Proceeding of the IEEE
International Conference on Robotics and Biomimetics
Dali, China, December 2019

A Monocular Visual Odometry Combining Edge Enhance with Deep Learning

Mingsu Yan and Chaoxia Shi
School of Computer Science and Engineering
Nanjing University of Science and Technology
Nanjing, Jiangsu Province, China
{ yanmingsu & scx}@njust.edu.cn

Yanqing Wang
School of Information Engineering
Nanjing Xiaozhuang University
Nanjing, Jiangsu Province, China
wyq0325@126.com

***Abstract* - Deep learning-based monocular visual odometry (VO) methods have recently shown highly promising results. However, most of these methods do not produce reliable results because of the insufficient features when the environment is low-textured and ignore the importance of features of images. This paper proposes a monocular VO method with edge feature extraction and deep Recurrent-Convolutional Neural Network (RCNN). This solution embedded traditional geometry algorithm in deep learning-based VO, in order to enhance the influence of edge feature information in images. Our method can have good performance in low-textured environments.**

Index Terms – visual odometry, deep learning, edge detection, RCNN.

I. INTRODUCTION

Visual odometry is a method which estimates the ego-motion according to input images, which is a core module of simultaneous localization and mapping (SLAM) system. For the reason that VO can determine the current position according to the feed from cameras, it has been a hot research field in computer vision, and has many applications in the fields of autonomous driving, robotics, and augmented reality. In recent years, visual odometry with stereo cameras has achieved great progress. And it's widely applied for its ability of reliable depth map estimation. However, once the distance between camera and baseline is not the same scale as the scene, it will degenerate into a monocular one. Moreover, great engineering efforts are needed for coding some state-of-the-art methods, designing and fine-tuning each module in the pipeline to ensure great performance both in terms of accuracy and robustness.

Different from stereo visual odometry, monocular VO cannot get environment map and robot motion in real scale, so it has to estimate an absolute scale with the help of prior knowledge or other information such as the camera's height, which makes it easier to have big drift and more challenging than the stereo VO. As a consequence, scale drift is a burning issue in monocular odometry, which needs to be eliminated by recovering the absolute scale.

These years, Deep Learning (DL) has been applied in many computer vision tasks and achieved great progress for its powerful ability of feature learning. Deep learning can accelerate the development of visual odometry [1]. Recognizing the promising performance of the idea, Wang et al. [2] presented a recurrent Convolutional Network architecture to learn monocular VO from video sequences. However, [2] suffers from dynamically moving objects and lacking features in the open area around the highway.

In this paper, we propose an idea paying more attention to the edge features, which provide structural information. Our work integrates the geometry algorithm—edge enhancement algorithm extended from the Canny edge detection algorithm [3] with DeepVO [2]. Through this method, we can have a monocular visual odometry which has stronger robustness in different scenarios and is able to perform more accurate feature extraction.

The rest of this paper is organized as follows. Section II reviews related work. The method description is given in Section III, and Section IV shows the results of the experiment. Then we draw the conclusion in Section V.

II. RELATED WORK

According to the technique and framework adopted, algorithms can be mainly divided into two categories: methods based on geometry and based on learning. In this section, various DL based methods on monocular VO are discussed. Additionally, some edge-based monocular VO are reviewed.

A. Deep Learning-Based Methods on Monocular VO

In recent years, great progress has been made in the development of monocular VO methods. As far as we know, the synchrony detection between image sequences and features was first proposed by [4]， which provides a workable plan for stereo VO based on deep learning. However, in essence, it treats the VO as a classification problem instead of the pose regression. [5] fine-tunes the images of a specific scene by using CNN, to solve the Camera relocalization with a single image. Its suggest to label these images by Structure from Motion (SfM) consumes more time and labor intensive in large-scale scenarios. To overcome this issue, Costante et al. [6] firstly used CNNs for ego-motion estimation from the dense optical flow obtained through image feature matching. FlowOdometry [7] used FlowNet [8] to extract features combining with CNNs to do the regression, and it's the first end-to-end method of VO learning. LS-VO [9] uses an auto-encoder network to represent the optical flow in a non-linear form manifold to estimate the ego-motion. Compared with the monocular VO based on model, these studies use a non-linear CNN to extract features and estimate motion, and accumulate the relative poses regressed directly as the global trajectory. Agrawal et al. [10] proposed an algorithm to learn visual features from ego-motion estimating which can estimate relative camera poses. Ummenhofer et al. [11] proposed an

978-1-7281-6321-5/19//$31.00 ©2019 IEEE

end-to-end visual odometry and the network to estimate depth through treating SfM as a supervised learning issue. Wang et al. [2] proposed a Recurrent Convolutional Network architecture DeepVO to learn monocular odometry from video sequences. DeepVO inputs FlowNet features into the Long Short-Term Memory (LSTM) as the sequence-to-sequence encoder-decoder for monocular visual odometry learning. In order to improve the performance, Clark et. al [12] integrated extra IMU readings into the same architecture. Additionally, GCN [13] made use of an RCNN architecture to generate the corresponding key-points' descriptors. These studies utilize CNNs to learn the image interpretation and make state estimation with the sequential prediction ability of RNNs.

Additionally, several studies attempted to realize global relocalization and monocular VO concurrently by combining the global and relative pose regression with the help of a shared neural network. The semantic based method VLocNet [14] and its advanced version VLocNet++ [15] can achieve 6DOF (Degree of Freedom) global pose regression and VO evaluation according to consecutive monocular images. In addition, MapNet [16] globally localizes the camera by learning a data-driven map representation, while eliminating the drift produced by estimating relative pose through geometric constraint of two consecutive images. However, this method is only capable in a known environment, and working local geometry constraint, relative pose estimation has limited effect. Thus when it comes to unknown scenario, this kind of monocular VO method cannot be applied.

With the help of CNNs and RNNs, supervised DL based monocular VO can achieve more accurate estimation of absolute scale. In the training of the deep neural network, we need more ground truth data than traditional VO. However, if comparing with other learning tasks, it's much cheaper to acquire training data, because we can obtain data directly by GPS-RTK or Vicon cameras instead of manual annotations.

B. Edge-Based Monocular VO

Edges carry important information of an image. Known as part of human vision, artificial detectors for edge recognition can achieve structural information extraction from complex images. Tarrio and Pedre [17] proposed a visual odometry pipeline based on edge which estimates depth with the use of edge features. However, the estimation is erroneous for the reason that odometry only works on consistent pairwise. For the accuracy of camera estimation, global consistency checking is of great importance in long trajectories.

Therefore, we try to design a method combining edge feature extraction with deep RCNN network in monocular VO for a good performance in low-textured environments such as large open areas around highway.

III. OUR METHOD

Our method combines the edge enhancement operation based on the edge detection algorithm Canny with a deep RCNN framework. The architecture of the proposed method is shown in Fig.1.

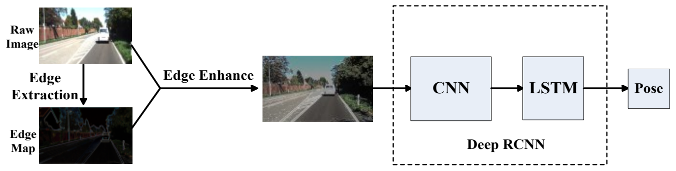


Fig. 1 The architecture of the proposed method.

A. Edge Enhance Algorithm

The edge enhance step is to extract the structural information from edge. The main output of this step is a combination of edge maps and original RGB images, which is equivalent to edge enhancement. And the choice of different edge detectors may have different impact on the final performance. In our system, robustness of repetitivity is important in terms of tracking edges in consecutive frames. Our proposed algorithm benefits from the fact that edge extraction is a highly robust, highly parallelizable and widely studied process. And as we all know, the Canny Edge Detector [9] is a classical and robust method based on maxima search of the gradient which detects a wide range of edges in images with the multi-staged algorithm. Inspired by it, we utilize part steps of this algorithm and then add some other operations to achieve edge enhancement.

The whole process is formulated as follows: We begin with smoothing of the RGB images to reduce noise. We calculate the first order derivatives in each direction as the gradient of an image, so that the edges seem to be more and thicker. We utilize central difference to compute the gradient of R, G, B three channels separately in order to prepare the images for edge detection. And the blur filtering of Gaussian is used to smooth the image. And we remove high frequency noises of the image by using the Gaussian kernel filter. The generating equation of (2k + 1) by (2k + 1) Gauss filter kernels is given by the following formula:

$$H_{ij} = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{(i-(k+1))^2+(j-(k+1))^2}{2\sigma^2}\right), \quad 1 \le i, j \le (2k+1) \, . \tag{1}$$

The kernel size 2k+1 is chosen according to the blurring effect we expect. Basically, if the kernel is smaller, the blur will be less visible. In our experiment, we use a filter with the 5 by 5 kernel size. And the brightness value of each pixel e can be calculated like this:

$$e = sum(H * A) \, . \tag{2}$$

where * represents the convolution operation, A is a window centered on pixel e, and the sum function means the sum of all elements in the matrix.

The second step is calculating the gradient. We use edge detection operators to calculate the image's gradient and detect the intensity and direction of detected edges. Edges correspond to a change of pixels' intensity. For the purpose of detecting edges, we apply filters to highlight the pixels' intensity change which is corresponded to edges in two directions—horizontal (x) and vertical (y) direction. After the image smoothing operation, the derivatives I_x and I_y w.r.t. x and y are calculated. We implement this step by convolving I in each channel i.e. I_R , I_G , I_B with Sobel kernels S_x and S_y respectively, and each channel's magnitude G and slope θ of the gradient can be calculated by the follow formula (taking G_R as an example):

$$G_R = \sqrt{{G_{R_x}}^2 + {G_{R_y}}^2} = \sqrt{(sum(S_x * A))^2 + (sum(S_y * A))^2} \,. \quad (3)$$

$$\theta_R = \arctan(G_{R_y} / G_{R_x}) \,. \quad (4)$$

Where G_{R_x} , G_{R_y} represents the magnitude in horizontal and vertical direction of the R channel respectively. Then, we can have an RGB edge map which is almost the expected one except that some edges are thick while some are thin. Therefore, we perform the non-maximum suppression in next step to make the edges thin. In the non-maximum suppression step, we need to figure out whether the point is a local maximum of the interpolated gradient magnitude in its direction. This step has a significant effect on the performance of edges. we compare the pixel with the ones next to it, if the pixel is larger we do not change it, otherwise, we set zero to the pixel, and we can get an RGB edge map of images as a result.

At the final step, we sum the pixel value of the original RGB image and the RGB edge map in proportion. Then we can gain an edge enhancement result of the original images which is the input of the following deep neural network. The calculation method of each channel is as follows (taking the R channel as an example):

$$I_R = (w_{raw} p_{raw} + w_{edge} p_{edge}) * \frac{255}{p_{\max}} \,. \quad (5)$$

Where w_{raw}, w_{edge} represent the weight of the raw image and the edge map of the R channel respectively, p_{raw} and p_{edge} are the pixel value of the R channel of the raw image and the edge map. $p_{\max}$ is the maximum pixel value of each image after the previous summation.

B. The RCNN Architecture

In order to solve the VO and other geometric issues, it's important for a framework to learn geometric feature representation. Meanwhile, because the VO system develops with time and operate on image sequences obtained in the process of motion, it is necessary to derive connections among consecutive image frames, such as motion models. Therefore, a deep RCNN architecture is utilized in our paper, which is shown in Fig 1. The RCNN based architecture which combines CNN and RNN has the advantage of allowing VO to extract features and sequential model at the same time.

The network takes the monocular edge enhancement image sequence as input. We subtract the mean RGB values of the training set at each time step, and resize it in the multiple of 64 to preprocess the RGB image frame. Then two consecutive images are stacked together as a tensor for the deep RCNN for the purpose of learning the way of motion information extraction and pose estimation. In detail, effective features are produced for monocular VO by feeding image tensor into a CNN, which are then passed into an RNN for sequential learning. The image pair at each time step generates a pose estimate through the network, and as time goes on the VO system develops. Once the images are captured, new poses will be estimated.

In our network, we define the loss consisting of Mean Square Error (MSE) of all positions p and orientations of the sequences:

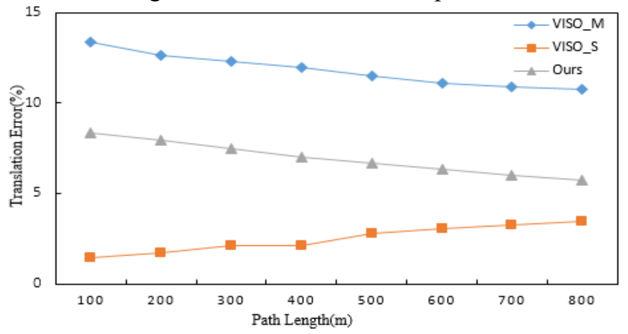
$$L^* = \arg\min_{L} \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \left\| \hat{p}_t - p_t \right\|_2^2 + \lambda \left\| \hat{\varphi}_t - \varphi_t \right\|_2^2 \,. \quad (6)$$

where (p_t, φ_t) and $(\hat{p}_t, \hat{\varphi}_t)$ are the truth pose and the estimated one at the time t respectively, $\|*\|$ represents a 2-norm computation, N is the number of the samples, and λ is a designed parameter to balance the scale weight of positions and orientations.

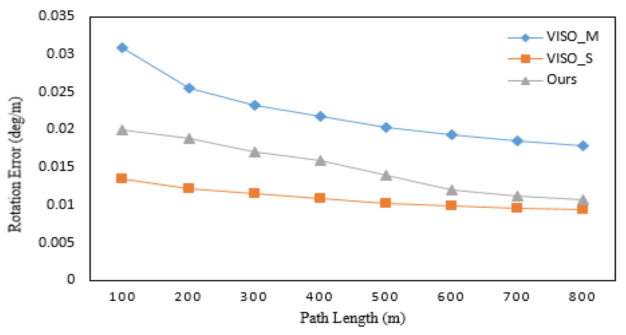
IV. EXPERIMENTS

In this section, we evaluate the proposed approach on the well-known KITTI dataset. The KITTI dataset has 22 sequences of images, and Sequence 00-10 provides the extra information of ground truth. And we conduct the experiment to evaluate the proposed method based on the first 11 sequences for quantitative performance analysis. The sequence 00, 01, 02, 06, 08 and 10 are separated for our model training. The rest sequence 04, 05, 07 and 09 are the test dataset which are used to evaluate the performance of our model. The network is implemented based on the PyTorch framework.

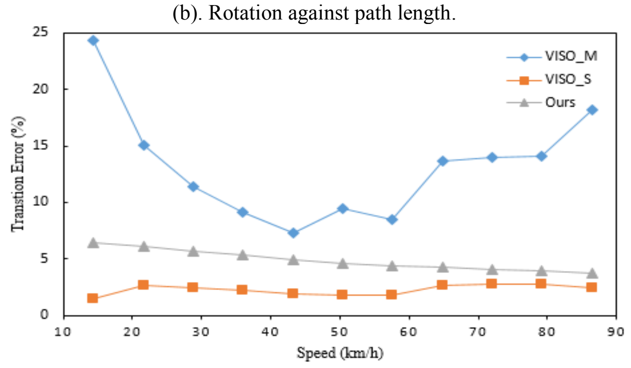
The performance of our model is also analyzed in the evaluation metrics of KITTI VO/SLAM, i.e., average Root Mean Square Errors(RMSEs) of the translational and rotational errors in the range of 100 to 800 meter lengths and different speeds. As shown in Figure 2 apparently, our method is much better than VISO_M, and a little worse than VISO_S. It can perform even more stable than VISO_S when the speed is higher than 60 km/h in Fig. 2(d). A visual comparison of reconstructed trajectories is shown in Figure 3. It shows that our method can get a better result than DeepVO.



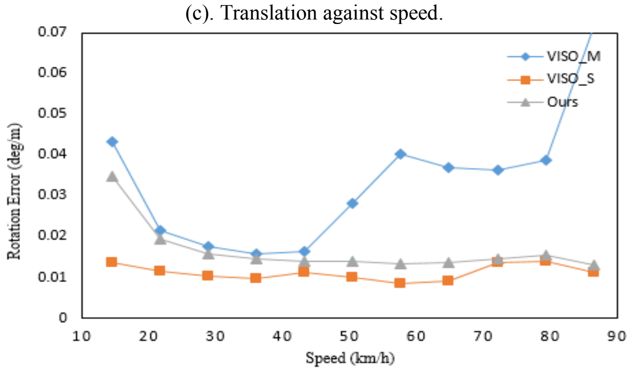
(a). Translation against path length.



(b). Rotation against path length.

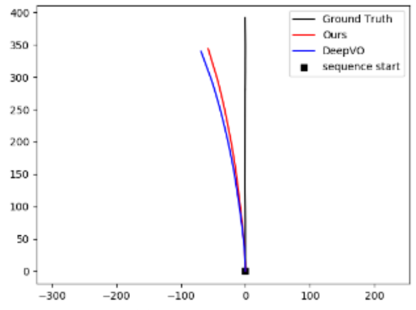


(c). Translation against speed.

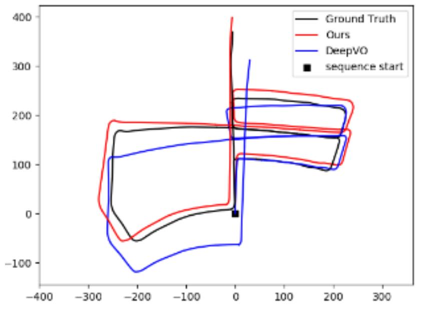


(d). Rotation against speed.

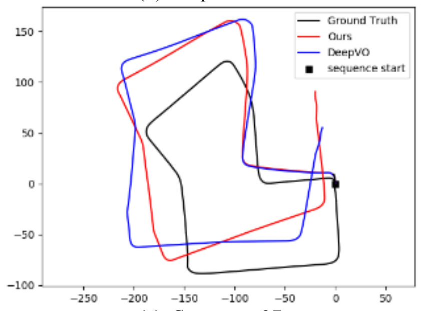
Fig. 2 The average RMSEs on translation and rotation against different lengths and speeds.



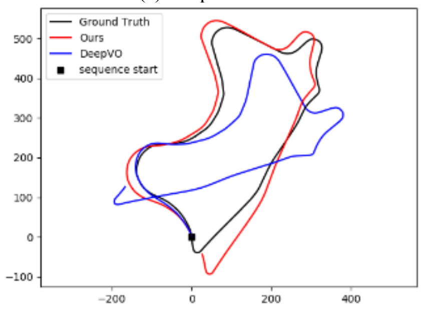
(a). Sequence 04.



(b). Sequence 05.



(c). Sequence 07.



(d). Sequence 09.

Fig. 3 Reconstructed trajectories of testing results on sequence 04, 05, 07 and 09.

V. CONCLUSION

This paper proposes a method integrating traditional geometry based edge enhance algorithm with an RCNN architecture. Leveraging the advantage of edge enhance algorithm, we can extract more features from the input images, making our model perform better in open areas and low-textured environment. The experiment conducted has shown that our method can produce more accurate results on monocular visual odometry.

ACKNOWLEDGMENT

This paper was supported by Chinese National Natural Science Foundation (granted No.61371040).

REFERENCES

[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. D. Reid, and J. J. Leonard. Past, present, and future of simultaneous localization and mapping: Towards the robust-perception age. 2016.

[2] S. Wang, R. Clark, H. Wen, and N. Trigoni. Deepvo: Towards end-to-end visual dometry with deep recurrent convolutional neural networks. In Robotics and Automation (ICRA), 2017 IEEE International Conference on, pages 2043–2050. IEEE, 2017.

[3] J. Canny, "A Computational Approach to Edge Detection," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 8, pp. 679-698, 1986.

[4] K. Konda and R. Memisevic, "Learning visual odometry with a convolutional network," in Proceedings of International Conference on Computer Vision Theory and Applications, 2015.

[5] A. Kendall, M. Grimes, and R. Cipolla, "Convolutional networks for real-time 6-DoF camera relocalization," in Proceedings of International Conference on Computer Vision (ICCV), 2015.

[6] G. Costante, M. Mancini, P. Valigi, T. A. Ciarfuglia, Exploring representation learning with CNNs for frame-to-frame ego-motion estimation, IEEE robotics and automation letters 1 (1) (2016) 18–25 (2016).

[7] P. Muller, A. Savakis, Flowdometry: An optical flow and deep learning based approach to visual odometry, in: IEEE Winter Conference on Applications of Computer Vision, 2017, pp. 624–631 (2017).

[8] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, T. Brox, FlowNet: Learning optical flow with convolutional networks, in: IEEE International Conference on Computer Vision, 2015, pp. 2758–2766 (2015).

[9] G. Costante, T. A. Ciarfuglia, LS-VO: Learning dense optical subspace for robust visual odometry estimation, IEEE Robotics and Automation Letters 3 (3) (2018) 1735–1742 (2018).

[10] P. Agrawal, J. Carreira, and J. Malik. Learning to see by moving. In Proceedings of the IEEE International Conference on Computer Vision, pages 37–45, 2015.

[11] B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy, and T. Brox. Demon: Depth and motion network for learning monocular stereo. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[12] R. Clark, S. Wang, H. Wen, A. Markham, N. Trigoni, VINet: Visual-inertial odometry as a sequence-to-sequence learning problem., in: The AAAI Conference on Artificial Intelligence, 2017, pp. 3995–4001 (2017).

[13] J. Tang, J. Folkesson, P. Jensfelt, Geometric correspondence network for camera motion estimation, IEEE Robotics and Automation Letters 3 (2) (2018) 1010–1017 (2018).

[14] A. Valada, N. Radwan, W. Burgard, Deep auxiliary learning for visual localization and odometry, in: IEEE International Conference on Robotics and Automation, IEEE, 2018, pp. 6939–6946 (2018).

[15] N. Radwan, A. Valada, W. Burgard, VLocNet++: Deep multitask learning for semantic visual localization and odometry, IEEE Robotics and Automation Letters 3 (4) (2018) 4407–4414 (2018).

[16] S. Brahmbhatt, J. Gu, K. Kim, J. Hays, J. Kautz, Geometry-aware learning of maps for camera localization, in: IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 2616–2625 (2018).

[17] J. J. Tarrio and S. Pedre. Realtime Edge-Based Visual Odometry for a Monocular camera. IEEE International Conference on Computer Vision (ICCV), pages 702–710, 2015.