SuperPointVO: A Lightweight Visual Odometry based on CNN Feature Extraction

Xiao Han, Yulin Tao, Zhuyi Li, Ruping Cen, Fangzheng Xue*
College of Automation, Chongqing University, Chongqing 400044, China
{xiaohan, 20173933, 20173932, rupingcen, xuefangzheng}@cqu.edu.cn

***Abstract*—In this paper, we propose a lightweight stereo visual odometry (SuperPointVO) based on feature extraction of convolutional neural network(CNN). Compared with the traditional indirect method of VO system, our system replace the hand-engineered feature extraction method with a CNN-based method. Based on the feature extraction network SuperPoint, we discard the redundant descriptor information it extracted, and expand the expression ability of the descriptor through NMS and grid sampling, making it more suitable for VO tasks. We build a complete stereo VO system without loop closing around the modified feature extractor. In the experiments, we evaluate the performance of the system on the KITTI dataset, which is close to other state-of-the-art stereo SLAM system. This shows that the accuracy and robustness of feature extraction methods based on deep learning are comparable to, or even better than the traditional methods in VO tasks.**

***Index Terms*—SLAM, Visual Odometry, CNN, feature extraction.**

I. INTRODUCTION

Simultaneous Localization and Mapping(SLAM), as a basic technology for robots to realize autonomous navigation and planning in complex environments, has attracted great attention in the computer vision and robotics communities in the past few decades. Among different sensor forms, cameras are cheap and can provide rich environmental information, thereby achieving reliable and accurate position recognition. The pipeline of vision SLAM generally includes four parts: front-end visual odometry, back-end optimization, loop detection and mapping. Visual odometry (i.e. relative motion estimation based on visual information) is the cornerstone of the visual SLAM system. For feature-based VO system, a widely followed classic pipeline includes feature extraction, feature matching, outlier rejection, motion estimation and local optimization based on bundle adjustment(BA). It is important to be able to efficiently extract robust features and descriptors for matching for VO systems which require high real-time performance. In addition, the robustness of the VO system greatly depends on the type of feature points. Until recently, hand-engineered feature extraction methods (SIFT[1], SURF[2], ORB[3]) were still widely used. ORBSLAM2[4] represents the state-of-the-art technology in the feature-based methods.

Deep learning has recently dominated many computer vision tasks and achieved excellent fairly good performance, especially in solving identification and classification problems.

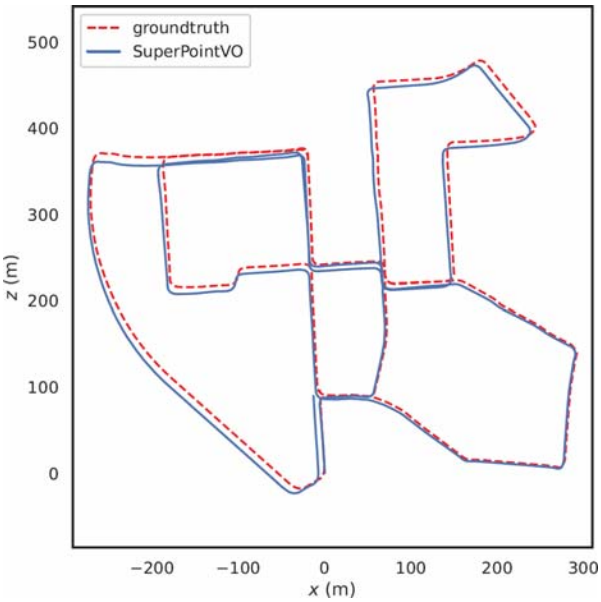


Fig. 1. SuperPointVO's evaluation trajectory on the KITTI dataset secquence 00. Estimated trajectory (blue) and ground-truth (red).

There is a trend in SLAM to investigate deep learning-based methods. Some modules and even the entire pipeline in the SLAM framework can be replaced with deep learning methods. For VO systems, the application of deep learning has two main directions. One is to learn poses directly from images in an end-to-end manner. [5–7] define the state-of-the-art technology in this category. The other is to replace one or more modules in the traditional VO pipeline with deep learning methods. Because pose estimation and local optimization in traditional methods are geometric optimization problems, deep learning is more suitable for replacing modules without mathematical expressions in a closed form, such as solving stereo depth[8], feature point extraction[9]. These methods based on deep learning have achieved better results than traditional methods to some extent. However, these methods are difficult to achieve real-time performance with sufficient computing resources.

This paper proposes a lightweight stereo visual odometry based on deep learning feature extraction. Our system generally maintains the traditional VO pipeline and replaces

978-1-7281-9888-0/20/$31.00 ©2020 IEEE

the feature extraction module with a method based on deep learning. Our contributions include:

- Based on the feature extraction network SuperPoint[10], we discard the redundant descriptor information it extracted, and expand the expression ability of the descriptor through NMS and grid sampling, making it more suitable for VO tasks.
- We build a complete VO system without loop closing around the modified feature extractor. Its lightweight framework guarantees the efficiency of the system.
- The experiment shows that our deep learning-based system performs equivalent to the classic SLAM system. Fig.1 shows the evaluation results of our system on the KITTI[11] dataset.

The rest of this article is organized as follows. Section II gives a review about related work. Our system is described in detail in Section III, and the experimental results are given in Section IV. Section V serves as a conclusion.

II. RELATED WORK

In this section, we mainly cover related work in two areas. Firstly, VO and SLAM methods are introduced, and then we mainly discuss image feature extraction and matching methods based on deep learning in the SLAM field.

A. VO and SLAM

In the indirect method, the first step of a typical SLAM pipeline is to extract keypoints and then match these keypoints to previous frames to estimate motion. Matching is based on descriptors of keypoints and geometric constraints. ORB-SLAM2[4] still defines the state-of-the-art technology in this category. It supports three image sensors: monocular, stereo and RGB-D. The entire system is calculated around ORB[3] features, including visual odometry and ORB vocabulary for loop closing. It showed that the ORB feature is an excellent compromise between efficiency and accuracy in current computing platform. ORB-SLAM[12] innovatively uses three threads to complete SLAM, including tracking, local BA, loop closing and global BA. Its excellent loop closing algorithm ensures that the system effectively prevents accumulative errors and can be quickly recovered after loss.

In addition, there is SVO[13] based on semi-direct method, which refers to the mixed use of feature-based method and direct method. Compared with other solutions, the biggest advantage of SVO is that it is extremely fast. Real-time performance can be achieved on low computing platforms, and hundreds of HZ can be achieved on computers. In SVO2[14], the speed reached 400HZ. This makes SVO suitable for applications where computing platforms are limited, such as drones and handheld AR / VR devices.

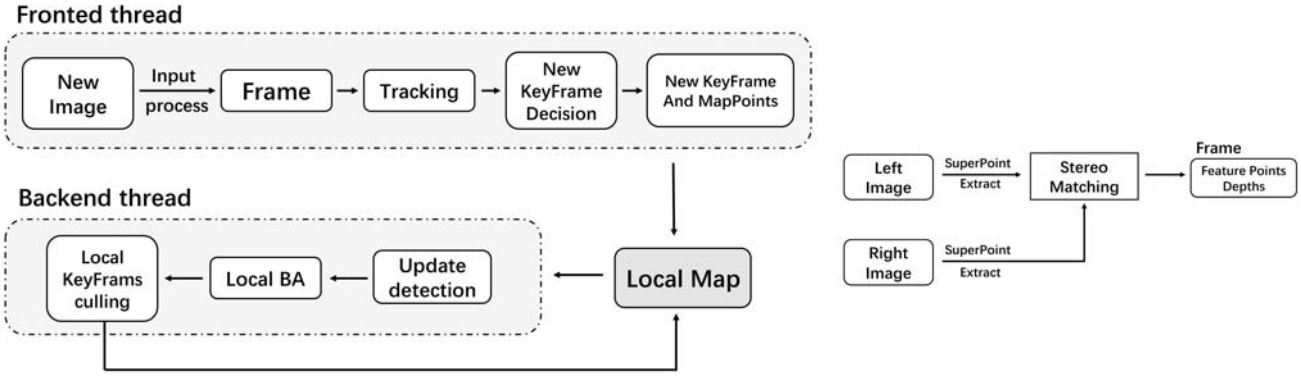
In recent years, the combination of deep learning and SLAM has become one of the trends. Some modules and even the entire pipeline in the SLAM framework can be replaced with deep learning methods. CNN-SLAM[15] is a relatively complete pipeline. It replaces both the depth estimation and image matching in LSD-SLAM with CNN-based methods. It has obtained more robust results and can fuse semantic information. VINet[6] uses CNN and Recurrent Neural Network(RNN) to build a Visual Inertial Odometry(VIO), which inputs the image and IMU information then directly outputs the estimated pose. The focus in [16] is deep learning-based single view depth estimation to reduce the scale drift inherent in monocular systems. [8, 17] Using Deep learning in stereo image matching to recover depth information. UnDeepVO[7] implements a monocular visual odometry based on an unsupervised learning method. The experiments on KITTI dataset show UnDeepVO achieves good performance in terms of pose accuracy. However, geometry-based optimization methods still outperform end-to-end systems as shown in DeepVO[5].

B. Image Feature Extraction and Matching based on Deep Learning

Local feature extraction and matching play an important role in the computer vision field, and a lot of research work is devoted to finding methods that can efficiently extract more robust feature information. In previous years, the best technology relied on hand- engineered features (SIFT, SURF, ORB). Until recently, some methods based on machine learning technology began to approach or even outperform those traditional methods.

A typical feature extraction pipeline consists of three parts: finding keypoints, calculating their directions and extracting robust expressions. LIFT[9] proposes a deep network framework, which mainly includes three modules: Detector, Orientation Estimator and Descriptor. Each module is a separate CNN network. Previous work TILDE[18], Learn Orientation[19] and DeepDesc[20] have proven that individual tasks can be handled well using the CNN network. This article unifies them into a large framework and uses back-propagation for end-to-end training. The training process starts from back to front, training descriptors first, then training the Orientation Estimator, finally training keypoints detection modules. From the experimental comparison results of the authors, this method is more robust to light and seasonal changes compared with the traditional SIFT[1] method.

GCNv2[21] is a new implementation of feature extraction and description method based on deep learning. This method uses a neural network to extract robust keypoints and binary feature descriptors with the same shape of the ORB. The first version of Geometric Correspondence Network(GCN) was implemented by a revised Fully Convolutional network with a Resnet-50 backbone and a bidirectional recurrent network. This structure has achieved good accuracy, but cannot be real-time. The new network designed by the author makes independent prediction for each cell (16x16 pixels) of an original image. This improves computing efficiency, so it can achieve real-time performance on embedded systems such as Jetson TX2. The descriptor of GCNv2 is binary like ORB, which can reduce the amount of matching calculations, to improve efficiency. GCNv2 is convenient to replace submodules of other SLAM systems, such as ORB-SLAM2 or SVO. The effectiveness and robustness of GCNv2 was verified



(a) Framework (b) Input Process

Fig. 2. SuperPointVO is composed of two main parallel threads.The front-end thread first processes the stereo image to obtain keypoints and descriptors, and then performs pose estimation.When the back-end thread detects new keyframe and map points insertions, it performs BA optimization on the local map.

by running GCN-SLAM on a real drone, and it was shown that it can handle the failure of ORB-SLAM2.

SuperPoint[10] is a state-of-the-art self-supervised framework for keypoint and descriptor extraction. Its performance on Hpatches[22] dataset is better than ORB and SIFT. The training system of the entire network consists of two parts: MagicPoint and Homographic. The former is a corner detector trained by rendering pictures with corner truth values. The latter trains the SuperPoint network by performing homography transformation on the detected pictures to obtain picture pairs with a certain transformation relationship. SuperPoint is a system for keypoint detection and descriptor, so the network consists of a common shared encoder and two separate decoders for generating detection locations and their 256-dimensional descriptor. The core of the codec is a typical VGG-like[23] network. Since the loss function is calculated by pose in network training, the proposed keypoints and descriptors have more advantages in pose solution. If the keypoints and descriptors extracted in this way are used in SLAM, the front-end algorithm will be improved. Descriptors learned by the network have better expressive ability than manually designed descriptors, and can be more robust to the changes of seasonal and lighting in localized scenes.

III. SUPERPOINTVO

Because it takes relatively more time while the feature extraction method based on neural network using in a full image. Comparing with other VO, the system structure of SuperPointVO is more streamlined to ensure the efficiency of the system. Fig.2 shows the system framework of SuperPointVO. The algorithm uses two parallel threads, which combine with a local map. One thread quickly handles inter-frame tracking to ensure real-time performance, and the other thread optimizes the local map to ensure good results.

The front-end thread performs pose estimation by extracting and matching image keypoints. The first step is to extract the keypoints and its descriptors of the image through SuperPoint, which based on convolutional neural network. Our system supports stereo input images. For stereo images, the keypoints are first extracted by SuperPoint in the left and right images, and then the depth of keypoints is restored by stereo matching of the descriptors, which shown in Fig.2(b). In the next step, the current keypoints match with the keypoints of the previous frame to obtain corresponding 3D map points, and then estimate the pose of the frame by minimizing the reprojection error of the map points in the frame. When there are fewer matches or the tracking is about to be lost, the system creates new keyframe and constructs new map points. The results of the front-end processing are used as the initial value of the back-end optimization.

The back-end is a slower thread. When new keyframe and map points insertions are detected, BA optimization is performed on a local map. As time goes on, after the end of back-end optimization, far away keyframes and related map points should be eliminated to control the scale of the local map to ensure the optimization efficiency. The local map serves as a bridge connecting the front-and back-end, and we maintain its scale to 7 keyframes and related map points.

A. CNN-based Feature Extraction

The core of SuperPointVO is to replace the module of extracting image features in the traditional SLAM pipeline with a method based on deep learning. Recently, some methods based on machine learning technology (deep learning) have begun to approach or even surpass those classic detection algorithm. The features extracted by a massive data supported trained convolutional neural networks are more robust, and the descriptors used for matching have better expression capabilities.

The SuperPoint[10] is an advanced self-supervised framework for keypoint and descriptor extraction. We have made some improvements to its prediction process to make it better integrate with the VO system. The entire feature extraction pipeline is shown in the fig.3. The grayscale image first input into a shared VGG-style encoder to reduce the image size. The encoder consists of convolutional layers, pooling

layers and non-linear activation function. For $H \times W$ size images, $Hc = H/8$ and $Wc = W/8$ are obtained after three 2×2 non-overlapping maxpooling layers. The encoder maps the input image $I \in \mathbb{R}^{H \times W}$ to an intermediate tensor $B \in \mathbb{R}^{Hc \times Wc \times 128}$ with a small spatial dimension and a large channel depth.

The second half of the network consists of two independent decoders. The first one is responsible for extracting the pixel position of the keypoint, and the intermediate tensor B is mapped to $X \in \mathbb{R}^{Hc \times Wc \times 65}$ through the convolution layer. The 65 channels of X correspond to a local 8×8 grids area and a mark that characterizes the area without keypoints. After the channel-based Softmax, the identification bits are discarded and X is explicitly reshaped into a $heatmap \in \mathbb{R}^{H \times W}$. Each pixel in the heat map corresponds to the probability that the pixel is a keypoint in the input image. The coordinates of a series of keypoints are obtained after threshold filtering the heatmap. Because the network may extract multiple probabilistic keypoints in local area, we perform a non-maximum suppression (NMS) of 8×8 grids on all keypoints to ensure that there is only one valid keypoint in the local area. The second decoder is responsible for extracting descriptors of keypoints. The intermediate tensor B is mapped into a semi-dense descriptor grid $D \in \mathbb{R}^{Hc \times Wc \times 256}$ through the convolutional layer. Grid D does not include descriptor information for all pixels (i.e. 8×8 pixels in the original input image share a descriptor). The next step is to use the bicubic interpolation algorithm for the semi-dense grid to obtain the descriptor of each pixel. Because the bicubic interpolation algorithm takes relatively more time, it is not suitable for a system with high real-time requirements such as VO. We skip the bicubic interpolation algorithm to directly perform L2 normalization on the descriptor grid and then use grid sampling to obtain a 256-dimensional floating-point descriptor corresponding to the keypoint. Also, grid sampling of descriptor is another reason why keypoints require NMS. The descriptor corresponding to the keypoint obtained by this method does not represent a single pixel, but a pixel block (8×8). Such descriptors may be more robust in image tracking matching between frames.

In specific engineering implementation, because SuperPointVO is implemented in C++, and the original author of SuperPoint only released its python version of the prediction model. We rebuilt an improved version of the model through C++ and used the released weight parameters.

B. Frontend Tracking

SuperPointVO supports stereo input image data. Compared to monocular images, which require initialization of polar geometry and triangulation, stereo images can be initialized using only one frame by depth information. When the system starts, we use the first frame to create a keyframe and set its pose as the map's original point. And all keypoints of the frame are created as map points.

For each subsequent frame, we perform interframe descriptor matching for pose tracking. We follow the principle of matching consistency, (i.e. cross-check the matching results,

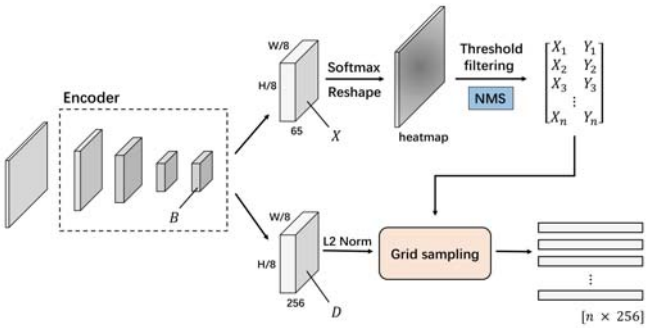


Fig. 3. SuperPoint[10] is an advanced self-supervised framework for keypoint and descriptor extraction.The grayscale image first input into a shared VGG-style encoder to reduce the image size. The second half of the network consists of two independent decoders. One is responsible for extracting the pixel position of the keypoint,the other is responsible for extracting descriptors.

and the keypoints in the two frames should be the best match to each other). After matching with the keypoints of the previous frame, the system obtained the correspondence between the keypoints of the current frame and the local map points. We then execute Motion BA to optimize the camera pose for the current frame. This motion BA is a non-linear optimization problem that only optimizes the camera pose. We use the Levenberg–Marquardt method provided by g2o[24]. For all matches, the system finds the optimal camera direction $\mathbf{R} \in SO(3)$ and position $\mathbf{t} \in \mathbb{R}^3$ by minimizing the reprojection error between the 3D map point $P_i \in \mathbb{R}^3$ and the matching keypoint $u_i \in \mathbb{R}^2$ in the world coordinate system. Our system uses a constant velocity motion model to determine the initial value of the pose of the current frame (i.e. the slight offset of the pose of the previous frame in the direction of motion). The optimization function is defined as follows, where $i \in N$ is the set of all matches.

$$\{\mathbf{R}, \mathbf{t}\} = \underset{\mathbf{R},\mathbf{t}}{\operatorname{argmin}} \sum_{i \in N} \left\| u_i - K(\mathbf{R}P_i + \mathbf{t}) \right\|^2 \tag{1}$$

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

Here K is the internal parameter matrix of the camera. f_x, f_y are the focal lengths of the camera, (c_x, c_y) is the principal point, all known from calibration. In addition, the above formula omits the process of projecting a 3D point in the camera coordinate system onto a normalized plane.

C. Backend Optimization

When the front-end constructs a new keyframe and inserts it into a local map, the back-end will detect update of the map and then trigger a BA optimization (including the first frame initialization). Different from the front-end motion BA optimization, the local BA not only optimizes the pose of keyframes, but also includes the positions of all map points that have a projection constraint relationship with these keyframes. The complexity of the local BA will

increase with the number of keyframes. In order to maintain a compact local structure, our system only optimizes one set of activated keyframes $\mathbf{F_L}$ and the set of map points that can be observed by these keyframes M_S. We define the set of matching relations between map point $\mathbf{P_i}$ and keypoint u_i in a certain frame as N_L. The optimization function is as follows:

$$\{\mathbf{P}_i, \mathbf{R}_l, \mathbf{t}_l | i \in M_s, l \in F_L\} = \underset{\mathbf{P}_i, \mathbf{R}_l, \mathbf{t}_l}{\operatorname{argmin}} \sum_{k \in F_l} \sum_{j \in N_l} E_{kj}$$

$$E_{kj} = \|u_j - K(\mathbf{R}_k \mathbf{P}_j + \mathbf{t}_k)\|^2 \tag{3}$$

We use the Levenberg–Marquart method provided by g2o to optimize the above problem. As time goes on, the system performs dynamic management of the active keyframe group, and the newly inserted keyframe of the local map also insert the active keyframe group. In order to control the size of the local map and ensure the efficiency of the optimization, we will remove keyframes that are far from the current frame position and that are longer in time after the optimization. At the same time, the correspondence between these keyframes and the map points that can be observed are also removed. Those isolated map points that can be observed without any frame are also deleted from the local map. After experimental tests, out of the balance of efficiency and accuracy, our system maintains 7 active keyframes.

IV. Experimental results

We introduce the evaluation results of SuperPointVO on the well-known KITTI[11] dataset and use the standard evaluation metrics published by the original author to compare them with other state-of-the-art stereo SLAM system. We ran SuperPointVO on a desktop computer (Intel Core i7-8700K with a GTX 1080ti GPU) to evaluate the system runtime. In order to accurately assess the system accuracy, we take repeated experiments on each data set to avoid multi-threaded uncertainty.

A. Runtime

Table I shows the time evaluation results of the modified version of the SuperPoint feature extraction module. We evaluated the forward process of the network and the running time of NMS and sampling at 2 image resolutions. The results show that the time spent by NMS and sampling is higher than the forward process of the neural network, which limits our system performance. But the architecture of other parts of VO is lightweight, our system can work at 20HZ with a resolution of 640×480 pixels, and for a resolution of 1240×376 pixels, the system can still maintain working at 14HZ.

TABLE I
Runtime of Feature Extraction

Resolution	Forwad Pass	NMS and Sampling	Total
640×480	9ms	10ms	19ms
1240×376	12ms	20ms	32ms

TABLE II
Evaluation Results on KITTI Dataset

	t_{rel}		t_{abs}		
Sequence	SuperPoint	LSD-SLAM	SuperPoint	ORBVO	LSD-SLAM
00	0.98	0.63	3.4	X	**1.0**
01	X	2.36	X	X	**9.0**
02	1.35	0.23	15.9	X	**2.6**
03	0.94	1.01	**0.8**	10.8	1.2
04	0.64	0.38	0.4	1.8	**0.2**
05	0.92	0.64	3.4	X	**1.5**
06	0.95	0.71	1.9	40.7	**1.3**
07	5.77	0.56	13.6	20.5	**0.5**
08	1.57	1.11	4.5	X	**3.9**
09	1.42	1.14	**4.9**	X	5.6
10	0.92	0.72	1.8	24.5	**1.5**

t_{rel}: average translational RMSE drift (%) on length of 100m-800m.
t_{abs}: absolute translation RMSE (m).

B. KITTI dataset

In order to better evaluate the performance of the improved version of SuperPoint in VO tasks, we keep the lightweight system framework unchanged and replace SuperPoint with traditional ORB features to build ORBVO as a comparison. The number of keypoints extracted by each system in each frame are approximately equal. Since the feature extraction and matching parts of the system are decoupled from other modules, the two different systems have exactly the same process for pose estimation and optimization. In addition, the famous stereo LSD-SLAM[25] is also compared as state-of-the-art SLAM system.

We evaluated our system on the well-known KITTI dataset, which contains stereo sequences recorded from cars in urban and highway environments. The stereo sensor has a baseline of approximately 54cm and works at 10Hz with a resolution of 1240×376 pixels after rectifying. Table II shows the evaluation results of SuperPointVO, ORBVO and stereo LSD-SLAM on 11 sequences with public ground-truth. We use two different metrics, the absolute translation RMSE t_{abs} and the average relative translation t_{rel}. Our system achieves good accuracy on the shorter time sequences 03, 04, 06, 08, 09 and 10, and the relative error is about 1%. Since our system framework is lightweight and does not have a module of loop closure detection, the accumulative error cannot be eliminated on long-term and having loop closure sequences 00, 02, 05 and 07, so the system performance is slightly worse. In addition, Sequence 01 is the only highway sequence in the test set, and our system failed to track due to the influence of moving objects.

The KITTI dataset provides images captured at 10Hz, and the camera moves at speeds up to 80 km/h, which is a challenge for the variable-scale matching of the grid descriptors extracted by SuperPoint. Unlike ORB features that use image pyramids, feature extraction in our system only

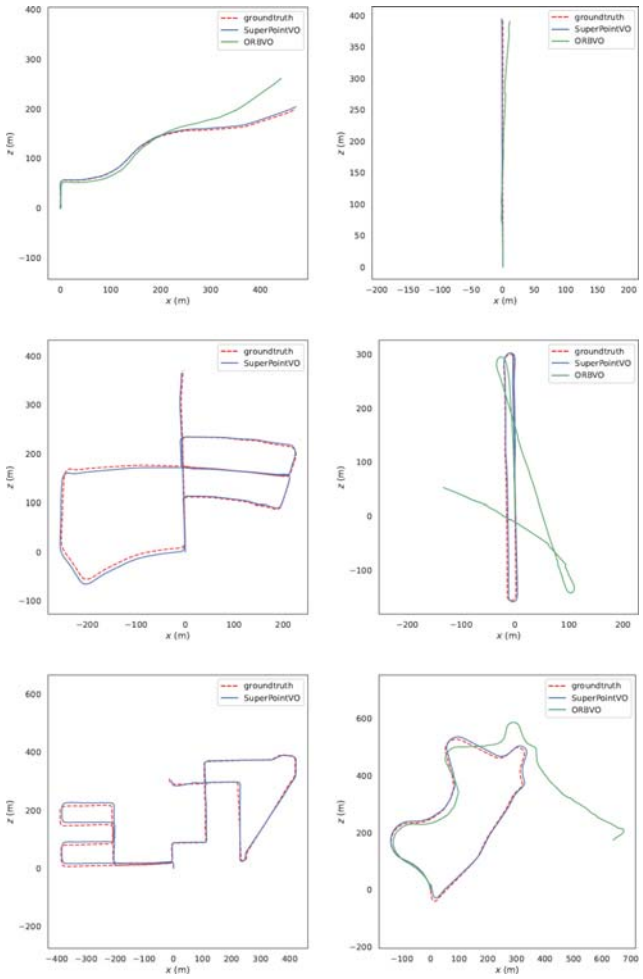


Fig. 4. Estimated trajectory (blue and green) and ground-truth (red) in KITTI 03, 04, 05, 06, 08 and 09.

applies to a single scale. The experimental results show that SuperPoint improved by NMS and grid sampling has good performance in variable-scale matching, the neural network can learn the robust descriptor information, and the feature extraction method based on deep learning has an appealing advantage in the VO problem.

As a comparison, we also evaluated ORBVO on the KITTI dataset. Since the system framework is lightweight and there are no complicated design techniques for improving accuracy and robustness, the system using ORB feature extraction and matching performs extremely badly. It failed to track on the sequences 00, 01, 02, 05, 08 and 09 that are long-term and having loop closure, And the errors in the sequences 03, 04, 06, 07 and 10 are very large. Compared with ORBVO, SuperPointVO only changes the method of feature extraction and matching, but the performance is greatly improved under the same system framework. In multiple sequences, the performance of SuperPointVO is close to or even exceeds stereo LSD-SLAM. This shows that the feature extraction method based on deep learning can be comparable to or better than traditional methods in VO tasks without complicated design techniques that improve accuracy and robustness. The system that achieves this performance requires only about 2,000 lines of code. Fig.4 shows some examples of estimated trajectories.

V. Conclusion

In this paper, we proposed SuperPointVO, a lightweight stereo visual odometry based on CNN feature extraction. This VO system maintains the traditional system pipeline and replaces the manual-based feature extraction method with a CNN-based method. Based on the feature extraction network SuperPoint, we discard the redundant descriptor information it extracted, and expand the expression ability of the descriptor, making it more suitable for VO tasks. In addition, BA-based tracking and back-end optimization ensure accuracy, while dynamic local map management ensures system efficiency.

In the experiments, we evaluated the runtime of the modified feature extractor. Although it has a certain gap with the traditional feature extractor, its better performance is still a good choice under the conditions of GPU. We evaluated the system on the KITTI dataset and its performance is close to other state-of-the-art stereo vision SLAM systems. This system performance is achieved in a lightweight framework of about 2,000 lines of code. This shows that the accuracy and robustness of feature extraction methods based on deep learning are comparable to, or even better than the traditional methods in VO tasks.

In future work, we consider expanding SueprPointVO into a complete SLAM system, including a closed-loop detection module and more processing details, which can further improve the accuracy and robustness of the system. Finally, we also want to change the network structure of the feature extractor and train it to extract more advanced information for matching and tracking.

References

[1] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91—110, 2004.

[2] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *European conference on computer vision*. Springer, 2006, pp. 404–417.

[3] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International conference on computer vision*. Ieee, 2011, pp. 2564–2571.

[4] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[5] 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, pp. 2043–2050.

[6] R. Clark, S. Wang, H. Wen, A. Markham, and N. Trigoni, "Vinet: Visual-inertial odometry as a sequence-to-sequence learning problem," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[7] 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, pp. 7286–7291.

[8] J. Žbontar and Y. LeCun, "Stereo matching by training a convolutional neural network to compare image patches," *The journal of machine learning research*, vol. 17, no. 1, pp. 2287–2318, 2016.

[9] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua, "Lift: Learned invariant feature transform," in *European Conference on Computer Vision*. Springer, 2016, pp. 467–483.

[10] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superpoint: Self-supervised interest point detection and description," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 224–236.

[11] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[12] 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.

[13] C. Forster, M. Pizzoli, and D. Scaramuzza, "Svo: Fast semi-direct monocular visual odometry," in *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 15–22.

[14] 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, 2016.

[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*, 2017, pp. 6243–6252.

[16] N. Yang, R. Wang, J. Stuckler, and D. Cremers, "Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 817–833.

[17] W. Luo, A. G. Schwing, and R. Urtasun, "Efficient deep learning for stereo matching," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5695–5703.

[18] Y. Verdie, K. Yi, P. Fua, and V. Lepetit, "Tilde: A temporally invariant learned detector," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5279–5288.

[19] K. Moo Yi, Y. Verdie, P. Fua, and V. Lepetit, "Learning to assign orientations to feature points," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 107–116.

[20] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer, "Discriminative learning of deep convolutional feature point descriptors," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 118–126.

[21] J. Tang, L. Ericson, J. Folkesson, and P. Jensfelt, "Gcnv2: Efficient correspondence prediction for real-time slam," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3505–3512, 2019.

[22] V. Balntas, K. Lenc, A. Vedaldi, and K. Mikolajczyk, "Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5173–5182.

[23] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[24] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A general framework for graph optimization," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 3607–3613.

[25] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in *European conference on computer vision*. Springer, 2014, pp. 834–849.