Deep Visual Odometry for 3D Mapping

Name: Ruoyang Xu, Advisor: Fumin Zhang, Group Name: XXLs (Team 30)

Introduction

Mobile robotic platforms are achieving their level of automation via intelligent navigation capabilities and self-awareness of its motion and location is crucial to its success. Visual odometry is the process of determining such information through visual inputs. Deep visual odometry is the machine learning approach to this problem and exceed conventional visual odometry techniques in terms of end-to-end capabitlies. This technical review briefly summarizes some existing application of deep visual odometry, and underlying concepts of this technology and provides ideal implementation conditions for this technology.

Commercial Application of Deep Visual Odometry

Due to the software nature of visual odometry and its reliance on fast hardware, until the early 2010s, application of visual odometry primarily exists in the fields of academia and pure research, with the application on martian rover being the most prominent[1]. Recent application utilizing visual odometry orients around autonmous car and drone companies. Skydio uses 13 cameras to perform its visual based simultaneous localization and mapping task, which is a combination of visual odometry and localisation task and puts the price of the entire drone at $1999 [2]. A competing company of Skydio, DJI, annouced a hardware platform for visual odometry platform research priced at $1249 [3, 4] in 2015.

Despite the availability of hardware platforms for deep and conventional visual odometry algorithms, odometry methods rarely exists as a standalone ready-to-ship products. The majority of design comes as a proprietary part of bigger product such as those for Skydio and DJI, and other designs are released as open-source projects aimed at promoting research in thie field. SfMLearner is an deep visual odometry project open-sourced on GitHub done by Google. This work features in its ability to conduct unsupervised deep learning and generating depth data with single-view camera [5]. OpenVSLAM is another open-sourced GitHub project for building visual odometry research with fully modular design and compability across multiple camera models as its primary feature. [6].

Underlying Technology for Deep Visual Odometry

Deep visual odometry aims at utilising machine learning techniques to solve visual odometry methods. Conventional visual odometry revolves around a multi-stage process that the pipeline detects feature points in a series of visual inputs, match the feature points between consecutive images and attempts to estimate motion updates from the change in feature points inbetween visual inputs.

Wang et, al. proposed an end-to-end method for deep visual odometry in 2017 [7] that uses two different types neural network in sequence in an attempt to solve the entire multi-stage process in the same framework. This framework utilized a number of convolutional neural network for capturing image features and directly outputs the feature vector into a recurrent neural network. The recurrent neural network then outputs the most probable pose estimation of the camera position which effectively contains the odometry information of the robot.

A 2D convolutional neural network extracts its feature by convolving a 2D matrix over input matrix and any images that has a match to the learned kernel inside the neural network would outputs a high value in convolution and therefore forms a feature vector. A recurrent neural network accepts the input of $t = \tau$ as well as $t = \tau - 1$ so temporal continuity is taken into consideration. A recurrent neural network outputs in the fashion that for every output at $t = \tau$ it generates a set of most probable output for $t = \tau + 1$ and select the most possible result for the next timestep.

Other techniques in deep visual odometry includes using k-nearest neighbour(k-NN) and support vector machine(SVM) to capture the feature changes in optical flow [8]. k-NN and SVM are both supervised classification algorithms, where k-NN determines the class of one particular feature vector based on the class of its neighbors in high dimensional plane and SVM classifies data set by placing a hyperplane that separates classes of different data.

Implementation of Deep Visual Odometry

Software Implementation

The implementation of a deep visual odometry for the blimps robot will revolve primary in the software. Either adoping an end-to-end framework or a non end-to-end framework, machine learning algorithms are required and several software packages are needed to implement these algorithms. TensorFlow and PyTorch are the dominant machine learning packages for implementing a machine learning algorithm [9, 10].

Simulated and Real Life Testing

Benchmarking the implemented algorithm requires a testing ground and simulated environment such as Gazebo used in conjunction with ROS provides a platform for both running and testing the algorithm [11]. The final deployment of the result is expected to run on the Blimps platform that is already constructed in the lab, see project description.

Design Constraint By Limitation of Hardware Platform

Due to the limit of final deployment being a light-weight mobile platform (a small indoor Blimps), considerable design choices are limited. A light-weight mobile platform limits the on-board computation ability and effectively eliminates the possibility of any online/adaptive algorithms. Options include minimizing and approximating the final implementation to be computation light-weight or design the system so that it transmits any inforamtion to remote workstation.

References

[1] M. Maimone, Y. Cheng, and L. Matthies, "Two years of visual odometry on the mars exploration rovers," *J. Field Robotics*, vol. 24, pp. 169–186, 03 2007.

[2] L. Z. Wei, "Lessons for deep-tech startups from skydio, the autonomous drone company," Feb 2018, accessed: 2019-09-29. [Online]. Available: https://medium.com/zhanwei/lessons-for-deep-tech-startups-from-skydio-the-autonomous-drone-company-61fa890ea990

[3] Stever, "Dji matrice 100 & dji guidance announced," Jun 2015. [Online]. Available: https://www.heliguy.com/blog/2015/06/09/dji-matrice-100-dji-guidance-announced/

[4] G. Zhou, Lu Fang, Ketan Tang, Honghui Zhang, K. Wang, and K. Yang, "Guidance: A visual sensing platform for robotic applications," in *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, June 2015, pp. 9–14.

[5] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 6612–6619. [Online]. Available: https://github.com/tinghuiz/SfMLearner

[6] S. Sumikura, M. Shibuya, and K. Sakurada, "OpenVSLAM: A Versatile Visual SLAM Framework," in *Proceedings of the 27th ACM International Conference on Multimedia*, ser. MM '19. New York, NY, USA: ACM, 2019. [Online]. Available: https://doi.org/10.1145/3343031.3350539

[7] 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)*, May 2017, pp. 2043–2050.

[8] A. Kreimer, "Algorithms for visual odometry," Ph.D. dissertation, Israel Institute of Technology, 2019.

[9] M. A. et al., "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[10] 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," in *NIPS Autodiff Workshop*, 2017.

[11] B. Gromov and J.-H. Ryu, "Implementation of semi-virtual multiple-master/multiple-slave system," 10 2013, pp. 243–246.