$ 0.05	0.06	0.36	0.04 $\pm$ 0.04	0.16	0.86	0.8 $\pm$ 0.12
MC-Fusion (Ours)	**0.04**	**0.15**	**0.03 $\pm$ 0.02**	**0.04**	**0.14**	**0.03 $\pm$ 0.03**	**0.07**	**0.26**	**0.05 $\pm$ 0.04**

Table I shows the VO performance from six different cameras compared to the result of our fusion model. We use the average RPE on the nuScenes test split which consist of 100 unseen sequences. As can be seen, the fusion results outperform all individual camera based estimations. This is because the proposed fusion model can employ complementary information from the different views. This demonstrates the efficacy of our multi-camera fusion approach and verifies that VO accuracy can be improved using the proposed uncertainty-based fusion approach.

C. Comparison against the state-of-the-art

We now compare our fusion model, which already outperforms all individual camera based estimates, against other state-of-the-art methods on the nuScenes dataset. In our comparisons, we use monocular ORB-SLAM [8], DeepVO [10] and the ROS implementation of an EKF [31]. We use the front camera images for ORB-SLAM and DeepVO. To be able to compare ORB-SLAM in the context of monocular VO, we run it without loop-closure following the protocols of [11], [14]. Since classical monocular VO algorithms do not recover the absolute scale for their estimated trajectories, we scale & align ORB-SLAM trajectories with the ground truth using a least-squares similarity transform [32]. For DeepVO results, we use its PyTorch implementation[1] and for a fair

comparison we train it with the same training split as used to train our fusion module. In order to evaluate EKF fusion performance, it is necessary to provide the algorithm with accurate pose and uncertainty estimates. In this test, we wish to compare our deep fusion to a traditional EKF fusion approach. As such, we feed the MDN outputs from each individual camera, including their estimated covariances, to the EKF.

In Table II, we share the RPE for three different categories based on weather/lighting conditions e.g. daylight, rain and night time. As a classical VO algorithm, ORB-SLAM works by extracting and tracking salient features in consecutive frames. Thus, it needs sufficient illumination and texture in the environment to be able to work effectively. This explains why, in our experiments, ORB-SLAM can often not initialise successfully for night time scenarios. Thus, we could not report its qualitative (See Figure 3c) and quantitative (See Table II) analysis for night time driving conditions. In contrast, deep learning based approaches do produce results for night time scenarios, which is one of the most challenging conditions to estimate VO due to the reduced visibility.

The Kalman filter is sensitive to initial parameter selection e.g. the initial estimate of covariance, process noise etc. We choose the set of parameters that yielded the best performance on average across the 100 test sequences. Under

---

[1] https://github.com/ChiWeiHsiao/DeepVO-pytorch/

this setup, the EKF produces better results than ORB-SLAM. In the daylight and rain scenarios, the EKF also performs better than DeepVO. However, for the night sequences, it has a larger error. This can be explained by the fact that an optimal Kalman filter requires careful parameter tuning, which is difficult to achieve for both daytime and nighttime. However, our fusion based approach outperforms best across all scenarios.

We share qualitative examples from all three categories (daylight, rain and night) in Figure 3. This figure shows the estimated trajectories against ground truth, and a sample image from the corresponding scene.

As can be seen, all approaches estimate the trajectories accurately in daylight condition. However, ORB-SLAM's performance decreases significantly in rain and night time scenarios. Deep approaches maintain accurate trajectories even in these challenging rain and night-time conditions. Most significantly, our multi camera fusion approach (MC-FUSION) outperforms other methods estimating more accurate trajectories.

V. CONCLUSION

This paper presents a novel deep learning based multi-camera fusion framework to estimate VO and evaluate it on the nuScenes dataset. Exploiting the advantages of deep learning, we show that the proposed approach can estimate trajectories even in night-time scenarios which is challenging for classical VO approaches. Furthermore, we demonstrate that our proposed fusion model outperforms single camera based estimations by exploiting the complementary information from multiple camera views. Finally, we validate the performance of our approach by comparing it with the state-of-the-art methods namely, DeepVO and ORB-SLAM. We also show that the efficacy of our single camera based pose and uncertainty estimations by feeding the EKF with them.

REFERENCES

[1] C. Chen, B. Wang, C. X. Lu, N. Trigoni, and A. Markham, "A survey on deep learning for localization and mapping: Towards the age of spatial machine intelligence," *arXiv preprint arXiv:2006.12567*, 2020.

[2] F. Caron, E. Duflos, D. Pomorski, and P. Vanheeghe, "Gps/imu data fusion using multisensor kalman filtering: introduction of contextual aspects," *Information fusion*, vol. 7, no. 2, 2006.

[3] G. Wang, Y. Han, J. Chen, S. Wang, Z. Zhang, N. Du, and Y. Zheng, "A gnss/ins integrated navigation algorithm based on kalman filter," *IFAC-PapersOnLine*, vol. 51, no. 17, 2018.

[4] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[5] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "Dtam: Dense tracking and mapping in real-time," in *2011 international conference on computer vision*. IEEE, 2011.

[6] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, 2017.

[7] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 6, 2007.

[8] 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, 2015.

[9] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *IEEE robotics & automation magazine*, vol. 18, no. 4, 2011.

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

[11] 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.

[12] 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.

[13] V. Mohanty, S. Agrawal, S. Datta, A. Ghosh, V. D. Sharma, and D. Chakravarty, "Deepvo: A deep learning approach for monocular visual odometry," *arXiv preprint arXiv:1611.06069*, 2016.

[14] 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, 2018.

[15] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[16] N. Yang, L. v. Stumberg, R. Wang, and D. Cremers, "D3vo: Deep depth, deep pose and deep uncertainty for monocular visual odometry," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[17] I. Parra, M. A. Sotelo, D. F. Llorca, C. Fernández, A. Llamazares, N. Hernández, and I. García, "Visual odometry and map fusion for gps navigation assistance," in *2011 IEEE International Symposium on Industrial Electronics*. IEEE, 2011.

[18] S. Lynen, M. W. Achtelik, S. Weiss, M. Chli, and R. Siegwart, "A robust and modular multi-sensor fusion approach applied to mav navigation," in *2013 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2013.

[19] C. Chen, S. Rosa, Y. Miao, C. X. Lu, W. Wu, A. Markham, and N. Trigoni, "Selective sensor fusion for neural visual-inertial odometry," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[20] J. Fayyad, M. A. Jaradat, D. Gruyer, and H. Najjaran, "Deep learning sensor fusion for autonomous vehicle perception and localization: A review," *Sensors*, vol. 20, no. 15, 2020.

[21] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2015.

[22] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. PMLR, 2015.

[23] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. icml*, vol. 30, no. 1. Citeseer, 2013.

[24] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, 2014.

[25] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, 1997.

[26] C. M. Bishop, "Mixture density networks," 1994.

[27] 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," 2017.

[28] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[29] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010.

[30] M. Grupp, "evo: Python package for the evaluation of odometry and slam." https://github.com/MichaelGrupp/evo, 2017.

[31] T. Moore and D. Stouch, "A generalized extended kalman filter implementation for the robot operating system," in *Intelligent autonomous systems 13*. Springer, 2016.

[32] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Computer Architecture Letters*, vol. 13, no. 04, 1991.