DUNE: Deep UNcertainty Estimation for tracked visual features

Katia Katia Lillo, Andrea de Maio, Simon Lacroix, Amaury Nègre, Michèle Rombaut, Nicolas Marchand, Vercier Nicolas Vercier

DUNE: Deep UNcertainty Estimation for tracked visual features

Katia Sousa Lillo[1,2], Andrea De Maio[3], Simon Lacroix[3], Amaury Negre[1], Michèle Rombaut[1], Nicolas Marchand[1], and Nicolas Vercier[2]

***Abstract*—Uncertainty estimation of visual feature is essential for vision-based systems, such as visual navigation. We show that errors inherent to visual tracking, in particular using KLT tracker, can be learned using a probabilistic loss function to estimate the covariance matrix on each tracked feature position. The proposed system is trained and evaluated on synthetic data, as well as on real data, highlighting good results in comparison to the state of the art. The benefits of the tracking uncertainty estimates are illustrated for visual motion estimation.**

I. Introduction

Detecting and tracking visual point features in an image sequence is a key process for various vision-based applications such as object tracking or 3D reconstruction. It is also the basis of feature-based visual odometry [1], which, in its basic pipeline, consists of feature extraction, feature tracking, and motion estimation. Being able to precisely estimate the error of the feature tracking process allows to properly fuse visual and inertial data [2], and also to actively select the best features for motion estimation.

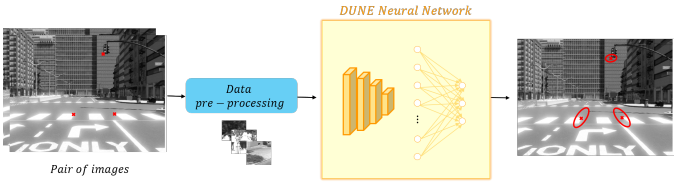


Fig. 1. Deep UNcertainty Estimation (DUNE) pipeline. The input is an image pair, on which point features are tracked. Pairs of image patches centered on the tracked points are processed by the DUNE network, which estimates the uncertainties on the position of every tracked features.

Related work: Only few works have been devoted to the estimation of the uncertainty of tracked points. In [3], the authors study the effect of covariance matrix computed on the image pixels on the precision of the localization of feature points. However, the work is not thoroughly validated and results are only shown on manually selected points. At the end, the authors provide a default precision value for every feature. In [4], the authors propose a linearised-based approach of the Kanade Lucas Tomasi (KLT) tracker [5] to estimate the uncertainty of the tracked points. The method is based on an incremental approximation, initialized with a fixed uncertainty, which bounds the uncertainty evolution. It is a derivative-based approach that is dependent on the KLT tracker.

As for visual motion estimation, most approaches that rely on tracked point features assume that the uncertainty on the image coordinates of the tracked features is constant [6]. This uncertainty is generally fixed through a preprocessing calibration [7] or a post-processing tuning [8]. In deep learning, most efforts are devoted to tackling the tracking problem at object level [9], [10]. Some approaches for point tracking and patch matching have also become publicly available [11], [12] but are rarely paired with uncertainty models of their predictions.

Approach and contribution: We advocate that the errors induced by tracking feature points are dependent on the local image signal that surrounds the points. Inspired by works that estimate the global uncertainty of visual motion estimation processes with a learning approach that processes the whole image [13] [14], we propose to estimate the uncertainty on the position of *each point* tracked between two image frames feeding a trained Convolutional Neural Network (CNN) with image patches centered on the points (Fig. 1). The benefit of using image patches is twofold, it allows for a shallow network with low inference time, and reduces the amount of data to process, matching the data used by the tracker itself. Our approach, named DUNE (Deep UNcertainty Estimation), is trained and evaluated on the SYNTHIA and KITTI datasets [15], [16], using feature points initially extracted with the Harris detector and tracked with KLT. Nonetheless, the proposed method can be applied to any combination of extracting and tracking or matching algorithms. By providing uncertainties of the tracked points in the form of a 2D covariance matrix, our longer term goal is to enable a fine estimate of the errors of a visual motion estimator, either coupled with INS data or not.

II. Learning tracking uncertainties

A. Input data

We aim at estimating a probabilistic uncertainty of the tracked interest points in the form of a 2D covariance matrix, which is not measurable with any device: only the tracking error can be observed. To estimate these errors,

[1] GIPSA-Lab, Université Grenoble Alpes, CNRS, Grenoble INP, 38000 Grenoble, France email: katia.lillo@grenoble-inp.fr
[2] Thales Avionics, THALES group, 25 rue Jules Védrines, 26000 Valence, France
[3] LAAS-CNRS, Université de Toulouse, CNRS 7, Avenue du Colonel Roche, 31031 Toulouse, France

we resort to SYNTHIA image sequences acquired with a camera under known motions, for which the dense depth map is known for each frame.

We opt for the Harris detector [17] to initialize the point features to be tracked, because of its low computational cost. A feature point ${}^{uv}\boldsymbol{m}_{k-1}$ of coordinates (u, v) detected at time $k-1$ is tracked by KLT as ${}^{uv}\tilde{\boldsymbol{m}}_k$ at time k. Knowing the camera calibration, its motion between times $k-1$ and k, and the 3D coordinates associated to ${}^{uv}\boldsymbol{m}_{k-1}$, one can predict the true image coordinates ${}^{uv}\boldsymbol{m}_k$ of the tracked point at time k. We define the tracking error ${}^{uv}\boldsymbol{\Delta}_k \in \mathbb{R}^2$ as:

$$ {}^{uv}\boldsymbol{\Delta}_k = {}^{uv}\boldsymbol{m}_k - {}^{uv}\tilde{\boldsymbol{m}}_k \tag{1}$$

Details are provided as supplementary material

B. Minimization problem

We define $\boldsymbol{\Sigma}_s \in \mathbb{R}^{2\times 2}$ the uncertainty related to the error ${}^{uv}\boldsymbol{\Delta}_s$ for a sample s. Assuming ${}^{uv}\boldsymbol{\Delta}_s \sim \mathcal{N}(0, \boldsymbol{\Sigma}_s)$, $\boldsymbol{\Sigma}_s$ is given by Eq. 2 where $(\sigma_u, \sigma_v) \in \mathbb{R}^2$ represents the variance in both direction u and v in $\boldsymbol{\mathcal{F}}_{pix}$:

$$ \boldsymbol{\Sigma}_s = \begin{pmatrix} \sigma_u^2 & \sigma_u \cdot \sigma_v \\ \sigma_u \cdot \sigma_v & \sigma_v^2 \end{pmatrix}_s \tag{2}$$

$\boldsymbol{\Sigma}_s$ is a symmetric matrix, and the uncertainty estimation problem comes to estimate the three parameters that compose it.

Let's consider the dataset $\mathcal{D} = \{\boldsymbol{W}_{i,k}, {}^{uv}\boldsymbol{\Delta}_k \,|\, \forall i \in |[0;1]| \,|\, \forall k \in |[1;s]|\}$ where s is the size of the dataset. The output prediction estimates the uncertainty $\boldsymbol{\Sigma}_k$ of the error ${}^{uv}\boldsymbol{\Delta}_k$ for all $k \in |[1;s]|$.

Under a zero-mean Gaussian noise model, the maximum likelihood is given by:

$$ \underset{\boldsymbol{\Sigma}_{1:s}\in\mathbb{R}^{2\times 2}}{\arg\max} \sum_{k=1}^{s} p({}^{uv}\boldsymbol{\Delta}_k | \boldsymbol{\Sigma}_k) \tag{3}$$

where $p({}^{uv}\boldsymbol{\Delta}_k | \boldsymbol{\Sigma}_k)$ is defined as:

$$ \begin{aligned} p({}^{uv}\boldsymbol{\Delta}_k | \boldsymbol{\Sigma}_k) = & \frac{1}{\sqrt{(2\pi)^{1/2} |\boldsymbol{\Sigma}_k|^{1/2}}} \cdot \\ & \exp\left[-\frac{1}{2} ({}^{uv}\boldsymbol{\Delta}_k)^T \, {\boldsymbol{\Sigma}_k}^{-1} ({}^{uv}\boldsymbol{\Delta}_k)\right] \end{aligned} \tag{4}$$

Maximizing the probability of an error under a Gaussian estimation is equivalent to minimizing the negative log-likelihood function [18]

$$ \underset{\boldsymbol{\Sigma}_{1:s}\in\mathbb{R}^{2\times 2}}{\arg\min} \sum_{k=1}^{s} -log(p({}^{uv}\boldsymbol{\Delta}_k | \boldsymbol{\Sigma}_k)) \tag{5}$$

$$ \sim \underset{\boldsymbol{\Sigma}_{1:s}\in\mathbb{R}^{2\times 2}}{\arg\min} \sum_{k=1}^{s} log(|\boldsymbol{\Sigma}_k|) + {}^{uv}\boldsymbol{\Delta}_k^T ({\boldsymbol{\Sigma}_k}^{-1}) {}^{uv}\boldsymbol{\Delta}_k \tag{6}$$

We apply an LDL decomposition to $\boldsymbol{\Sigma}$ where $\boldsymbol{D}$ is a diagonal matrix and $\boldsymbol{L}$ a unitriangular matrix, as described in [13]. The loss function is then defined by Eq. 7. An exponential function is applied to $\boldsymbol{D} \in \mathbb{R}^{2\times 2}$ to ensure $\boldsymbol{\Sigma}$ is positive semi-definite.

$$ \begin{aligned} L_{\boldsymbol{\Sigma}} = & \sum_{k=1}^{s} \sum_{i=1}^{2} \boldsymbol{d}_{i,k} \\ & + {}^{uv}\boldsymbol{\Delta}_k^T \left[(\boldsymbol{L}(\boldsymbol{l}_k) \boldsymbol{D}(\exp(\boldsymbol{d}_k)) \boldsymbol{L}(\boldsymbol{l}_k))^T)^{-1} \right] {}^{uv}\boldsymbol{\Delta}_k \end{aligned} \tag{7}$$

where $\boldsymbol{l}_k \in \mathbb{R}^s, \forall k \in |[1;s]|$ and $\boldsymbol{d}_{i,k} \in \mathbb{R}^{2\times s}, \forall i \in |[1;2]| \,|\, \forall k \in |[1;s]|$ are the output vectors predicted by the neural network model.

C. DUNE architecture

In view of the recent results of [13], we base our DUNE structure on a Convolutional Neural Network (CNN). We do not use max-pooling and choose to work at a window level to maximize information in the neighbourhood of a feature point.

DUNE takes two image patches centered on ${}^{uv}\tilde{\boldsymbol{m}}_k$ and ${}^{uv}\tilde{\boldsymbol{m}}_{k+1}$, and returns the uncertainty $\boldsymbol{\Sigma} \in \mathbb{R}^{2\times 2}$ on the image coordinates of ${}^{uv}\tilde{\boldsymbol{m}}_{k+1}$.

A sample $s \in \mathbb{N}$ is defined by a pair of stacked image patches $\boldsymbol{W}_k$ and $\boldsymbol{W}_{k+1}$ from two consecutive grayscale images $\boldsymbol{I}_k$ and $\boldsymbol{I}_{k+1}$, centered on ${}^{uv}\tilde{\boldsymbol{m}}_k$ and ${}^{uv}\tilde{\boldsymbol{m}}_{k+1}$. A popular KLT setting considers a window analysis of size 21×21: we set the size of $\boldsymbol{W}_k$ and $\boldsymbol{W}_{k+1}$ to 21×21, so the input size data is $21 \times 21 \times 2$.

DUNE is a deep CNN composed of two main blocks: the deep convolutional structure, detailed in Fig. 2, and the dense layers. The convolutional structure is a succession of 4 convolutional layers with fixed kernel size of 3×3. A batch normalization layer and dropout set at 20% are added after each convolutional layer. The batch normalization layer is used for training stability, whereas the dropout avoids overfitting.

The convolutional layers are followed by two dense layers with respectively 256 and 3 neurons. Each layer, with the exception of the last one, is followed by a *leakyrelu* activation function.

We observed a correlation between the Harris point response (λ_1, λ_2), defined by the eigenvalues of the Hessian matrix according to [17], and the number of frames during which a point is successfully tracked in time, suggesting these values are representative of the tracking precision. We defined a variation DUNE architecture that incorporates them: we concatenate them in the late stage of the network as described in Fig. 2.

III. Results and evaluation

A. Datasets and training

We firstly evaluate DUNE on the SYNTHIA dataset [15], drawing from the latest release SYNTHIA-AL [19]. We create our data set with a total of 56508 input samples from 12 different scenarios part of the SYNTHIA-AL-Train package. Each scenario is urban or semi urban and includes various visual perturbations such as rain or overexposure. The video sequences are 250 to 750 images

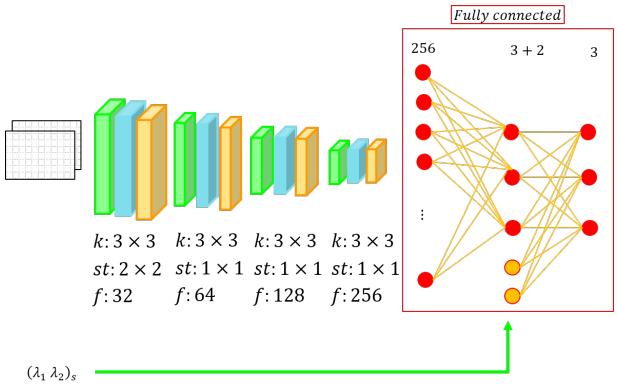


Fig. 2. DUNE architecture integrating the Harris point response (λ_1, λ_2) as an input information. Each layer is a sequence of convolution (green), batch normalization (blue) and dropout (yellow). Convolution parameters, i.e. kernel size k, stride st and number of filters f, are included.

long. A maximum of 1000 points per image are detected by the Harris detector, points corresponding to dynamic objects are deleted. Finally, gross tracking errors (above the patch size) are removed, so as not to pollute the training dataset – such errors could easily be removed, e.g. thanks to a RANSAC process. 70% of the data is used to train the NN while 15% is kept for testing and 15% for evaluation. We train for 300 epochs with a batch size of 64. We select Nadam ($lr = 1 \cdot 10^{-4}$) as optimizer with its parameters set to $\beta_1 = 0.9$ and $\beta_2 = 0.99$. The final model is selected as the one minimizing the loss for the test set. Such evaluation is carried out after each epoch.

B. Evaluation metric

The evaluation is assessed between the ground truth error ${}^{uv}\boldsymbol{\Delta}_k$ and the predicted uncertainty. We select two metrics:

$$NNE = \frac{1}{N}\sum_{k=1}^{d}\sqrt{\frac{||\boldsymbol{\Delta}_k||_2^2}{tr(\boldsymbol{\Sigma}_k)}}$$
$$MD = \frac{1}{N}\sum_{k=1}^{d}\sqrt{\frac{\boldsymbol{\Delta}_k^T\,\boldsymbol{\Sigma}_k{}^{-1}\,\boldsymbol{\Delta}_k}{dim(\boldsymbol{\Delta}_k)}} \quad (8)$$

The first metric is the *Normalized Norm Error* (NNE), where $||\boldsymbol{\Delta}_k||_2^2$ is the squared norm of the error and $tr(\boldsymbol{\Sigma}_k)$ defines the trace of the uncertainty. NNE gives information on the precision of the predicted variances, but does not consider off-diagonal covariances. The second metric is the Mahalanobis Distance (MD) that measures the distance between DUNE estimations and the errors ${}^{uv}\boldsymbol{\Delta}_k$ also accounting for the covariance. For both metrics, the optimal uncertainty model is obtained at values equal to 1. Values below qualify a pessimistic uncertainty estimation, and above an optimistic estimation model.

Moreover, we assume the error ${}^{uv}\boldsymbol{\Delta}_s \sim \mathcal{N}(0, \boldsymbol{\Sigma}_s)$ to be Gaussian, thus one can verify the following inequality (Eq. 9) as an indicator of the dynamic of the estimation model.

$$-n\sigma_{\{u,v\}} \geq {}^{uv}\boldsymbol{\Delta}_{\{u,v\}} \geq +n\sigma_{\{u,v\}} \quad (9)$$

where σ defines the standard deviation of the predicted model, n is the number of standard deviation and ${}^{uv}\boldsymbol{\Delta}$ the observed errors. For a normal distribution, the quantiles for $n = 1$, 2, and 3 should respectively be 68%, 95% and 99.7% of the observed errors ${}^{uv}\boldsymbol{\Delta}$.

C. Impact of the Harris response

Table I compares the results of DUNE and DUNE with the Harris response. Values shown are obtained averaging 5 independent learning sessions.

mean \ Method	DUNE	DUNE + (λ_1, λ_2)
3σ	98.04	96.98
2σ	93.75	91.28
1σ	76.23	69.93
MD	0.83	**0.99**
NNE	0.77	**0.89**

TABLE I
COMPARISON OF THE IMPACT OF HARRIS PARAMETERS

The overall mean evaluation of DUNE without the Harris parameters tends to be pessimistic and highlights the impact of the Harris parameters on DUNE estimations that leads to better results.

D. Evaluation with the SYNTHIA dataset

In this section, we present the uncertainty estimates of two selected scenarios, one with a rotational motion of the camera (Fig. 3(a), 726 tracked points), and the other with a smooth rectilinear motion (Fig. 3(c), 949 tracked points). Fig. 3(b) and Fig. 3(d) compare the variance estimates with the ground truth tracking errors. It is interesting to note that the variance estimates are very different between u and v for the rotational motion (Fig. 3(b)), and that for the small forward motion (Fig. 3(d)) the predicted estimates are smaller. In both cases, the variance predictions match well the ground truth errors – see for instance the close-up view around keypoint 280 in Fig. 3(c).

DUNE demonstrates the ability to adapt the variance predictions to the tracking errors, independently from the type of motion. The estimates fits well the error ground truth, indicating that the system is able to correctly capture the uncertainty under the Gaussian assumption.

Evaluation: We compare our results to [4] based on incremental uncertainty estimation with a fixed initial value set at 0.5 pixel, and to a Fixed Covariance (Fix-C) equal to 0.5 pixels in both image directions for all points – the tracking error model mostly considered in the visual motion estimation literature. The results are presented in table II.

One can notice that both [4] and Fix-C method produce very pessimistic estimations. Indeed, $MD < 0.7$ for [4] and $MD < 1/2$ for Fix-C. These results are also visible at 1σ where Fix-C covers $\sim$ 96.6% of the error and [4] $\sim$ 86.1%. The quantiles are only indicators of the dynamic of the predicted model, but do not intend to evaluate

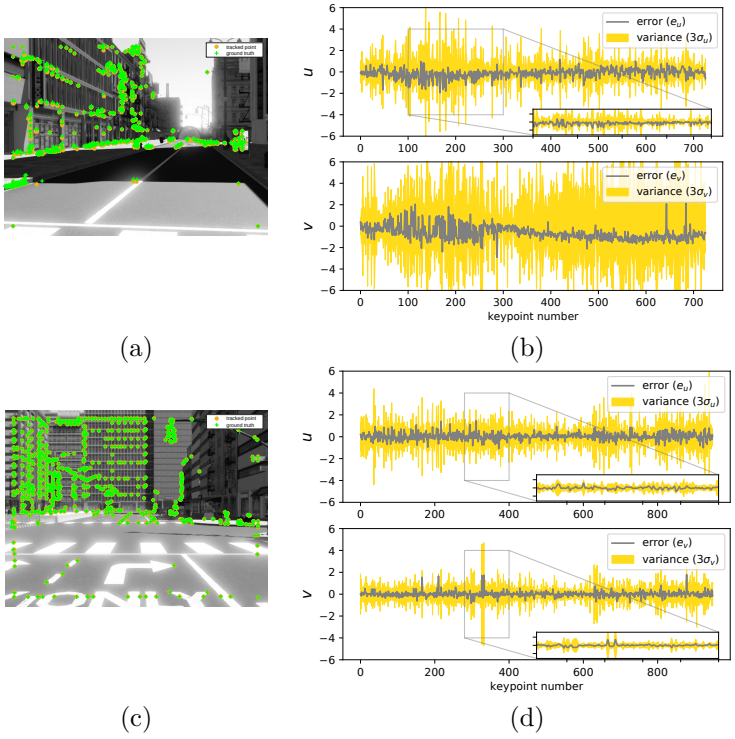


Fig. 3. Evolution of the predicted uncertainty - (a) Scene 1: Image t_{k+1}; (b) Predicted uncertainty with a rotational motion ; (c) Scene 2: Image t_{k+1}; (d) Predicted uncertainty with a forward motion

Metric	DUNE + λ_i	[4]	Fix-C
$3\sigma_u$	97.08	97.71	99.95
$3\sigma_v$	95.75	98.84	99.91
$2\sigma_u$	91.28	95.21	99.36
$2\sigma_v$	90.03	96.80	99.69
$1\sigma_u$	71.27	84.46	96.02
$1\sigma_v$	70.84	87.74	97.21
MD	**1.02**	0.68	0.32
NNE	**0.8**	0.58	0.3

TABLE II
COMPARISON OF DUNE PREDICTION WITH REGARD TO STATE OF THE ART

the gaussianity of a model. With $MD = 1.02$, DUNE + (λ_1, λ_2) outperforms the two approaches by a significant margin.

This analysis is highlighted by Fig. 4, representing the uncertainty predicted by DUNE + (λ_1, λ_2) with regard to [4] and Fix-C. It appears that DUNE presents a very good dynamic and fits the best the error model compared to [4] and Fix-C. In particular, at keypoints 180 and 380 where the error is very small.

Impact of DUNE on motion estimation: Finally, we propose an indirect validation to assess the relevance of our work through a straightforward motion estimation. For each image, the tracked points are split in two sets: one for which $\sqrt{tr(\boldsymbol{\Sigma}_k)} \leq 0.5$ (the *best points*), and one for which $\sqrt{tr(\boldsymbol{\Sigma}_k)} > 0.5$ (the *worst points*). The camera motion is estimated for both sets with EPnP [20], using the 3D position of the tracked features at time k provided by SYNTHIA (by no means this process is an approach for pose reconstruction, neither for outliers rejection: the goal is to assess the possibilities brought by fine variance prediction on motion estimation).

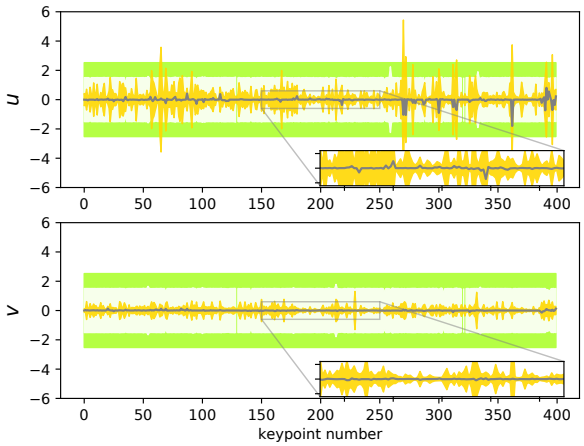


Fig. 4. Comparison of DUNE predictions to the state of the art - The gray line represents the error in direction u and v on frames with smooth motion, ▭ describes the Fix-C at 3σ, ▭ refers to [4] at 3σ and ▭ refers to DUNE uncertainty predictions at 3σ.

Using the ground truth camera pose for each frame provided by SYNTHIA, we compute for both sets the error between the true and the estimated motion as:

$$ {}_{k+1}T_{err} = {}_{k+1}T_{groundtruth} \cdot {}_{k+1}T^{-1}_{estimated} \tag{10} $$

where ${}_{k+1}T_{err}$ defines the rigid transformation error at time $k+1$. The results are presented on Fig. 5. All our results are compared to a motion estimation integrating RANSAC [21], with the same threshold. Fig. 5(a) shows that bigger uncertainties tends to produce more error in translation with a mean translation error of 0.67m whereas with the best points it is on average 0.2m. This is even more visible on Fig. 5(b), where the average error for best and worst points are respectively $2 \cdot 10^{-3}$ deg and $8 \cdot 10^{-3}$ deg. Note that estimations produced by applying RANSAC on EPnP lead to slightly smaller error in both cases, which is expected.

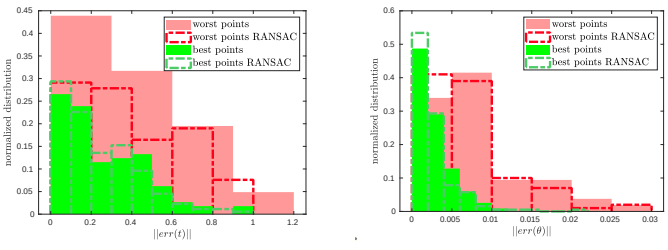


Fig. 5. Impact on the motion estimation - (a) Error on the translation (in m); (b) Error on the rotation (in deg)

E. Evaluation with the KITTI dataset

In order to validate our approach in real world scenarios, we also test DUNE on KITTI raw data [16] using

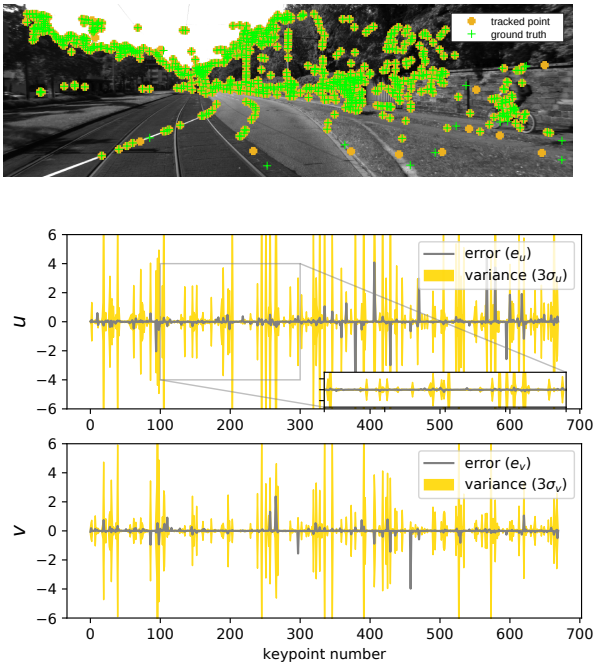


Fig. 6. Image $t_{k|k+1}$ on top; Evolution of the predicted uncertainty compared to the tracking error

sequences `2011_09_26_0005`, `2011_09_29_0013` and `2011_09_26_0026` for training and `2011_09_26_0002` for evaluation. Since the exact 3D structure of the world (e.g. in form of a depth map), which would be necessary to precisely retrieve a tracked point at any time, is not available, the ground truth is given by computing the Harris detector at time t followed by the tracker KLT at time $t+1$, and applying KLT backward at time t. Ideally, the point should return to its initial position. We define the error as the distance between both position ${}^{uv}_{Harris}\boldsymbol{m}_k - {}^{uv}_{KLT}\boldsymbol{m}_{k|k+1}$. Similarly to the training on SYNTHIA, we train for 500 epochs using batches of 64.

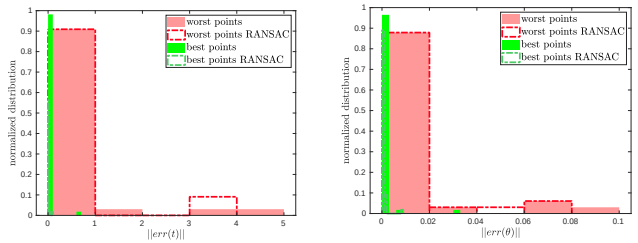
The evaluated sequence is illustrated on Fig.6 where 670 keypoints are detected and tracked. DUNE produces uncertainties that fit the error model highlighting a very good dynamic (Fig.6).

Besides, Table III compares DUNE trained with KITTI to the state of the art. The results obtained with [4] and Fix-C, set at 0.5^2 pixels as the ground truth is defined by the cumulated error over two timesteps, are very pessimistic estimations with $MD = 0.16$ and $MD = 0.3$ respectively. DUNE model outperforms both method on real data, with $MD = 1.39$.

Impact of DUNE on motion estimation: Similarly to section III-D, we apply a geometric validation where the ground truth is defines by the identity matrix ${}_{k+1}T_{groundtruth} = \begin{bmatrix}\mathbb{I}_{3\times3} & \mathbb{0}_{3\times1}\end{bmatrix}$. Indeed, the forward-backward motion provides a known and theoretically perfect motion. The predicted $3D$-transformation is evaluated using stereovision during the initial Harris detection allowing the triangulation of the $2D$ points into $3D$.

Metric	DUNE	[4]	Fix-C
$3\sigma_u$	94.97	99.53	98.92
$3\sigma_v$	94.83	99.73	99.22
$2\sigma_u$	91.08	99.34	98.59
$2\sigma_v$	91.05	99.55	98.93
$1\sigma_u$	79.34	99.86	97.44
$1\sigma_v$	79.27	98.17	98.18
MD	**1.39**	0.17	0.3
NNE	**1.39**	0.10	0.24

TABLE III
COMPARISON OF DUNE PREDICTION WITH REGARD TO STATE OF THE ART



(a) (b)

Fig. 7. Impact on the motion estimation - (a) Error on the translation (in m); (b) Error on the rotation (in deg)

Results are illustrated Fig. 7. The mean translation error is 0.0128m for the best selected points, and 0.3310m for the worst ones(Fig. 7(a)). Regarding the error on the rotation, the best selected keypoints produce a mean error of $6 \cdot 10^{-4}$ where the worst points achieve $8 \cdot 10^{-3}$. In both cases, a factor 10 differentiate the two collections.

One may notice the difference between SYNTHIA (Fig. 5) and KITTI (Fig. 7) results. In fact, KITTI highlights a factor 10 improvement on the motion estimation between best and worst point selection, whereas SYNTHIA shows a progression factor of approximately 3 to 4. The strength of SYNTHIA dataset is the direct availability of the ground truth given by the depth maps. However, the synthetic data produces unwanted noises such as aliasing and sampling effects on the depth maps impacting the ground truth. On the other hand, KITTI approach provides a perfect ground truth by applying a forward-backward motion. SYNTHIA came in handy to proof and validate our model, when in fact KITTI led to even better results with a real data application.

IV. Conclusion and future work

We presented a CNN-based method that produces estimates of the uncertainty on the position of tracked feature points, and we have shown that it is possible to capture the dynamic of the observed error model. DUNE predictive model has been validated by two dedicated metrics, and a motion estimation scheme shows that the estimated uncertainties can help to select the best features for motion estimation. The next step is to integrate these estimated uncertainties into a INS-visual motion scheme: the uncertainties could first help to select the best points

for motion estimation (among other known criteria, such as a balanced spread of the points on the whole image), and be used to estimate a precise uncertainty for the computed motions.

Further improvements on DUNE architecture can also be considered, such as the combination of DUNE with a recurrent network, exploiting the whole tracking history through a sequence to provide more precise uncertainties.

References

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

[2] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 3565–3572, 2007.

[3] Y. Kanazawa and K. Kanatani, "Do we really have to consider covariance matrices for image features?," in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, vol. 2, pp. 301–306 vol.2, 2001.

[4] X. I. Wong and M. Majji, "Uncertainty quantification of lucas kanade feature track and application to visual odometry," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 950–958, 2017.

[5] D. L. Bruce and T. Kanade, "An iterative image registration technique with an application to stereo VIsion," 1981.

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

[7] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[8] G. Huang, "Visual-inertial navigation: A concise review," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 9572–9582, 2019.

[9] Y. Cui, C. Jiang, L. Wang, and G. Wu, "Mixformer: End-to-end tracking with iterative mixed attention," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13608–13618, 2022.

[10] B. Yan, Y. Jiang, P. Sun, D. Wang, Z. Yuan, P. Luo, and H. Lu, "Towards grand unification of object tracking," in *ECCV*, 2022.

[11] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg, "Matchnet: Unifying feature and metric learning for patch-based matching," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3279–3286, 2015.

[12] M. Parchami and S. I. Sayed, "Deep feature tracker: A novel application for deep convolutional neural networks," *arXiv preprint arXiv:2108.00105*, 2021.

[13] K. Liu, K. Ok, W. Vega-Brown, and N. Roy, "Deep inference for covariance estimation: Learning gaussian noise models for state estimation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1436–1443, 2018.

[14] A. De Maio and S. Lacroix, "Simultaneously learning corrections and error models for geometry-based visual odometry methods," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6536–6543, 2020.

[15] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, "The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[16] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research (IJRR)*, 2013.

[17] C. Harris and M. Stephens, "A combined corner and edge detector," in *Procedings of the Alvey Vision Conference 1988*, 1988.

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

[19] J. Zolfaghari Bengar, A. Gonzalez-Garcia, G. Villalonga, B. Raducanu, H. H. Aghdam, M. Mozerov, A. M. Lopez, and J. van de Weijer, "Temporal coherence for active learning in videos," in *The IEEE International Conference in Computer Vision, Workshops (ICCV Workshops)*, 2019.

[20] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o(n) solution to the pnp problem," *International Journal of Computer Vision*, vol. 81, 02 2009.

[21] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," vol. 24, no. 6, 1981.