Postprint of Homography Estimation Method Based on Multi-scale Residual Network

Authors: Tang Yun, Shuai Pengfei, Jiang Peifan, Deng Fei, Yang Qiang

Date: 2022-06-06T00:00:00+00:00

Abstract

Homography estimation is a fundamental and crucial step in many computer vision tasks. Traditional homography estimation methods based on feature point matching perform poorly on weakly textured images. Deep learning has been applied to homography estimation to improve its robustness, but existing methods have not taken into account the multi-scale problem caused by object scale differences, thus limiting their accuracy. To address these issues, we propose a multi-scale residual network for homography estimation. This network is capable of extracting multi-scale feature information from images and employs a multi-scale feature fusion module to effectively fuse features. Furthermore, it reduces the network optimization difficulty by estimating normalized offsets of the four corner points. Experimental results demonstrate that on the MS-COCO dataset, the method achieves an average corner error of only 0.788 pixels, reaching sub-pixel accuracy, and maintains high accuracy in 99% of cases. Due to the comprehensive utilization of multi-scale feature information and easier optimization, this method significantly improves accuracy and exhibits stronger robustness.

Full Text

Preamble

Vol. 39 No. 10
Application Research of Computers
Accepted Paper

Homography Estimation Method Based on Multi-Scale Residual Network

Tang Yun[1], Shuai Pengfei[1]†, Jiang Peifan[1], Deng Fei[1], Yang Qiang[1],[2]
[1] College of Computer & Network Security (Oxford Brookes College), Chengdu

University of Technology, Chengdu 610059, China
[2] College of Control Engineering, Chengdu University of Information Technology, Chengdu 610225, China

Abstract: Homography estimation is a fundamental and crucial step in many computer vision tasks. Traditional homography estimation methods rely on feature point matching, which struggle with weakly textured images. Deep learning has been applied to homography estimation to improve robustness, but existing methods fail to address the multi-scale problem caused by object scale differences, limiting their accuracy. To address these issues, this paper proposes a multi-scale residual network for homography estimation. The network extracts multi-scale feature information from images and employs a multi-scale feature fusion module for effective feature integration. Additionally, it reduces network optimization difficulty by estimating four-corner normalized offsets. Experiments on the MS-COCO dataset demonstrate that the proposed method achieves an average corner error of only 0.788 pixels, attaining sub-pixel accuracy while maintaining high precision in 99% of cases. By comprehensively leveraging multi-scale feature information and enabling easier optimization, the proposed method significantly improves accuracy and exhibits stronger robustness.

Keywords: homography estimation; multi-scale residual network; feature fusion; four-corner normalized offset; average corner error

0 Introduction

Homography refers to an invertible mapping from one plane to another, which can be represented by a 3$×$3 non-singular matrix encompassing translation, scaling, rotation, and perspective transformations, known as the homography matrix [**?**]. Given two images, estimating the homographic transformation between them is a common requirement in computer vision. Homography estimation has broad applications and serves as foundational work in tasks such as image registration [**?**], image stitching [**?**], image rectification [**?**], 3D reconstruction [**?**], and SLAM [**?**], where its accuracy critically impacts overall performance.

Traditional homography estimation methods typically rely on feature point matching. They extract feature points using algorithms like SIFT [**?**], SURF [**?**], or ORB [**?**], establish correspondences through brute-force matching or FLANN [**?**], and finally solve for the homography matrix after outlier rejection using RANSAC [**?**]. However, these methods heavily depend on the quantity and distribution of feature points, making them unsuitable for weakly textured images. Moreover, the process is cumbersome, requiring manual specification of numerous hyperparameters [**?**].

With the rise of deep learning, several deep learning-based homography estima-

tion methods have been proposed. In 2016, DeTone et al. [**?**] first introduced a VGG-based network for homography estimation, demonstrating the potential of deep learning approaches. In 2017, Nowruzi et al. [**?**] employed a hierarchically stacked network that progressively refined estimates by stacking multiple identical network modules. Nguyen et al. [**?**] proposed an unsupervised learning method for homography estimation. In 2020, Zhang et al. [**?**] used ResNet as the backbone with content masks to select reliable regions for estimation. While these methods achieved certain results, they all neglected the multi-scale nature of homography estimation. Due to differences in camera position, distance, and angle between shots, the same object may appear at different scales in two images—a factor ignored by existing network models that employ single-scale features, resulting in inherent limitations.

To address the multi-scale problem in homography estimation and inspired by SKNet' s [**?**] multi-scale feature fusion approach, this paper proposes a Multi-scale Residual Homography Estimation Network (MRHENet). The network' s key innovations include: (a) using convolutional layers with different receptive fields to extract multi-scale features for homography estimation; (b) proposing a Multi-Scale Feature Fusion Module (MFF Module) for effective multi-scale feature integration; and (c) estimating four-corner normalized offsets rather than absolute pixel offsets to further reduce optimization difficulty. Experimental results on the MS-COCO [**?**] and Apolloscape [**?**] datasets demonstrate the proposed method' s superiority. On MS-COCO, the average corner error is only 0.788 pixels, achieving sub-pixel accuracy while maintaining high precision in 99% of cases. By comprehensively utilizing multi-scale features and enabling easier optimization, the method significantly improves accuracy and robustness.

1.1 Principle of Traditional Homography Estimation Methods

Assuming a pair of images A and B are captured from the same planar object through a pinhole camera model, they exhibit a homographic transformation relationship. This relationship can be expressed using a 3$×$3 non-singular homography matrix H. According to the definition of homography [**?**], the transformation is given by equation (1):

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = H \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

The homography matrix H maps point (x, y) in image A to point (x', y') in image B. Transforming equation (1) yields two linear equations:

$$x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}}, \quad y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}}$$

In matrix H, h_{33} is a non-zero scaling coefficient typically set to 1, leaving eight degrees of freedom. Each matching point pair provides two linear equations via equation (2), requiring a minimum of four point pairs to solve for the homography matrix, with the constraint that no three points from the same image are collinear [**?**].

The homography matrix solution method is shown in equation (3), where Corners_A and Corners_B represent the coordinates of matched feature points, and n denotes the number of matching pairs. With fewer than 4 pairs, the matrix cannot be solved; with exactly 4 pairs, Direct Linear Transformation (DLT) can be used; with more than 4 pairs, Least Squares (LS) can be applied.

$$H = f(\text{Corners}_A, \text{Corners}_B, n) = \begin{cases} \text{unsolvable}, & n < 4 \\ \text{DLT}, & n = 4 \\ \text{LS}, & n > 4 \end{cases}$$

Traditional homography estimation proceeds as follows: (a) extract feature points from both images using detection algorithms; (b) establish correspondences between the two point sets using matching algorithms; and (c) solve for the homography matrix based on these correspondences. Numerous studies have addressed feature detection: SIFT [**?**] offers high matching accuracy but high computational complexity; SURF [**?**] improves SIFT' s speed; ORB [**?**] provides higher efficiency but lower quality than SIFT. Matching can use brute-force or FLANN [**?**]. Due to potential mismatches, RANSAC [**?**] is required to exclude outliers during homography matrix computation.

Traditional methods depend heavily on feature point quality and distribution. In practice, achieving ideal accuracy requires slower feature detection algorithms, and weakly textured images often lack sufficient matching pairs, leading to large errors or unsolvable cases. Consequently, traditional methods exhibit weak robustness and numerous practical limitations.

1.2 Principle of Deep Learning-Based Homography Estimation

Deep learning-based homography estimation uses deep networks to estimate the homographic transformation between two input images. As illustrated in Figure 1 [Figure 1: see original paper], given image pair A (source) and B (target), where B is generated by applying homography H to A, the deep learning approach: (1) preprocesses images A and B, (2) inputs them into a network

to estimate the transformation in some representation, and (3) computes the estimated homography matrix H^* (where * denotes estimation).

Homography can be represented in multiple forms: directly as a homography matrix, as four-corner absolute pixel offsets [**?**], or other representations. Since homography matrix elements have different meanings and ranges—for instance, $h_{11}, h_{12}, h_{21}, h_{22}$ represent rotation while h_{13}, h_{23} represent translation, with translation values typically much larger than rotation values—direct estimation is difficult without normalization. Therefore, [**?**] parameterizes the homography matrix into four-corner absolute pixel offsets, estimating these offsets to obtain four matching point pairs, then solving for the homography matrix using DLT from equation (3).

Compared to traditional methods, deep learning approaches offer advantages in speed and robustness. Traditional methods are slow due to feature detection and matching, and struggle with weakly textured images where stable matches are unavailable. Deep learning methods bypass detection and matching, enabling faster processing. For weakly textured images that traditional methods cannot handle, deep learning can learn patterns from extensive training data to estimate reasonable homography matrices. Thus, deep learning homography estimation faces fewer practical constraints, exhibits stronger robustness, and holds significant application value.

In 2016, [**?**] first applied a VGG-based network to homography estimation, but limited improvements due to its simple, shallow architecture. As traditional CNNs deepen, they may suffer from degradation and training difficulties. Consequently, He et al. [**?**] proposed ResNet in 2016, using identity mapping to ease deep network training. In 2020, [**?**] used ResNet34 as the backbone with content masks for homography estimation, achieving improved results over prior work. However, these methods all ignored the multi-scale problem in homography estimation, creating inherent limitations.

In homography estimation, differences in camera position, distance, and angle between shots introduce distortion and scaling, causing the same object to appear at different scales in two images. This presents a multi-scale challenge. To address this, we propose a multi-scale residual homography estimation network that integrates multi-scale feature information to estimate four-corner normalized offsets. The network architecture is shown in Figure 2 [Figure 2: see original paper].

The network takes two normalized 128$\times$128 $gray scale images as input. The output is a 4\times$2 matrix $H^*_{4pt\_norm}$ representing four-corner normalized offsets. The computation process: First, the two images are normalized, stacked into two channels, and fed into three feature extraction branches for large, medium, and small-scale features. The medium and small-scale branches include additional stride-2 convolutional layers for feature map reduction. After ReLU activation, large-scale features enter ResNet34' s stage1 block. The stage1 output and medium-scale features are fused via an MFF module

(scaling factor $r = 2$) and fed into stage2. The stage2 output and small-scale features undergo another MFF module ($r = 4$) before passing through stage3 and stage4 blocks. Finally, the feature map is average-pooled to $1\$\times 1\times 512 and passed through a fully connected layer to output the 4\times\$2$ matrix $H^*_{4pt\_norm}$. BatchNorm layers [**?**] are used after each convolutional layer to accelerate training.

2.3 Multi-Scale Feature Fusion

In homography estimation, camera position, distance, and angle differences between shots cause distortion and scaling, making the same object appear at different scales in two images. This presents a multi-scale challenge. However, [**?**]-[**?**] ignore this issue, treating both images as having the same scale and using single-scale convolutional layers for feature extraction. A single convolutional kernel has a fixed receptive field, extracting features at only one spatial scale. While subsequent layers and activation functions gradually aggregate these into deep semantic features with larger receptive fields, the original spatial and geometric details are lost [**?**]. Therefore, using single-scale features for homography estimation has limitations, particularly when images have large scale differences. Multi-scale feature information is crucial for homography estimation.

Our network introduces multi-scale feature information to address scale inconsistency and improve accuracy, ensuring effective performance even with large scale differences. As shown in Figure 2, the network has three branches (large, medium, small) to extract corresponding scale features for homography estimation. Specifically, each branch uses dilated convolutional layers [**?**] with receptive fields of $7\$\times 7, 5\times 5, and 3\times\$3$ to extract multi-scale features. Figure 3 [Figure 3: see original paper] illustrates dilated convolution: compared to standard convolution, it maintains receptive field size while reducing parameters and computation, improving efficiency.

Original ResNet34 [**?**] uses max pooling for downsampling, which discards non-maximum features. We avoid max pooling by setting the first convolutional layer in stage1 to stride 2 (originally stride 1), preventing information loss while reducing computation. Since MFF modules require same-shaped feature maps, the medium and small-scale branches use 1 and 2 layers of $2\$\times\$2$ stride-2 convolutions respectively for downsampling to match subsequent MFF modules and enhance cross-channel feature communication.

In convolution-based homography estimation networks, features evolve from shallow to deep layers. Shallow features have higher resolution with more positional and geometric details but lower semantics, while deep features have stronger semantics but poor detail perception. Effectively leveraging both is key to improving accuracy.

Therefore, rather than fusing all three scales initially, our network progressively

integrates medium and small-scale features after stage1 and stage2 blocks. This gradual fusion uses shallow feature details to complement deep features, achieving complementary advantages. Direct addition of multi-scale features would cause mixing and underutilization, while channel concatenation would double channels and drastically reduce efficiency. Since features from different scales originate from the same input, redundancy exists between them. To fully utilize multi-scale features while reducing redundancy and improving efficiency, inspired by [**?**], we propose the MFF Module for multi-scale feature fusion.

The MFF module structure is shown in Figure 4 [Figure 4: see original paper]. It takes two different-scale feature maps $x_1, x_2 \in \mathbb{R}^{H\times W\times C}$ and outputs a fused feature map $x_{out} \in \mathbb{R}^{H\times W\times C}$. Unlike [**?**], which directly adds features before 1\$×\$1 average pooling, our approach differs in two aspects: First, we concatenate x_1 and x_2 in the channel dimension to preserve individual features for subsequent channel-wise extraction. Second, we use both $1\$\times 1 average pooling and 1\times\$1$ max pooling to extract channel information, as average pooling captures global information while max pooling captures local information, enabling comprehensive global-local feature extraction.

The MFF module computation proceeds as follows:

a) Concatenate x_1 and x_2 to obtain $x_{cat} \in \mathbb{R}^{H\times W\times 2C}$. Apply $1\$\times 1 average pooling and 1\times\$1$ max pooling to x_{cat}, sum the results to obtain $x_g \in \mathbb{R}^{1\times 1\times 2C}$:

$$x_g = \text{AvgPool}(x_{cat}) + \text{MaxPool}(x_{cat})$$

b) Use a fully connected layer fc_0 with C/r nodes (where r is the scaling factor) to compress x_g for efficiency, followed by ReLU to obtain $z_r \in \mathbb{R}^{1\times 1\times C/r}$. Then pass z_r through two separate fully connected layers fc_1 and fc_2 with C nodes each to get $z_1, z_2 \in \mathbb{R}^{1\times 1\times C}$:

$$z_r = \text{ReLU}(fc_0(x_g)), \quad z_1 = fc_1(z_r), \quad z_2 = fc_2(z_r)$$

c) Stack z_1 and z_2 in the channel dimension and apply SoftMax to obtain channel-wise weights $w_1, w_2 \in \mathbb{R}^{1\times 1\times C}$ for the input feature maps:

$$[w_1, w_2] = \text{SoftMax}([z_1, z_2])$$

d) Finally, multiply x_1 and x_2 with w_1 and w_2 via broadcasting, sum the results to obtain the fused feature map:

$$x_{out} = w_1 \cdot x_1 + w_2 \cdot x_2$$

Unlike simple feature addition, the MFF module allocates appropriate weights to different-scale feature maps based on comprehensive channel-wise global information, enabling the network to select suitable scale features for homography estimation. This preserves effective features while reducing redundancy,

facilitating full utilization of multi-scale information and improving estimation accuracy.

2.4 Four-Corner Normalized Offset

To address optimization difficulties from direct homography matrix estimation, [**?**] parameterized the homography into four-corner absolute pixel offsets H_{4pt}, reducing optimization difficulty to some extent. However, absolute pixel offsets still exhibit large numerical variations, causing significant gradient differences during optimization. Additionally, network weights are typically initialized between -1.0 and 1.0, while absolute pixel offsets often far exceed 1 pixel, requiring substantial weight changes from initial values and hindering convergence.

To further reduce optimization difficulty, we estimate four-corner normalized offsets H_{4pt_norm}, computed as:

$$H_{4pt\_norm} = \begin{bmatrix} \frac{\Delta x_1}{W} & \frac{\Delta y_1}{H} \\ \frac{\Delta x_2}{W} & \frac{\Delta y_2}{H} \\ \frac{\Delta x_3}{W} & \frac{\Delta y_3}{H} \\ \frac{\Delta x_4}{W} & \frac{\Delta y_4}{H} \end{bmatrix}$$

where Δx_i and Δy_i $(i = 1, 2, 3, 4)$ represent normalized offsets in width and height directions for the i-th corner point (ordered clockwise from the origin), and W and H are image width and height. The conversion from estimated normalized offsets $H^*_{4pt\_norm}$ to homography matrix H^* is:

$$\text{Corners}_B = \text{Corners}_A + \begin{bmatrix} W & 0 \\ 0 & H \end{bmatrix} \cdot H^*_{4pt\_norm}$$

$$H^* = \text{DLT}(\text{Corners}_A, \text{Corners}_B)$$

3.1 Network Training

We use the MS-COCO [**?**] and Apolloscape [**?**] datasets, generating experimental data following [**?**]' s method but without scaling images to $320\$\times 240, which would force the network to learn from fewer features and reduce robustness. Images are normali$ with maximum corner offset $\rho = 32$ pixels (one-quarter of image size), using 180,000 pairs for training and 40,000 for validation.

The loss function is Average Corner Error (ACE) [**?**], representing the average Euclidean distance between predicted and ground-truth corner offsets in pixels:

$$\text{ACE} = \frac{1}{4}\sum_{i=1}^{4}\sqrt{(\Delta x_i - \Delta x_i^*)^2 + (\Delta y_i - \Delta y_i^*)^2}$$

Experiments are implemented in PyTorch. Training uses random flipping (probability 0.5) for data augmentation, Adam optimizer with L2 weight decay 0.003, batch size 256, initial learning rate 0.0002, decayed by 0.7 every 20K iterations, for 200K total iterations.

3.2 Experimental Testing

To evaluate performance, we generated 40,000 image pairs each for maximum corner offsets ρ = 8px, 16px, 24px, and 32px, totaling 160,000 test pairs. Smaller ρ values represent minor offsets, while ρ = 32px represents large offsets, providing a comprehensive test set with varying distortion levels.

During testing, ACE may occasionally exhibit extreme values, inflating the mean average corner error (Mean-ACE). Traditional methods may also fail due to insufficient feature points. Therefore, we cap ACE at 32px for cases where ACE > 32px or traditional methods fail. For 128$×$128 images, ACE > 32px indicates virtually worthless results, making 32px an appropriate threshold. Since Mean-ACE only reflects average error without showing distribution, we also report median ACE (Median-ACE) and invalid rate (IR)—the proportion of invalid cases (ACE > 32px or method failure). All methods are tested multiple times to avoid 偶然性.

Ablation Study: To validate each proposed improvement, we conducted ablation experiments on MS-COCO [**?**] with identical training/testing protocols. Table 1 shows results where "MFE"denotes multi-scale feature extraction, "MFF" denotes MFF module usage, and "Norm" denotes four-corner normalized offsets. Results show that MFE or Norm alone improves performance, and MFF module further enhances results. Combining all three yields the best performance.

Comparison Experiments: The final model is compared against traditional methods (SIFT [**?**]+RANSAC [**?**], ORB [**?**]+RANSAC [**?**]) and deep learning methods [**?**, **?**, **?**] on MS-COCO [**?**] and Apolloscape [**?**] datasets (Tables 2 -3). Since Apolloscape images have weaker textures than MS-COCO, all methods show increased errors. However, traditional methods exhibit dramatically larger errors and invalid rates on weaker textures, making them impractical. Deep learning methods show smaller error increases, confirming their stronger robustness.

Analysis focuses on MS-COCO results. Figures 5 [Figure 5: see original paper] and 6 [Figure 6: see original paper] show Mean-ACE and Median-ACE across different offset levels. As ρ increases from 8px to 32px, all methods' errors increase, but ours increases most gradually, maintaining sub-pixel accuracy

throughout—unique among all methods. While deep learning methods [**?**, **?**, **?**] have lower Mean-ACE than SIFT+RANSAC on MS-COCO, their Median-ACE is higher. Our method outperforms SIFT+RANSAC in both metrics.

Figure 7 [Figure 7: see original paper] shows ACE cumulative distribution for large offsets ($\rho = 32$px). Traditional ORB+RANSAC performs poorly with high error and 49.32% invalid rate. SIFT+RANSAC works well in ~70% of cases (ACE $<$ 4px) but fails catastrophically in the remaining 30% (20.1% invalid rate). Deep learning methods work in >99% of cases (ACE $<$ 32px), but [**?**, **?**, **?**] have higher error than SIFT+RANSAC in >60% of cases. Our method achieves lower error than SIFT+RANSAC in most cases and maintains high precision (ACE $<$ 4px) in 99% of cases, demonstrating the best robustness.

Table 4 compares performance: our model is smaller than [**?**, **?**] and significantly faster than traditional methods, with speed comparable to [**?**].

3.3 Effect Demonstration

Figure 8 [Figure 8: see original paper] visualizes homography estimation results. The leftmost column shows input image pairs. Blue and red boxes indicate original image positions; green boxes show estimation results. The average distance between red and green box corners is the ACE error from Section 3.1—smaller distances indicate better performance. Errors are displayed below each image; "fail" indicates method failure. SIFT+RANSAC and ORB+RANSAC nearly fail on weakly textured images, while our method consistently maintains low error.

---

4 Conclusion

Homography estimation is a fundamental step in computer vision tasks like image stitching and rectification, with broad applications where improved accuracy is significant. Traditional feature-matching methods struggle with weakly textured images. Existing deep learning methods neglect multi-scale characteristics, using single-scale features to estimate absolute pixel offsets, resulting in poor performance for large offsets. This paper proposes a multi-scale residual homography estimation network that extracts and fuses multi-scale features using the MFF module, effectively combining shallow and deep feature advantages while estimating four-corner normalized offsets to reduce optimization difficulty. Experiments on multiple datasets demonstrate significantly improved accuracy and robustness compared to traditional and deep learning methods, offering substantial practical value.

References

[?] Muja M, Lowe D G. Fast approximate nearest neighbors with automatic algorithm configuration [J]. VISAPP (1), 2009, 2 (331-340): 2.

[?] Fischler M A, Bolles R C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography [J]. Communications of the ACM, 1981, 24 (6): 381-395.

[?] DeTone D, Malisiewicz T, Rabinovich A. Deep image homography estimation [EB/OL]. (2016) [2022-03-13]. https://arxiv.org/abs/1606.

[?] Nowruzi F E, Laganiere R, Japkowicz N. Homography estimation from image pairs with hierarchical convolutional networks [C]// Proc of IEEE International Conference on Computer Vision. Piscataway, NJ: IEEE Press. 2017: 913-920.

[?] Nguyen T, Chen S W, Shivakumar S S, et al. Unsupervised deep homography: A fast and robust homography estimation model [J]. IEEE Robotics and Automation Letters, 2018, 3 (3): 2346-2353.

[?] Zhang Jirong, Wang Chuan, Liu Shuaicheng, et al. Content-aware unsupervised deep homography estimation [C]// Proc of European Conference on Computer Vision. Cham: Springer, 2020: 653-669.

[?] Hartley R, Zisserman A. Multiple view geometry in computer vision [M]. 2nd ed. Cambridge University Press. 2004: 25-48.

[?] Li Xiang, Wang Wenhai, Hu Xiaolin, et al. Selective kernel networks [C]// Proc of IEEE Conference on Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE Press, 2019: 510-519.

[?] Lin T Y, Maire M, Belongie S, et al. Microsoft COCO: Common objects in context [C]// Proc of European Conference on Computer Vision. Cham: Springer, 2014: 740-755.

[?] Huang Xinyu, Wang Peng, Cheng Xinjing, et al. The ApolloScape Open Dataset for Autonomous Driving and its Application [C]// Proc of IEEE Transactions on Pattern Analysis and Machine Intelligence. Piscataway, NJ: IEEE, 2020: 2702-2719.

[?] He Kaiming, Zhang Xiangyu, Ren Shaoqing, et al. Deep residual learning for image recognition [C]// Proc of IEEE Conference on Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE Press, 2016: 770-778.

[?] Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift [C]// Proc of International Conference on Machine Learning. New York: ACM Press, 2015: 448-456.

[?] 姚铭, 邓红卫, 付文丽, 等. 一种改进的 Mask R-CNN 的图像实例分割算法 [J]. 软件, 2021, 42 (09): 78-82. (Yao Ming, Deng Hongwei, Fu Wenli, et al. An Improved Mask R-CNN Image Instance Segmentation Algorithm [J]. Computer Engineering & Software, 2021, 42 (09): 78-82.)

[?] Yu F, Koltun V. Multi-scale context aggregation by dilated convolutions [EB/OL]. (2015) [2022-03-13]. https://arxiv.org/abs/1511.07122.

[?] Lowe D G. Distinctive image features from scale-invariant keypoints [J]. International Journal of Computer Vision, 2004, 60 (2): 91-110.

[?] Bay H, Tuytelaars T, Van Gool L. SURF: Speeded up robust features [C]// Proc of European Conference on Computer Vision. Berlin: Springer, 2006: 404-417.

[?] Rublee E, Rabaud V, Konolige K, et al. ORB: An efficient alternative to SIFT or SURF [C]// Proc of IEEE International Conference on Computer Vision. Piscataway, NJ: IEEE Press, 2011: 2564-2571.

[?] 夏丹, 周睿. 视差图像配准技术研究综述 [J]. 计算机工程与应用, 2021, 57 (02): 18-27. (Xia Dan, Zhou Rui. Survey of Parallax Image Registration Technology [J]. Computer Engineering and Applications, 2021, 57 (02): 18-27.)

[?] Brown M, Lowe D G. Recognising panoramas [C]// Proc of IEEE International Conference on Computer Vision. Piscataway, NJ: IEEE Press, 2003: 1218.

[?] Yang Xieliu, Yin Chenyu, Tian Dake, et al. Rule-based perspective rectification for Chinese text in natural scene images [J]. Multimedia Tools and Applications, 2021, 80 (12): 18243-18262.

[?] Zhang Zhongfei, Hanson A R. 3D reconstruction based on homography mapping [J]. Proc. ARPA96, 1996: 1007-1012.

[?] Davison A J, Reid I D, Molton N D, et al. MonoSLAM: Real-time single camera SLAM [J]. IEEE Trans on Pattern Analysis and Machine Intelligence, 2007, 29 (6): 1052-1067.

Note: Figure translations are in progress. See original paper for figures.