Electronics, Communications and Networks
A.J. Tallón-Ballesteros et al. (Eds.)

doi:10.3233/FAIA231259

Self-Supervised Sparse Direct Visual Odometry with Half-Geometric Correspondence Network

Tenglong ZHANG[a,b,c] , Qing LI [a,b,c,1] Miaosheng ZHOU[d] and Fei YU[e]
[a] *Beijing Information Science & Technology University, Beijing 100192, China*
[b] *Beijing Key Laboratory of High Dynamic Navigation Technology, Beijing 100192, China*
[c] *Ministry of Education Key Laboratory of Modern Measurement & Control Technology, Beijing 100192, China*
[d] *Beijing SinsTek Measurement & Control Co.,Ltd, Beijing 101300, China*
[e] *Beijing Star Network Yuda Technology Co, Beijing 100097, China*

Abstract. Current mainstream visual odometry method often suffers from tracking lost in low texture and motion blur scenarios due to fewer effective features and difficulty in getting stable matches. And feature matching process affects the overall real-time performance. For the fast localization task in low texture environments, this paper proposes an efficient self-supervised direct visual odometry framework based on keypoint extraction network, HGCN-VO. First we build the half-geometric correspondence network, HGCN, for fast extraction of robust keypoints in images. During training, we propose training method which uses basic shape elements to render generated simulated images with pseudo-labels as well as random homography transformations on real images for pre-training and migration learning and optimizing the keypoint loss from forward and reverse perspective transformed images. Finally we optimize the inter-frame position using a multilayer sparse direct method combined with bundle adjustment to improve the robustness of the method in low texture environments while increasing the processing speed. We evaluated the proposed method in KITTI, TUM, and challenging low-texture real-world scenarios and compared it with the current mainstream visual odometry methods, and the results show that the algorithm is sufficiently robust and accurate in low-texture environments and has a fast processing speed.

Keywords. Visual odometry, feature extraction, pose estimation, self-supervised learning, spares direct method

1. Introduction

With the popularity of machine vision technology, the use of visual sensors to realize mobile robots' localization in three-dimensional space has been widely researched[1]. Visual Odomtry (VO) can recover the positional attitude information of the camera from the video captured by the vision sensor, which is an important part of visual SLAM and

[1] Corresponding Author: Qing Li, Beijing Information Science & Technology University, Beijing, 100192, China; E-mail: liqing@bistu.edu.cn.

plays a crucial role in the fields of unmanned navigation, autonomous driving, and augmented reality[3], etc.

Traditional sparse feature-based VO methods have limited accuracy and robustness in challenging environments (texture loss, motion blur, dynamic scenes) because the fact that the algorithm obtains fewer valid, matchable feature points in a given environment, and therefore often fails to track[6]. As deep learning gradually domi-nates computer vision tasks[7]. In general, deep learning-based VOs have the following drawbacks: large-scale neural networks are required to generate accurate image features[10], which directly leads to the fact that the deployability and real-time performance of visual odometry cannot be guaranteed; The descriptors generated by the neural networks are generally poorly matched, which also causes the increase in the feature matching time[11]; In addition, the pose solving process of the end-to-end network that obtains the camera pose from the input image is also unreliable and difficult to adjust.

To address these issues, this paper proposes a new visual odometry framework, HGCN-VO. our contributions are as follows:

1. A fast and robust keypoint extraction network HGCN is proposed. The network has a shorter inference time than current feature extraction methods since it inducts based on a single frame of low-resolution images and is not trained for matching.
2. A self-supervised transfer learning approach is used to train HGCN networks. The network is pre-trained using computer-synthesized rendered geometries with pseudo-labels and further trained using a combination of homographic adaptation and geometric correspondence to reduce the dependence on the dataset and improve the robustness of the algorithms in the migration setting.
3. We combine the keypoint extraction network HGCN with the multilayer sparse direct method to form an efficient self-supervised direct visual odometry framework based on a feature extraction network, which we call HGCN-VO.We evaluate the method of this paper on publicly available datasets as well as realistic and challenging scenarios, and demonstrate the effectiveness of the method of this paper.

The rest of the paper is organized as follows: in Section 2 we introduce the related work on feature extraction and visual odometry methods. Section 3 describes the construction and training of the feature extraction network and the sparse direct visual odometry framework. Section 4 shows comparison results in different scenarios. In Section 5 we summarize some conclusions and outlook for future work.

2. Related work

A feature represented by the position of a point is one of the simplest image features. For feature extraction, even though the traditional geometric model-based sparse feature extraction methods[12] are currently the preferred solutions, most of their extracted feature points suffer from poor scale invariance or the number of extracted features is insufficient in low texture environments[15]. Recent research results[16] show that the features generated by convolutional neural networks are more robust than conventional features. Mihai et al. [17] used a single convolutional neural network for dense feature and descriptor extraction, enabling the odometry to find image correspondences even in the presence of motion blur or image degradation. Tang et al.[10] proposed GCN, a scheme that combines a convolutional neural network and a recurrent neural network for keypoint extraction network and generates the descriptors in a unified architecture. Although the method has excellent tracking performance, its storage consumption is equally impressive, making the method difficult to deploy. Daniel et al.[18] proposed a lightweight feature point extraction network, Superpoint, which extracts features using a shallow VGG architecture and generates keypoints and descriptors using a dual-decoder structure, which have high real-time performance. Nevertheless, the descriptors generated in this way are of poor quality and prone to mismatching. Tang et al.[19], inspired by[18], proposed the GCNv2 network, which predicts image features in a single low-resolution mapping, and simplifies the structure of the original GCN network with a shallow convolutional neural network for feature encoding, and use two decoders to decode the keypoints and descriptors, input the keypoints to the descriptor decoder, and use billnear sampling to get the binarized descriptors similar to ORB_SLAM2[13] to reduce the matching time, and to improve the computing speed while ensuring the accuracy and robustness.

For visual odometers, feature-based methods represented by[13] and direct methods represented by [20] have now been widely developed and applied. The expense of complex feature extraction matching, low texture environment, etc. have caused great obstacles to the development of traditional methods. Therefore reseachers turned to introducing deep learning methods[7]. Current visual odometry methods based on model-learning fusion have shown advantages over traditional methods[22]. Loo et al.[22] used convolutional neural network to predict the average depth of the image, improve the depth uncertainty of the pixel points, and used a semi-direct method to estimate the camera motion; The method in [25] used ORB features as the extracted objects and, in the next step, combined them with neural networks for temporal modeling. Yang et al.[26] utilized deep neural networks for estimation at three levels: depth, pose and uncertainty to improve depth estimation accuracy. The limitations of such methods are also obvious: the inference of the deep learning part is slow, while the fusion of the two requires complex parameter settings.

We found that the keypoints extracted by convolutional neural networks are uniformly distributed and robust, while the descriptors may not be as stable[27], so we propose the idea of using the keypoints extracted by neural networks for the direct method, and propose a self-supervised learning scheme to train feature points that can be adapted to visual localization tasks in multiple scenarios.

3. Methodology

3.1. Overall architecture of HGCN-VO

The overall architecture of our HGCN-VO is shown in **Figure 1**. Specifically, we can divide the overall framework into two parts: feature extraction and motion estimation.

For the feature extraction part, we first preprocess the image obtained from the visual sensor. Then we calculate the displacement vectors of the key points by using the inter-frame transformation matrix of the previous frame as the initial value, and rely on the grayscales of the key points and the surrounding region to guide the optimization and achieve the alignment of the key points between frames without relying on descriptor matching. These points will be used for motion estimation in the current frame. If the keypoints does not reach the specified number after alignment, using HGCN to perform feature extraction then using non-maximum suppression to deduplicate keypoints, and if the features are still insufficient, then uniform sampling is performed, and in this way the number of keypoints used for alignment to the next frame is maintained.

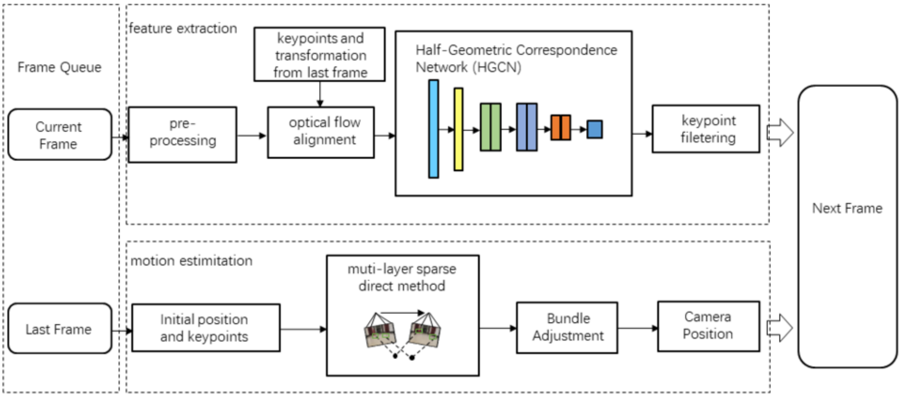


Figure 1. The framework of HGCN_VO

For the motion estimation part, we construct a 4-layer image pyramid for the input image, gradually increase the resolution to prevent local minima problems. In the next step we follow the idea of the inverse compositional method to minimize the inter-frame photometric error by optimizing an inter-frame transformation matrix that maps the keypoints of the previous frame and their surrounding blocks to the current frame, which in turn enables the camera position estimation. Finally, we perform Bundle Adjustment optimization on the iteration results to estimate a camera motion that minimizes the overall reprojection error of the keypoints, and input the estimated camera position as well as the tracked keypoints to the next frame. We will describe our work in detail in the next two subsections.

3.2. Half Geometric Correspondence Network

3.2.1 Network Architecture

The process of constructing the network in this paper is inspired by the GCNv2 network that utilizes multi-layer convolution to extract features and train models from warps. We abandon the process of descriptor training and feature matching and leave the task of pixel tracking to the subsequent sparse direct method, hence our network is named HGCN, specifically, our network architecture is shown in **Figure 2**.

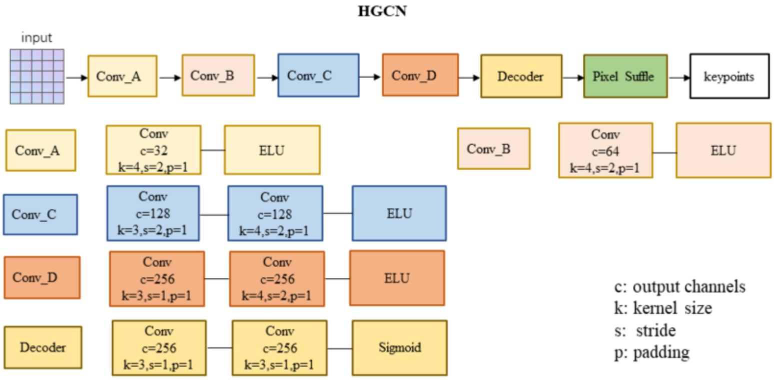


Figure 2. Architecture of HGCN

We use a single image as input, and the backbone network takes a VGG-like network architecture to encode the image for dimensionality reduction and predict the probability map at low resolution. For the simplicity of the network, we use four convolutional layers in the backbone to encode the image, to get the number of channels of 256, the size of the H/8 * W/8 feature map, followed by the use of two consecutive convolutional layers for the decoding of the keypoints, after a nonlinear activation, the use of pixel shuffle on the feature map for the up-sampling, and thus get the results of the keypoints of the detection.

Our network is able to save the spent on descriptor extraction and computation. In the end, we able to extract keypoints at 320fps (frames per second) on a device with an Intel i7-11800H and a Laptop version of the Nivia GeForce RTX 3070, which leaves a lot of time for the subsequent motion estimation part.

3.2.2 Self-supervised training based on homographic adaptation geometric and correspondence

Producing datasets specifically for keypoint training is difficult, and there is a tendency to train networks using artificial keypoints such as generated surf, harris etc. This approach ignores the focus on line features and is susceptible to environmental disturbances and poor geometric invariance[29].

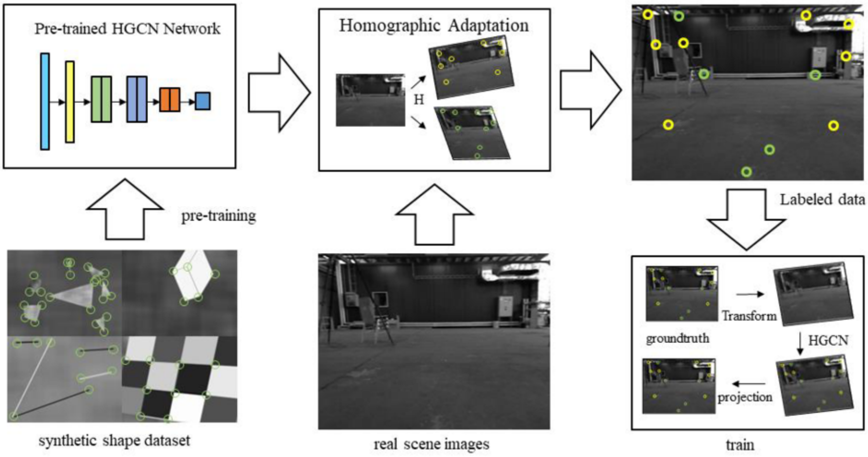


Figure 3. Schematic of training based on homographic adaptation and geometric correspondence

Our training method combines the advantages of both [10] and [18]. Specifically, our approach first pre-trains the HGCN using simulated images with pseudo-labels of virtual structural corner points randomly generated by rendering basic shape elements to obtain a pre-trained model. Next, a real dataset with randomized uni-responsive transformations is added to the network input, and the pre-trained detection model is used to generate keypoints for the images undergoing homographic transformations. The detection results under homograpic transformation under multiple angles are aggregated to generate data with detector labels under the real dataset. This enhances the generalization ability of the model in different scenarios. The process is shown in **Figure 3**. We constructed the keypoint detector F^(·) as in Equation (1).

$$\hat{F}(I;f_\theta)=\frac{1}{N}\sum_{i=1}^{N_h}H_i f_\theta(H_i(I)) \quad (1)$$

Where $f_\theta(\cdot)$ is the initial interest point detection function, and we tweak the detector on top of it. Assuming I is the input image and H is a random homomorphic transformation, we perform an empirical summation based on a sufficiently large number of H samples. The detector F^(·) adjusts the coordinates of key points in the new scene to make them more accurately represented in the coordinate system of the original scene, and we use this detector to automatically generate images with interest point labels. Finally, we directly input the self-supervised generated interest points and labels into our HGCN network for training. The overall loss function is composed of both:

$$Loss = -\frac{\lambda_p}{H_cW_c}\sum_{\substack{h=1\\w=1}}^{H_cW_c}\log\left(\frac{\exp(\eta_{hw}y_{hw})}{\sum_{i=1}^{c}\exp(\eta_{hwi})}\right)+\lambda_m\sum_{i}\max(0,s_p^2-s_n^2+m) \qquad (2)$$

Where H, W are the height and width of the image, η is the feature vector of X on the plane position, y_{hw} is the interest point position, the subscript represents the corresponding position of the interest point on the plane position, and c is the number of channels. s_p^2 and s_n^2 are the distances between positive and negative samples. m is the margin value. λ_p and λ_m are the weights of the Feature point loss and geometric correspondence loss.

During the course of our work we found that the network trained using this approach has better robustness in extracting binary features in various scenarios, whereas training using the original method only yields better results in certain environments. A comparison of the binary features extracted using the HGCN trained using the self-supervised training in this paper and the HGCN trained using the HGCN trained for the motion estimation task in the same scenario is shown in **Figure 4**. In addition, we have also included the traditional ORB features in the comparison.

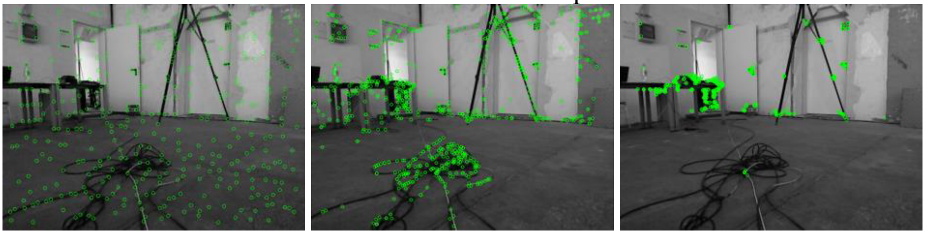


Figure 4. Effectiveness of different methods of feature extraction in low texture environments. HGCN(left), SIFT(middle), ORB(right)

From the results, it is obvious to see that the HGCN utilizing the self-supervised training method of this paper extracts richer feature points and has better geometric invariance.

3.3. Motion estimation with muti-layer sparse direct method and bundle adjustment

Our process is as **Figure 5**: We use the direct method to solve the inter-frame transform, taking the posture and pose of the previous frame as the initial posture of the current frame, and taking the result of the upper pyramid as the initial value of the lower layer, and solving frame by frame and layer by layer. Taking the position that minimizes the error in the optimization process between the two frames as inputs. The transformation matrix T between two frames that minimizes the photometric error is optimized by applying a perturbation to the previous frame.

For each layer, assuming that the projection points of the world point P in the previous frame of the three-dimensional world in the camera images of the two neighboring frames are p_{k-1}, p_k, and that it is necessary to minimize the photometric error using the minimizing photometric error to predict an optimal inter-frame transformation T that tracks the position of the pixel p_{k-1} in the previous frame in the current frame image. Assuming that T_{k-1} is a transformation matrix between k to k-1 frames and Ω is a region of the image with known image depth, the minimization photometric error function can be constructed as follows:

$$T_{k,k-1} = \arg\min_{E_{k,k-1}} \frac{1}{2} \sum_{i\in\Omega} \left\| \delta I(T_{k,k-1}, u_i) \right\| \tag{3}$$

where the vector $I(u_i)$ represents the luminosity of the 4*4 pixel block around the key point at position ui, and δI(T,u) is the luminosity residual.

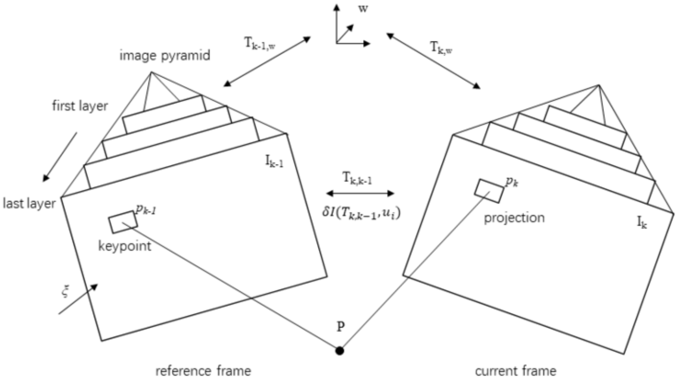


Figure 5. Schematic of the framework of the muti-layer sparse direct method

In order to avoid solving the Jacobi matrix iteratively, we use the idea of inverse compositional method by applying a perturbation ξ to the location of the keypoints, and due to the nonlinear nature of $T_{k,k-1}$, the calculation of the residuals is computed using the incremental update as follows:

$$\delta I(\xi, u_i) = I_k(\pi(\hat{T}_{k,k-1} \cdot p_i) - I_{k-1}(\pi(T(\xi) \cdot p_i))) \tag{4}$$

where δI(ξ,u) is a function of the photometric residuals with respect to the perturbations, which we will use as an intermediate quantity for the calculation of the Jacobi matrices, after which we base our calculation of the Jacobi matrices on the photometric information, according to the chain method as follows:

$$J = \frac{\partial I(\xi, u_i)}{\partial \xi} = \frac{\partial I_{k-1}(u_i)}{\partial u_i} \cdot \frac{\partial \pi(p_i)}{\partial p_i} \cdot \left. \frac{\partial T(\xi)}{\partial \xi} \right|_{\xi=0} \cdot p_{k-1} \tag{5}$$

We perform a combinatorial operation on each feature point pk-1 on frame k-1, as well as on the 4x4-sized pixel block in the upper-left corner of the feature point, and use Gaussian-Newton method on the computation of the perturbation ξ that makes the gradient of the photometric error function is 0 and iterates it as a new perturbation into the process of minimizing the photometric error. After obtaining the optimal ξ, we update the optimal interframe transformation by it. Here the iteratively optimized interframe transform Tk,(k-1)' is introduced as follows:

$$T_{k,(k-1)'} = T_{k,k-1} T_{k-1,(k-1)'} = T_{k,k-1} T(\xi)^{-1} \tag{6}$$

After iterative updating, the optimal inter-frame transformation matrix $T_{k,k-1}$ is obtained, and this inter-frame transformation minimizes the photometric error of the keypoints and surrounding blocks projected from the reference frame to the current frame.

The predicted camera pose is obtained by combining the camera pose $T_{k-1,w}$ solved from the previous frame, where w is the world coordinate system.

$$T_{k,w} = T_{k,k-1} T_{k-1,w} \tag{7}$$

Finally, we combine the method of Bundle Adjustment to choose an optimal camera position to minimize the reprojection error of the key point to its tracking position, k_p is set to denote the pixel p under the k-system (camera coordinate system), and the camera position is calculated as follows.

$$T_{k,w} = \arg\min_{T_{k,w}} \frac{1}{2} \sum \left\| u_i - \pi(T_{k,w}\ {}_k p_i) \right\|^2 \tag{8}$$

In this way we obtain an optimized current frame positional pose $T_{k,w}$ that enables the estimation of the camera motion.

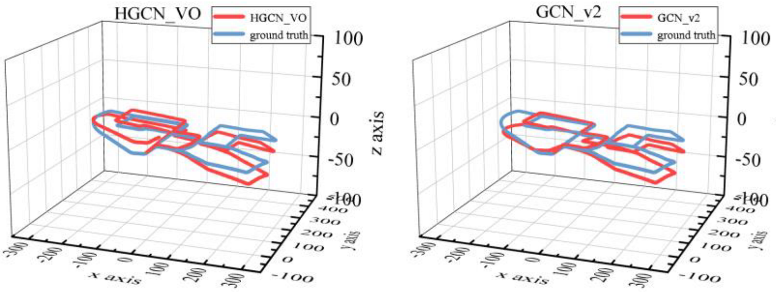
4. Experiment and Analysis

We compare our method with GCNv2, DSO and ORB_SLAM2 algorithms in some scenes of KITTI and TUM datasets. Finally, we analyze these methods in a practical scenario.

We only compared the predicted trajectory information and inference time of the algorithms without the modules of back-end optimization and loopback detection, and verified the performance of the visual odometry by the trajectory error as well as the operation speed. The performance of the visual odometer is verified by the trajectory error and running speed. To ensure fairness, we set the upper limit number of keypoints that can be extracted by these methods to 1500.

4.1. Comparison results on KITTI dataset

The KITTI dataset is a widely used public dataset for algorithm performance evaluation and experimental results validation of visual odometry, among which the large highway area in the highway scenario is a typical weakly textured scenario. In this paper, we use the KITTI dataset to validate the performance of the algorithm in outdoor environments.



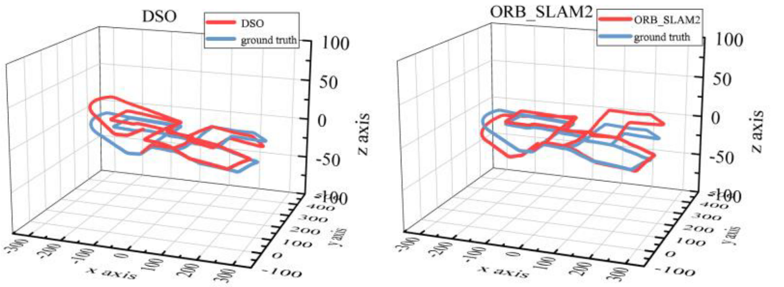


Figure 6. The trajectory results of HGCN-VO, GCN-v2, DSO and ORB-SLAM2 under KITTI-00 sequence.

The trajectory information of HGCN_VO, GCNv2, DSO, and ORB_SLAM2 under KITTI-00 sequence is shown in Figure 6. A comparison of the trajectory errors of each algorithm is shown in **Figure 7**, where err_* represents the trajectory position error of each axis and m_err* is the average position error of each axis. The trajectory error of the algorithm is gradually increasing because no back-end optimization is set in the comparison. With the mean position error we can find that HGCN_VO outperforms the other algorithms in terms of front-end position estimation performance. In addition, we simultaneously tested the results of these algorithms on other sequences of the kitti dataset, listing the visual odometry position root mean square error (RMSE) as well as the overall running time in **Table 1**. In addition, in order to verify the performance of the proposed method in avoiding tracking loss, when the algorithm has tracking loss, we pause and restart the visual odometry in the lost frame. We also list the tracking loss in the table to compare the tracking loss of the algorithm. Where "-" means no tracking lost.

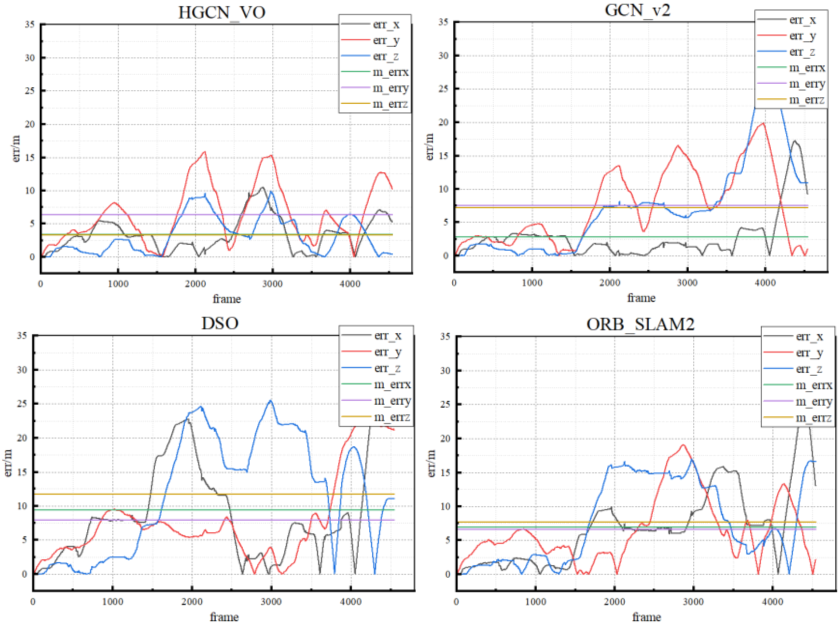


Figure 7. Comparison of 3-axis position error and average trajectory error of HGCN-VO, GCN-v2, DSO, ORB-SLAM2 under KITTI-00 sequence.

Table 1. Root mean square error of position and running time of each algorithm under KITTI dataset

Method	Sequence kitti	KITTI00	KITTI02	KITTI03	KITTI04	KITTI06
Ours(HGCN_VO)	RMSE	6.901	23.483	2.030	1.561	10.023
	time	40.869	44.280	6.492	4.650	17.531
	tracking lost	-	-	-	-	-
GCN v2	RMSE	8.234	29.818	2.737	1.961	11.558
	time	98.197	107.117	11.251	9.335	29.004
	tracking lost	-	1	1	-	-
DSO	RMSE	13.292	62.170	5.449	1.675	19.288
	time	78.918	101.824	9.325	5.691	25.455
	trackinglost	-	2	-	-	2
ORB SLAM2	RMSE	7.846	30.995	2.131	1.642	16.360
	time	151.539	164.912	16.605	9.756	37.698
	tracking lost	-	5	-	-	2

4.2. Comparison results on TUM dataset

The TUM dataset contains multi-sensor data collected from mobile devices, and the dataset is often accompanied by challenging scenarios such as dramatic viewpoint movement and low features. In this paper, we utilize the TUM dataset to study the performance of HGCN-VO in indoor environments. Our comparison setup is consistent with the section4.1.

The trajectory information of HGCN_VO with GCNv2, DSO and ORB_SLAM2 on fr1_room and the 3-axis atti-tude error information on the fr1_room sequence are shown in **Figure 8**., **Figure 9.**

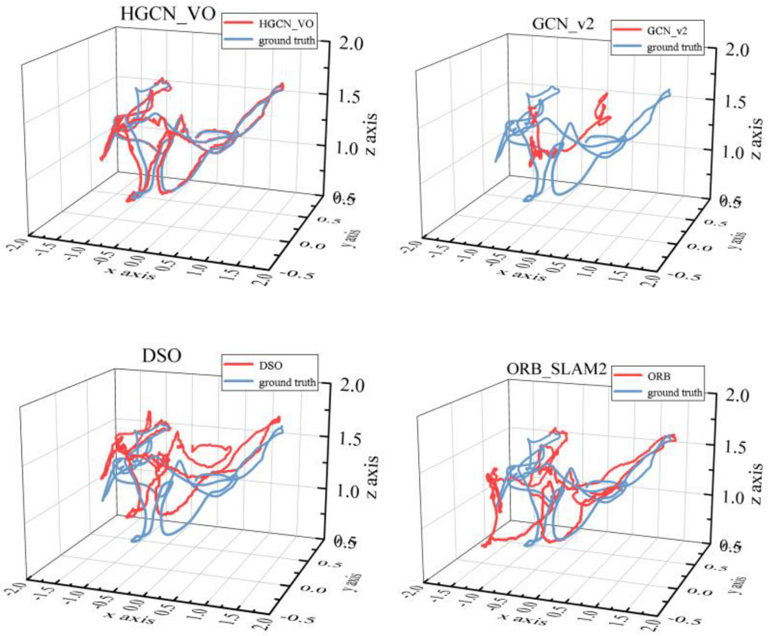


Figure 8. The trajectory results of HGCN-VO(a), GCN-v2(b), DSO(c) and ORB-SLAM2(d) under TUM fr1_room sequence.

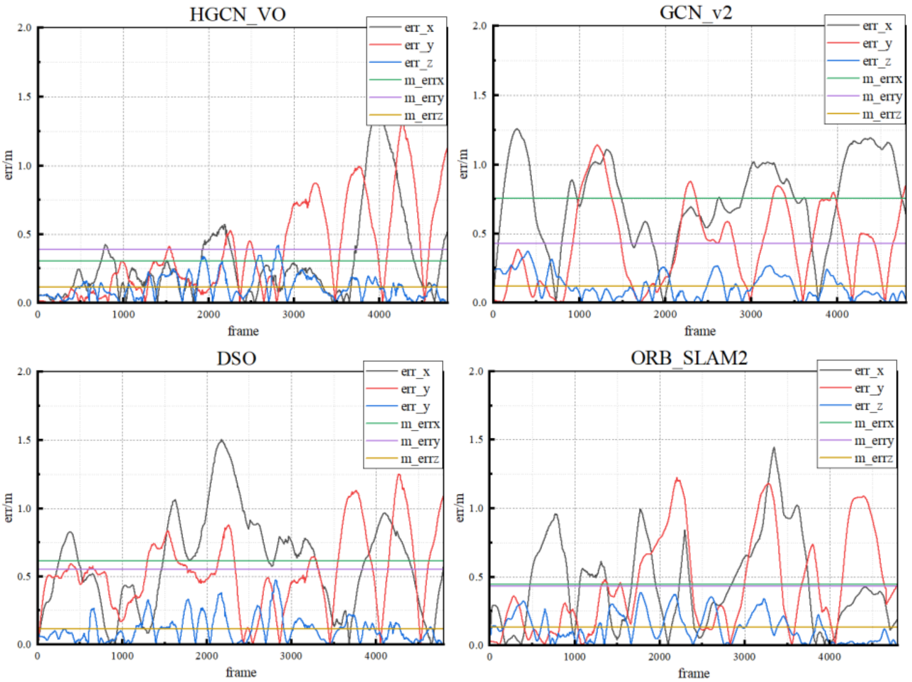


Figure 9. Comparison of 3-axis position error and average trajectory error of HGCN-VO, GCN-v2, DSO, ORB-SLAM2 under TUM fr1_room sequence

The fr1_room sequence has typical motion blur and low-feature scenarios, and the number of feature points in these scenarios decreases significantly, leading to loss of tracking by the feature-point method. the number of features extracted by the HGCN network is much larger and stable, and in addition the tracking is not prone to loss of matches using the direct method. To further validate the algorithm's localization effect in indoor weak texture environment under robot view, we also used the contents of fr2_pioneer_360, fr2_pioneer_slam2, and fr2_pioneer_slam3 sequences from TUM data, and listed the algorithm's RMSE of position in these sequences as well as the running time in **Table 2.**

Combining the trajectory images and statistical results, it can be concluded that HGCN_VO has an advantage over other visual odometry in terms of localization accuracy and stability in weakly textured scenarios in both indoor and outdoor datasets, which is partly due to the robust feature extraction and good training method of HGCN. In addition, other methods have feature loss in each scenario. Because the proposed method has a more stable key point maintenance method, the tracking loss on the data set is better than most of the algorithms. The method in this paper also outperforms other algorithms in terms of running speed, thanks to the feature extraction by efficient shallow neural networks and the camera position solving using the sparse direct method framework.

Table 2. Root mean square error of position and running time of each algorithm under TUM dataset

Method	Sequence TUM	fr1_room	fr2_pioneer_360	fr2_pioneer slam2	fr2_pioneer slam3
Ours(HGCN_VO)	RMSE	0.387	0.076	0.181	0.144
	time	21.465	20.665	35.456	43.121
	tracking lost	2	-	-	-

GCN v2	RMSE	0.892	0.093	0.169	0.912
	time	20.665	33.209	28.726	44.112
	tracking lost	2	1	1	2
DSO	RMSE	0.595	0.089	0.160	0.179
	time	31.189	28.726	50.690	57.647
	trackinglost	4	2	-	-
ORB SLAM2	RMSE	0.572	0.080	0.178	0.183
	time	44.360	44.112	83.165	89.040
	tracking lost	10	5	7	1

4.3. Results on Practical scenarios

We deploy HGCN_VO on a mobile robot **Figure 10.**(left) and verify its performance with real scenarios. The robot is equipped with the ORBBEC Astra sensor, which can directly obtain image depth information.

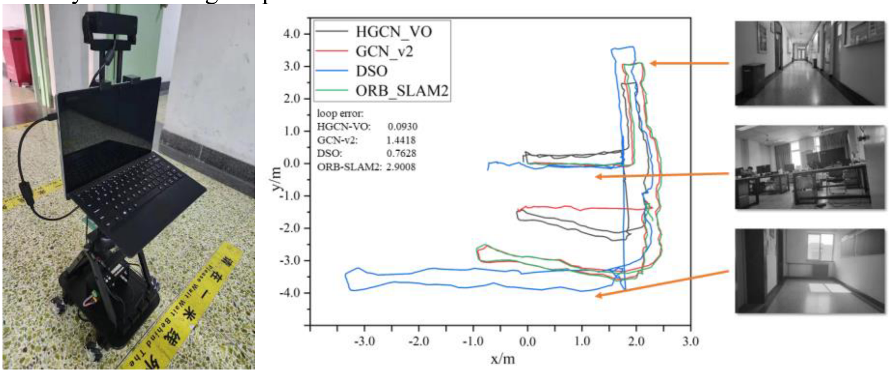


Figure 10. CV robot(left) and Trajectory results of HGCN-VO, GCNv2, DSO and ORB-SLAM2 in real scenes (right). actual scene: dim corridor (top), room (middle) and bright elevator room (bottom)

We chose a test environment that contains normally lit room, dimly lit corridor, and bright elevator room, with environments that contain variations in lighting conditions and weakly textured environments **Figure 10.**(right) to validate the algorithm's performance in challenging environments. Since obtaining real trajectory information in the 3D world is difficult, we use the method of setting the start and end points of the mobile robot at the same location, while turning off the loopback detection module in the comparison algorithms, and comparing the offset position of the end point of the trajectory movement with respect to the start point (loopback error), to estimate the performance of the odometry. If the loop error is smaller, the visual mileage calculation method has less positional error during operation.

The trajectory information obtained by each algorithm is shown in **Figure10**, and in the results we can find that the GCNv2 algorithm and the ORB-SLAM2 algorithm both appear tracking loss to a certain extent, resulting in inaccurate positioning, while the proposed algorithm is less prone to tracking loss due to its fast running speed and small gray changes in adjacent frames. It is proved that the proposed algorithm has great advantages in the actual scene positioning.

5. Conclusion

We propose a new keypoint extraction network and design an efficient learn-ing-model fusion visual odometry framework, HGCN-VO, which extracts robust key-points by a self-supervised convolutional neural network and uses the keypoints for solving the camera position by direct method. It is shown experimentally that HGCN-VO has higher accuracy and speed in both outdoor and indoor complex envi-ronments. Our odometry framework only contains front-end pose solving. Alhough it has been able to show excellent performance in localization, the use of the di-rect method makes it difficult to implement descriptor-based relocation and loopback detection, and it does not have map building capabilities. In the future, we will seek to implement back-end optimizations to reduce cumulative drift while constructing a complete SLAM framework. This process may incorporate other sensors.

References

[1] Wang Y, Chen H, Liu H, Zhang S. Edge-Based Monocular Thermal-Inertial Odometry in Visually Degraded Environments. IEEE Robotics and Automation Letters. 2023 Apr; 8(1): 2078-2085.

[2] Xu D, Zhang Z. Application and Analysis of Recurrent Convolutional Neural Network in Visual Odometry, 2022 IEEE International Conference on Mechatronics and Automation (ICMA); 2022, Aug 7-10 Guilin, Guangxi, China, p. 1222-1226.

[3] Zhi M, Deng C, Zhang H, Tang H, Wu J, Li B. RNGC-VIWO: Robust Neural Gyroscope Calibration Aided Visual-Inertial-Wheel Odometry for Autonomous Vehicle. Remote Sensing. 2023; 15(17):4292.

[4] Piao J, Kim S. Adaptive Monocular Visual–Inertial SLAM for Real-Time Augmented Reality Applications in Mobile Devices. Sensors 2017, 17, 2567.

[5] Lianos KN, Schnberger JL, Pollefeys M, Sattler T. VSO: Visual Semantic Odometry. European Conference on Computer Vision (ECCV), Munich, GERMANY, 09-08 September 2018.

[6] Zeng Q, Gao C, Chen Z, Jin Y and Kan Y. Robust Mono Visual-Inertial Odometry Using Sparse Optical Flow With Edge Detection, IEEE Sensors Journal. 2022 Mar 15; 22(6): 5260-5269.

[7] Wang S, Clark R, Wen H, Trigoni N. DeepVO: Towards end-to-end visual odometry with deep Recurrent Convolutional Neural Networks, 2017 IEEE International Conference on Robotics and Automation (ICRA); 2017 July 24; Singapore. p.2043-2050.

[8] Almalioglu Y, Saputra MRU, Gusmo PPBD, Markham A, Trigoni N. GANVO: Unsupervised Deep Monocular Visual Odometry and Depth Estimation with Generative Adversarial Networks. IEEE International Conference on Robotics and Automation (ICRA);2019 May 20-24 Montreal, CANADA: IEEE Press; c2019 p.5474-5480

[9] Wang S, Clark R, Wen H, Trigoni N. End-to-end, sequence-to-sequence probabilistic visual odometry through deep neural networks. The International journal of robotics research. 2018, 37(4-5): 513-542.

[10] Tang JX, Folkesson J, Jensfelt P. Geometric Correspondence Network for Camera Motion Estimation. IEEE Robotics & Automation Letters. 2018; 3(2): 1010-1017.

[11] Wei SE, Ramakrishna V, Kanade T, Sheikh Y. Convolutional Pose Machines. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2016 June 27-30; Seattle(WA): p.4724-4732.

[12] MurArtal R, Montiel JMM, Tardos JD. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. IEEE Transactions on Robotics, 2015; 31(5): 1147-1163.

[13] MurArtal R, Tardós JD. Orb-slam2: an open-source slam system for monocular, stereo, and rgb-d cameras. IEEE Trans. Robot. 2017; 33: 1255–1262.

[14] Campos C, Elvira R, Rodríguez JJG, Montiel JMM, Tardós JD. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM. IEEE Transactions on Robotics. 2020; 37: 1874-1890.

[15] Lin Y, Liu Z, Huang J, Wang C, Du G, Bai J, Lian S. Deep Global-Relative Networks for End-to-End 6-DoF Visual Localization and Odometry. 16th Pacific Rim International Conference on Artificial Intelligence (PRICAI);2018 Aug 26-30; Cuvu, FIJI.

[16] Bojanic D, Bartol K, Pribanic T, Petkovic T, Donoso Y, Mas J. On the Comparison of Classic and Deep Keypoint Detector and Descriptor Methods. 11th International Symposium on Image and Signal Processing and Analysis (ISPA); 2019 Sep 23-25; Dubrovnik, CROATIA.

[17] Dusmanu, M.; Rocco, I.; Pajdla, T.; Pollefeys, M.; Sivic, J.; Torii A.; Sattler T. D2-Net: A Trainable CNN for Joint Description and Detection of Local Features, 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); Long Beach, CA, USA, 16-20 June 2019.

[18] DeTone D, Malisiewicz T, Rabinovich A. SuperPoint: Self-Supervised Interest Point Detection and Description. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW); 2018 June 18-22; Salt Lake City, UT, USA.

[19] Tang J, Ericson L, Folkesson J, Jensfelt P. GCNv2: Efficient Correspondence Prediction for Real-Time SLAM. IEEE Robotics and Automation Letters. 2019; 4(4): 3502-3512.

[20] Engel J, Schps T, Cremers D. LSD-SLAM: Large-scale direct monocular SLAM. 13th European Conference on Computer Vision (ECCV); 2014 Sep 06-12; Zurich, SWITZERLAND.

[21] Forster C, Pizzoli M, Scaramuzza D. SVO: Fast semi-direct monocular visual odometry. 2014 IEEE International Conference on Robotics and Automation (ICRA);2014 May 31-June 07; Hong Kong, CHINA.

[22] Loo SY, Amiri AJ, Mashohor S, Amiri AJ, Mashohor S, Tang SH. CNN-SVO: Improving the Mapping in Semi-Direct Visual Odometry Using Single-Image Depth Prediction. 2019 International Conference on Robotics and Automation (ICRA); 2019 May 20-24; Montreal, CANADA: IEEE Press.

[23] Li R, Wang S, Long Z, Gu D. UnDeepVO: Monocular Visual Odometry Through Unsupervised Deep Learning. 2018 IEEE International Conference on Robotics and Automation (ICRA); 2018 May 21-25; Brisbane, Australia.

[24] Radwan N, Valada A, Burgard W. VLocNet++: Deep Multitask Learning for Semantic Visual Localization and Odometry. IEEE Robotics and Automation Letters. 2018; 3(4), 4407-4414.

[25] Yu L, Yang E, Yang B, Fei Z, Niu C. A Robust Learned Feature-Based Visual Odometry System for UAV Pose Estimation in Challenging Indoor Environments. IEEE Transactions on Instrumentation and Measurement. 2023; 72: 1-11.

[26] Yang N, von-Stumberg L, Wang R, Cremers D. D3VO: Deep Depth, Deep Pose and Deep Uncertainty for Monocular Visual Odometry. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2020 June 14-19; ELECTR NETWORK.

[27] Iman AK, Luke F, Gerard D, Daniel T. A survey of state-of-the-art on visual SLAM. Expert Systems with Applications. 2022; 205: 117734.

[28] Zhang X, Zhao B, Yao J, Wu G. Unsupervised Monocular Depth and Camera Pose Estimation with Multiple Masks and Geometric Consistency Constraints. Sensors. 2023; 23: 5329.

[29] Cao Y, Zhang X, Luo F, Peng P, Lin C, Yang K, Li Y. Learning generalized visual odometry using position-aware optical flow and geometric bundle adjustment. Pattern Recognition. 2023; 136: 109262.