Vision-Based Pose Optimization Using Learned Metrics

SHAOPENG LI,1,2, YONG XIAN2, TAO ZHANG1, (Senior Member, IEEE), BANGJIE LI2, DAQIAO ZHANG2, AND WEILIN GUO2

1Department of Automation, Tsinghua University, Beijing 100084, China
2High-Tech Institute of Xi’an, Xi’an 710025, China

Corresponding authors: Tao Zhang (taozhang@tsinghua.edu.cn) and Daqiao Zhang (yamazz@163.com)

ABSTRACT Camera pose optimization is the basis of geometric vision works, such as 3D reconstruction, structure from motion, and visual odometry. A pose optimization method based on learned metrics is proposed to improve the optimization convexity. The neural network was designed and trained based on the collected datasets, respectively. The network inputs pairwise patches and outputs the Euclidean distance of its center. This distance is involved in the residual calculation of Gauss-Newton, and the Jacobian corresponding to this distance can be analytically solved. The simulation verified convergence and generalization of the designed network. The accuracy and robustness of the proposed pose optimization compared with intensity- and feature-based optimizations are also verified.

INDEX TERMS Geometric vision, pose optimization, metric learning, residual network.

I. INTRODUCTION Vision-based pose optimization is used to recover the camera-to-world translation and orientation from an image, which is a key technology in geometric vision. It is a process of predicting the transformation of an object from a user-defined reference pose, given an image. It arises in computer vision or robotics where the pose or transformation of an object can be used for alignment of a Computer-Aided Design models, identification, grasping, or manipulation of the object. It is used extensively in visual odometry (e.g., [1]), wide-baseline stereo (e.g., [2]), three-dimensional reconstruction (e.g., [3]), and structure from motion (e.g., [4]). The crucial component of pose optimization is the objective design and corresponding Jacobian calculation. The similarity metrics between images participate in the loss and gradient calculations, which determines the optimization convergence.

The camera pose is usually estimated by a feature- or intensity-based method. The feature-based method accomplishes pose optimization by minimizing the alignment error between corresponding features. Feature extraction and correspondence establishment are generally conducted by cumbersome engineering methods, such as SIFT [5], SURF [6], ORB [7]. However, in extreme environments, textureless or gradient texture scenes make features difficult to extract, and repeated textures can easily cause mismatches. The camera pose must be also estimated without feature extraction and matching [8].

By contrast, the intensity-based method optimizes relative pose by minimizing photometric errors [9]. It directly measures the similarity of pixel patches from the target and reference images via similarity metrics, such as sum of squared differences (SSD) and sum of absolute difference (SAD) [10]. The intensity-based method avoids feature extraction and matching. It works as long as the gray of pixels in image differs.

However, using the similarity metric, such as SSD, to optimize relative pose depends on the following correlation: the relative pose gradually approaches the true value as the similarity score increases. This correlation determines the convexity of the optimization model. Unfortunately, this positive correlation is weak, especially when the current value is far from the true value. Therefore, the intensity-based odometry, such as LSD [9] and DSO [11], requires good initial values, and large sampling is also needed to maintain this positive correlation for ensuring the convergence of pose optimization. In addition, the intensity-based method is based on luminosity invariance, which is susceptible to external illumination. Therefore, a good metric must be designed to enhance the optimization model convexity and robustness with respect to factors, such as illumination and initial values, when feature correspondences are unknown.
This study intends to design a metric that can directly calculate a metric from the pairwise patches, one of which is extracted from the reference frame and the other is the projected patch in the target frame. This metric do not need feature correspondence but maintain the optimization model convexity and robustness to the initial pose value. To achieve this goal, we constructed the neural network with the patches in the reference and target frames as inputs and the Euclidean distance ($\delta u, \delta v$) in the pixel coordinates as output. After training, the network can learn the Euclidean distance of the patch center. The learned metric was then applied to pose optimization to improve its convergence and robustness. The work carried out around this topic is summarized as follows:

- The regression and improved residual neural network [12] were designed, and the datasets for network training were collected.
- The learned metric was applied to pose optimization, and the corresponding Jacobian of Gauss-Newton was derived.
- The performance verification of the pose optimization method based on learned metrics was completed by comparison experiments.
- The key affecting factors of the accuracy and robustness of pose estimation were analyzed.

Judging the approximate movement between patches can be done manually but is difficult using the conventional model. We apply machine learning to the parts that the mapping is stable. The remainder of the paper is organized as follows. Section II presents related works. Section III gives an overview of the proposed method. Section III-A presents the datasets collection and network training for the metric. The pose optimization method based on learned metric is designed in Section III-B. Section IV verifies the performance of the network and optimization method based on the learned metric. Section V and Section VI contain the discussion and conclusion of this work.

II. RELATED WORKS

The related works are introduced from the metric improvement in pose optimization and the metric learning application.

A. METRIC IMPROVEMENT IN POSE OPTIMIZATION

Intensity-based pose optimization methods are mostly based on the similarity metric mentioned in [10]. A few works have attempted to improve optimization performance by changing the similarity metric. Reference [13] applied the artificially designed metric to pose optimization, it proposes a self-similarity weighted graph-based implementation of $\alpha$-mutual information ($\alpha$-MI) for nonrigid image registration. Reference [14] also completed the nonlinear optimization of 3D pose based on mutual information feature. This artificial design metric improves the convergence and robustness of pose optimization to a certain extent. However, the positive correlation between learning-based metrics and pose parameters can be stronger than mutual information. Neural network has strong mapping fitting ability, which provides new ideas for metric design.

The learning-based metric was initially applied to medical image registration. For example, [15] used higher-level corresponding structures, such as the same organ and lesion appearing in two images, to learn the similarity metric. Reference [16] directly estimated the displacement vector field from a pair of input images, which is better than explicitly defining a dissimilarity metric. However, medical images are treated as a 2D graph. Thus, the pose estimation is based on homography, spatial transformation and dataset collection are much easier than those in our work. Reference [17] propose a method of learning suitable convolutional representations for camera pose retrieval based on nearest neighbor matching and continuous metric learning-based feature descriptors. Similar to our learned metrics, these descriptors are sensitive to pose errors. But it directly uses these features to regress the pose, and we optimize it based on the learned metric.

B. METRIC LEARNING APPLICATION

The metric learning problem is concerned with learning a distance function tuned to a particular task and is useful when used in conjunction with nearest-neighbor methods and other techniques that rely on distances or similarities [18]. It has been gradually applied to computer vision, such as image classification [19], RGB or depth image matching [20], and image search and understanding [21]. Typical applications are listed as follows:

- MatchNet [22] achieved patch matching through metric learning, which effectively improves the accuracy of matching.
- [23] used learned metrics for similarity search by learning a Mahalanobis distance function that captures the images' underlying relationships well.
- [24] used a multi-view deep metric learning architecture to recognize volumetric image stacks.
- [25] trained a convolutional neural network to predict how well two image patches match and use it to compute the stereo matching cost.

Some metrics are difficult to describe through modeling. However, neural networks can effectively describe these metrics for participation in related calculations. So that the neural networks can improve the performance of related visual computing through metric learning. Like [22] and [25], our study takes patches as the input, the output metric participates in the calculation by large patch sampling to ensure the robustness of pose optimization.

C. POSE ESTIMATION WITH DEEP LEARNING

Deep learning is applied to pose estimation in different ways. Early works [26]–[28] used images as input to construct neural networks and output absolute poses corresponding to images in fixed scenes. Some works [29] output the relative poses by learning adjacent images. The learning methods are divided into supervised learning with pose labels [30], [31].
and unsupervised learning without pose labels [32]–[34]. Most of the unsupervised learning methods also used stereo images [33], depth images [35], optical flow images [36], etc. as supervised information. The above works have great theoretical value, but no matter what the supervision method is, the pose of the camera is output by the network, and the truncation error of the regression network output cannot be overcome, so that the accuracy of pose estimation is difficult to be guaranteed. In addition, deep learning was also used in image depth estimation [37], feature extraction [38], data association [39], hyperparameter selection [40], etc. to improve the performance of pose estimation. Our research applied deep learning to compute optimization metrics to improve the accuracy and robustness of pose estimation.

III. ALGORITHM OVERVIEW

Pose optimization between two frames is achieved by ensuring that the projection points in the target frame and the points extracted in the reference frame are from the same 3D point in the real scene. The optimization usually minimizes alignment errors

$$\sum_{i=1}^{n} \|u'_{i} - \pi(T \cdot \pi^{-1}(u_{i}, d_{u}))\|^2, \quad (1)$$

or a photometric error

$$\sum_{i=1}^{n} \|I_{i}(u_{i}) - I_{i}(\pi(T \cdot \pi^{-1}(u_{i}, d_{u}))\|2 \quad (2)$$

where $T \in SE(3)$ is the relative pose, $d_{u_{i}}$ is the depth of $u_{i}$ in the reference frame, and $u'_{i}$ is the pixel point in the target frame that matches $u_{i}$. $\pi$ and $\pi^{-1}$ are the projection and inverse projection, respectively. The calculation of $u'_{i}$ in (1) requires feature extraction and matching. Mismatching can cause $u'_{i}$ to deviate and undermine the optimization convergence. The gradient calculated using the photometric error in (2) is also difficult to guarantee the optimization convergence and robustness with respect to initial values and illumination.

As shown in Figure 1, the neural network is trained to input the patches and output alignment error $\delta u$ as the Gauss-Newton optimization residual (the blue $r$ in Figure 1). Then,

$$r_{i} \approx \delta u = u'_{i} - \pi(T \cdot \pi^{-1}(u_{i}, d_{u})) \quad (3)$$

where the weak error between $r_{i}$ and $\delta u$ is caused by the network learning error. The paired patches are obtained through $u_{i}$ in the reference frame and $\tilde{u}_{i} = \pi(T \cdot \pi^{-1}(u_{i}, d_{u}))$ in the target frame and are input to the network through channel stacking. The Jacobian is calculated directly using the gradient w.r.t the alignment error (the red $J$ in Figure 1), which is only related to $\tilde{u}_{i}$.

In this way, the metric of pose optimization is output from the patches through the network, and the $u'_{i}$ does not have to be calculated. The Jacobian calculated using the alignment error avoids the unreliability caused by illumination and the initial value dependence caused by the photometric error. As a result, the pose optimization can converge at a wider range of initial values.

A. DATASET COLLECTION AND NETWORK TRAINING

This learning task is first proposed, and no ready-made datasets and network structures are available for training.

1) DATASET COLLECTION

As shown in Figure 3, the training data requires each pair of patches to correspond to its label, which is the Euclidean distance of patch center in the pixel coordinate system. The patch dataset is collected from CoRBS dataset [41], which includes RGB and depth images captured by Kinect v2, and camera poses corresponding to each frame of the image. The depth image in CoRBS is more accurate than the depth image captured by Kinect v1 [42], [43]. On the basis of CoRBS, feature points $u_{i}$ are extracted in the reference frame, and these points are projected into the target frame. Thus, we obtain $\tilde{u}_{i}$.

$$\tilde{u}_{i} = \pi(T \cdot \tilde{T} \cdot \pi^{-1}(u_{i}, d_{u})) \quad (4)$$

The label is

$$Label : \quad \delta u_{i} = \tilde{u}_{i} - u'_{i}, \quad (5)$$

where $u'_{i}$ is the matching point of $u_{i}$ in the target frame. $d_{u_{i}}$ is the depth corresponding to $u_{i}$, which is obtained from the depth image. $\Delta T \in SE(3)$ is poses interference and is used to generate data for different labels.

$$\Delta T(\xi) = \exp(\xi^\wedge) \quad \xi_{i} \sim U[-a, a], \quad i = 1, 2, \cdots, 6, \quad (6)$$

where $\xi \in \mathbb{R}^{6}$ denotes Lie algebra.

The input item of the dataset are pairwise patches centered on $\tilde{u}_{i}$ and $u_{i}$. The orientation of the patch corresponding to $\tilde{u}_{i}$
and $u_i$ is different due to rotation between frames.

$$o_{\tilde{u}_i} = R_{[1:2,1:2]} \cdot o_{u_i}, \quad (7)$$

where $o$ is $2 \times 1$ vectors representing the patch orientation in the image and $R$ is the rotational part of $\Delta T \cdot T$. $R_{[1:2,1:2]}$ is the 2 $\times$ 2 element in the upper left corner of $R$.

2) NETWORKS

For this special learning task, we designed a regression neural network and improved ResNet-18 [12]. As shown in Figure 2, the networks take a $2 \times 32 \times 32$ patch as the input and a $2 \times 1$ vector as the output, which is the optimization metric. Two patches are stacked in the channel dimension. The regression network extracts features through the convolution layer, $\text{relu}$ acts as an activation function and $\text{BatchNorm}$ maintains the network generalization. The $2 \times 1$ vector is output through the fully connected layer.

ResNet-18 has four large blocks. However, we only use two of them because the input patch is small. The input channel of the first convolution layer is changed to 2, and the output dimension of the last FC is changed to 2. The original layer of ResNet-18 loads the trained parameters ($\text{pytorch}$ saves the network parameters trained on ImageNet [44]), and the modified layer (the red layers in Figure 2) is re-initialized ($\text{xavier\_uniform}$ for weight and $\text{constant}$ for bias). The entire network is retrained on the basis of the initial parameters.

B. POSE OPTIMIZATION

The pixel depth in the reference frame can be obtained from the depth image when solving the pose, which is a 3D/2D optimization. The optimization is based on Lie algebra $\mathfrak{se}(3)$, and the optimization parameter is $\xi = (\omega, v)$, where $\omega$ is the angular velocity and $v$ the linear velocity. We verify the optimized performance of the learned metric from two aspects: only optimizing the pose, and simultaneously optimizing the pose and the mappoint. The typical feature-based method using ORB [7] and the intensity-based method using SSD are designed as the baseline for comparison.

1) FEATURE-BASED METHOD

The optimization model of the feature-based method is

$$\xi^* = \arg \min \frac{1}{2} \sum_{i=1}^{N} ||u_i' - K \exp(\xi^\wedge)P_i||^2, \quad (8)$$

where $K$ is intrinsic camera parameters. $P_i = \pi^{-1}(u_i, d_u)$, and $\pi$ is also determined by $K$. The Jacobian corresponding to the residual of this objective is

$$J = -\frac{\partial \tilde{u}}{\partial \xi} = -\begin{bmatrix} -\frac{f_x X'}{Z^2} & -\frac{f_y Y'}{Z^2} \\ -\frac{f_x Y'}{Z^2} & -\frac{f_y X'}{Z^2} \\ Z' & Z' \\ 0 & 0 \\ Z' & Z' \\ 0 & 0 \\ \end{bmatrix}^T, \quad (9)$$

where $f_x, f_y$ is the focal length of $K$, $P_i' = [X', Y', Z'] = \exp(\xi^\wedge)P_i$, and $\tilde{u} = K \exp(\xi^\wedge)P$.

2) INTENSITY-BASED METHOD

The optimization model of the intensity-based method is

$$\xi^* = \arg \min \frac{1}{2} \sum_{i=1}^{N} e_i^T e_i$$

$$e_{i,j} = ||I_r(u_i, j) - I_i(\tilde{u}_i, j)||^2$$

$$\tilde{u}_i = K \exp(\xi^\wedge)P_i, \quad (10)$$

where $j$ is the $j$-th pixel of $32 \times 32$ patch. $I_r, I_i$ represent gray of the reference and target images, respectively. $e_i$ is a vector of $1024 \times 1$. The orientation of the patch in the target image needs to be rotated in accordance with (7). The Jacobian corresponding to the residual of this objective is

$$J = -2(I_r(u) - I_i(\tilde{u})) \frac{\partial I_i}{\partial \tilde{u}} \frac{\partial \tilde{u}}{\partial \xi} \quad (11)$$
where \( \frac{\partial I_r}{\partial u} \) is the gray gradient of patch, which is a 1024 × 2 vector. \( I_r(u) \) is gray of patch in reference image, which is a 1 × 1024 vector. The same is true for \( I_t(\bar{u}) \). \( \frac{\partial I_t}{\partial \delta} \) is the same as in (9).

3) BASED ON LEARNED METRIC

The optimization model of the patch-based method using learning metric is

\[
\xi^* = \arg \min_{\xi} \frac{1}{2} \sum_{i=1}^{N} \| \text{Net}(Pa_r(u_i), Pa_t(\bar{u}_i)) \|^2
\]

where \( Pa_r(u) \) is the patch corresponding to \( u \) in the reference image, \( Pa_t(\bar{u}) \) is the patch corresponding to \( \bar{u} \) in the target image. The residual \( \mathbf{r} = \text{Net}(Pa_r(u_i), Pa_t(\bar{u}_i)) \) is a 2 × 1 vector, and Jacobian \( \mathbf{J} \) is the same as in (9). We derive the Jacobian directly from the alignment error \( (u' - \bar{u}) \), because the alignment error is approximately equal to the network output. The model is optimized using Gauss-Newton, which is computed as

\[
\mathbf{H} = \mathbf{J}^T \mathbf{WJ} \quad \text{and} \quad \mathbf{b} = -\mathbf{J}^T \mathbf{Wr}
\]

where \( \mathbf{W} \in \mathbb{R}^{2 \times 2} \) is the diagonal matrix containing the weight, \( \mathbf{H} \in \mathbb{R}^{6 \times 6} \) is approximate Hessian matrix and \( \mathbf{b} \in \mathbb{R}^6 \) is gradient vector. As shown in Figure 1, the calculation of \( \mathbf{J} \) and \( \mathbf{r} \) does not require the matching point \( u' \) of \( u \). Thus, using this method does not require extracting and matching features.

C. POSE OPTIMIZATION WITH MAPPPOINTS

If the mappoints need to be optimized, then the optimization parameters also include \( \mathbf{P}_i \), which is the coordinate of the \( i \)-th mappoint. The Jacobian corresponding to \( \mathbf{P}_i \) is

\[
\frac{\partial \mathbf{e}}{\partial \mathbf{P}} = \begin{bmatrix}
    f_c & 0 & \frac{f_c X'}{Z^2} \\
    0 & f_c & \frac{f_c Y'}{Z^2} \\
    0 & 0 & -1
\end{bmatrix} \mathbf{R},
\]

where \( \mathbf{R} \) is the rotating part of the relative pose. When \( \mathbf{P}_i \) is involved in optimization, interference is applied on the pixel depth

\[
d = d + \delta d, \quad \delta d \sim U[-0.2, 0.2].
\]

In this way, the pose optimization performance based on the learned metric can be verified when both the pose and the mappoints are optimized simultaneously.

IV. EXPERIMENTS

The network is trained through the torch architecture in Python. The pose optimization uses the Gauss-Newton in g2o [45] library. The trained network is called by C++ via the torch.jit module. The paired images of CoRBS dataset are randomly divided into a training set and test set. The patch data set used for network training is collected on the images of the training set, and the pose accuracy is verified on the test set. A series of experiments are conducted to evaluate the learned metric performance: IV-A demonstrates the dataset and verifies the performance of the network; IV-B verifies the accuracy and robustness of the method based on learned metric compared with feature- and intensity-based methods; IV-C analyzes the key affecting parameters of the pose optimization performance.

| \( \delta u \) | -8.3 | -3.5 | 1.2 | 4.6 | 4.0 |
| \( \delta v \) | 1.0 | -4.2 | -8.2 | 1.4 | 3.1 |

FIGURE 3. Dataset example: the leftmost of the first row is the patch in reference image, and the other is the patch in target image and correspondence to the different labels in the second and third rows. The last two rows are the gradient map about network output \( (\delta u, \delta v) \) relative to the input patch in the target frame.

A. DATASET AND NETWORK

As shown in Figure 3, each reference patch corresponds to different projection patches depending on \( \Delta T \). The second and third lines are the corresponding labels of the pairwise patches, that is, the Euclidean distance between two patch centers. The values of \( a \) in (6) are set to 0.02, 0.06, 0.1, and 0.2 to ensure wide distribution of the dataset. The dataset is divided into training and test sets by a ratio of 10 to 1.

In an ideal situation, if the pose and scene depth (\( \mathbf{T} \) and \( d_{ui} \) in (4)) in the RGB-D dataset are accurate, the projection patch \( \bar{u}_i^* \) of the reference patch \( u_i \) under the real pose transformation \( \mathbf{T} \) should be equal to the matching patch \( u'_i \) of \( u_i \), that is, \( \bar{u}_i^* = u'_i \). The difference between \( \bar{u}_i^* \) and \( u'_i \) can characterize the accuracy of the dataset. We compared the accuracy of the patch pairs collected from the TUM RGB-D dataset (Kinect v1) and CoRBS dataset (Kinnect v2). The results are shown in Figure 4, if the CoRBS collected by Kinect v2 is used to make the dataset, the patch dataset accuracy is higher.

We use this dataset to train the networks in Figure 2. The network training uses the pytorch architecture, the learning rate is 0.01, the momentum is 0.9, and the optimization method is SGD. The training and test results are shown in Figure 5. The improved ResNet is significantly better than the ordinary regression network in convergence and generalization. ResNet has strong feature extraction and expression capabilities and is suitable for this task.

The statistical results of improved ResNet output are shown in Figure 6. The horizontal and vertical coordinates in the figure are the network label and output, respectively. The points scattered in the figure are the average of 1000 sets of
B. POSE OPTIMIZATION ACCURACY

The pose optimization is performed on the images in the test set of CoRBS. The frame relative pose is optimized by feature-based method, the intensity-based method, and our method using learned metric. The optimization model is divided into two types:

1) **pose only**: the pixel depth is known and is considered to be accurate, and only the pose is optimized;
2) **with point**: the coordinates of the mappoints have an error and need to be optimized together with the pose.

The position and attitude accuracy are represented by relative pose error (RPE) and rotating angle (RA) of relative pose, respectively:

\[
RPE = ||\text{trans}(T^{-1}T^*)||^2 \\
RA = \arccos\left(\frac{\text{tr}(\text{rotat}(T^{-1}T^*)) - 1}{2}\right)
\]

where \(T\) is the optimized pose; \(T^*\) is the true value; and \(\text{trans}\) and \(\text{rotat}\) refers to the translational and rotational component of \(T\), respectively.

We set several different initial value distributions to verify the convergence of optimization algorithm under various initial values.

\[
T_0 = \exp(\xi^\wedge)T^* \\
\xi_i \sim U[-a, a], \quad i = 1, 2, \ldots, 6 \quad (15)
\]

where \(T_0\) is the initial pose; \(T^*\) is the true value; and the values of \(a\) are set to 0.02, 0.06, 0.1, and 0.2 to represent four initial value environments.

Under the four initial conditions of the two optimization modes, we verify the optimization results of the three methods, obtaining the position and angle errors. A total of eight simulations are conducted, and each simulation generates 900 sets of random initial poses for the sequence images in accordance with (15). As a result, 900 sets of optimization results of the three methods are obtained. The statistical results (position and attitude error) of the eight groups of simulations are shown in Table 1. The 900 sets of data for each simulation are sorted by the initial value and divided into nine groups, and the mean and standard deviation of 100 data of each group are calculated. The result of each simulation is shown in Figure 7.

Table 1 shows the optimization accuracy of the position and attitude of the two modes (pose only and with point) in different initial values. The pose accuracy of the three methods decreases as the initial environment becomes increasingly harsh. However, our method has weak initial value dependence and can maintain high precision in harsh initial conditions. The learned metric method achieves the highest precision in 13 of the 16 group simulation results.

Figure 7 shows the comparison of the three methods in terms of the position and attitude accuracy under four initial conditions of two optimization modes. Each row in the graph presents an optimization result under an initial condition. The first two columns are the position and attitude optimization results in pose only mode, and the last two columns are the optimization results in the with point mode.
TABLE 1. Pose optimization accuracy statistics results of the three methods.

| Method                      | Error (10^{-2}) | Mode | Pose only (RPE RA) | With point (RPE RA) |
|-----------------------------|-----------------|------|--------------------|---------------------|
|                             |                 |      | a = 0.02          | a = 0.06            | a = 0.1          |
|                             | Initial         |      | 0.960 0.960       | 2.859 2.926         | 4.625 4.781     |
|                             | Intensity-based (SSD) | 0.468 0.362 | 0.675 0.558 | 1.793 2.055 | 7.767 8.240 |
|                             | Feature-based (ORB) | 0.485 0.328 | 0.754 0.827 | 1.160 1.443 | 2.587 2.938 |
|                             | Our (Learned Metric) | 0.501 0.390 | 0.524 0.370 | 0.556 0.337 | 0.779 0.447 |
|                             |                 |      | a = 0.1          | a = 0.2            | a = 0.2          |
|                             | Initial         |      | 9.608 9.467      | 2.837 2.894         | 4.901 4.816     |
|                             | Intensity-based (SSD) | 1.182 0.488 | 6.977 3.153 | 13.53 6.480 | 22.08 12.22 |
|                             | Feature-based (ORB) | 0.477 0.283 | 0.811 0.543 | 1.250 0.654 | 2.600 1.348 |
|                             | Our (Learned Metric) | 0.473 0.345 | 0.624 0.398 | 0.708 0.432 | 1.769 0.653 |

FIGURE 7. Pose optimization accuracy includes mean and standard deviation. Each row represents results under a initial condition. The first two columns and the last two columns are position and attitude optimization results in the pose only and with point modes, respectively.

For *pose only* mode, when the initial value condition is good (a = 0.02), the traditional method can effectively converge, and the three methods converge to nearly the same precision. The true pose of the dataset also has small errors. Thus, the error of the algorithm cannot be infinitely close to zero. At this time, the error between the metric of network output and the real label causes our method inferior to the two other methods. The method based on learned metric can maintain strong robustness and good convergence as the initial condition becomes increasingly harsh. Its optimization accuracy is much higher than those of the two other methods.
When the mappoints are involved in the optimization, the optimization precision considerably decreased compared with that in pose only mode due to the noise added to the mappoint position. For the intensity-based method based on SSD, the degree of accuracy reduction is the most evident or even worse than the initial value because of the strong initial value dependence of the calculation metric SSD. Comparing (a), (b) with (c), (d) shows that the optimization of joining mappoints further improves the accuracy when the initial value of the optimization is close to the true value although the mappoints are noisy. Our method and feature-based method have better convergence under this condition, and the small error in the depth image is corrected by optimization.

In short, our learned metric has better performance than the computational metric SSD. The learned metric-based optimization method is also better than the SSD-based direct method and even better than the feature-based method that requires feature extraction and matching.

We draw the gradient verification graph to further compare the optimization performance of the algorithm. This graph takes the deviation between the current and true values as the horizontal axis and the optimization parameter increment calculated by the algorithm as the vertical axis. As shown in Figure 8, ideally, the increment calculated by the algorithm should be equal to the deviation between the current and true values. At this point, the algorithm will converge to the true value in the most efficient way. Thus, the best distribution of points in the figure is distributed along \( y = -x \). The learned metric-based optimization method is close to this distribution by virtue of its reasonable gradient calculation and robustness. The results of the intensity-based method based on SSD scatter around \((0, 0)\), and the algorithm is likely to converge only when the estimated value is close to the true value. This condition indicates the initial value dependence based on the SSD metric. Partial results of the ORB feature-based method are distributed along with \( y = -x \), and the poor robustness caused by mismatching results in partially unsatisfactory gradient calculations. The gradient calculation along \( z \) is particularly poor due to the motion pattern of the sequence images.

C. PARAMETER ANALYSIS

To fully analyze the learned metric and the network performance, the pose accuracy with different number of features is calculated to verify the mismatch impact on the optimization system, and the different labels are used to train the network to verify whether the network learned the inherent law between patches.

1) FEATURE POINT NUMBER

We calculate the accuracy of the three optimization methods under different feature numbers. The intensity-based method only uses the position of the features. As shown in Figure 9, the accuracy of the SSD- and learned metric-based methods increases as the number of features increases because a large number of features enhance their robustness. However, the accuracy of the feature-based method decreases because
Figure 9. Pose accuracy w.r.t. point number. In the horizontal brackets, $P$ represents position and $A$ represents attitude.

A large number of features increase the probability of mismatching.

The best output of the designed network should be equal to the label. At this time, the optimization objective of the learned metric-based method is the same as that of the feature-based method. From this perspective, the highest precision of our method is the accuracy of the feature-based method. However, the accuracy and robustness of our method are better because it does not involve matching and therefore does not have risk of mismatch.

2) NETWORK TRAINING LABEL

When making the dataset, the label is calculated in accordance with (5). When $\Delta T = I$, $\tilde{u}_i^* = \pi(T \cdot \pi^{-1}(u_i, d_u))$, ideally $\tilde{u}_i^* = u'_i$. However, a slight error exists between the two due to the error of the true pose and depth map in the dataset. When $\tilde{u}_i = \tilde{u}_i^*$ instead of $\tilde{u}_i = u'_i$, the true pose in dataset is obtained. We also define the $Label^*$ to train the network.

$$Label^* : \delta u_i^* = \tilde{u}_i - \tilde{u}_i^*,$$

(16)

The network trained with these two labels is used for pose optimization, respectively. The optimization results are shown in Figure 10. The optimization results using the network trained by $Label$ show higher precision and stronger robustness. When the result of RPE greater than 0.1 or RA greater than 0.02 is eliminated, the average accuracy of using $Label^*$ is higher. The difference in optimization results is caused by the weak error between $u'_i$ and $\tilde{u}_i^*$. The above-mentioned comparison shows that the mapping of patches to $Label^*$ is unstable. The network training does not have good generalization. As a result, only data close to the sample can fit the better optimization results.

The training using $Label$ in (5) allows the network to learn the inherent mapping of patches to their center differences in addition to the fit of data. This mapping is stable, which indicates that we have applied the neural network to a reasonable part.

Figure 10. Pose accuracy w.r.t. network label. The top and bottom charts are the results of position and attitude. $Label$, and $Label^*$ only retained good results that RPE smaller than 0.1 and RA smaller than 0.02.

3) JACOBIAN CALCULATION

For the objective function of the learned metric-based method in (12), if the corresponding Jacobian is directly calculated, then it should be

$$J = \frac{\partial Net}{\partial Pa_t(\tilde{u})} \frac{\partial I_t}{\partial \tilde{u}} \frac{\partial \xi}{\partial \tilde{u}}$$

(17)

where $\partial Net/\partial Pa_t(\tilde{u})$ is a $2 \times 1024$ derivative matrix about network output relative to the input patch, which is obtained by network backpropagation. We also try the optimization based on this Jacobian, but the calculation of $\partial Net/\partial Pa_t(\tilde{u})$ is slow and the optimization accuracy is low. This result may be caused by a deviation of $\partial I_t/\partial \tilde{u}$ or some outliers in $\partial Net/\partial Pa_t(\tilde{u})$.

V. DISCUSSION

We discussed our research’s relationships with visual odometry and learning-based pose estimation.

A. RELATIONSHIPS WITH VISUAL ODOMETRY

A high-precision visual odometry requires various auxiliary strategies, such as pose initialization of front-end, image pyramid, sliding window, and local map optimization. We did not use any auxiliary strategies in pose optimization to clearly verify the advantages of the learned metric over traditional metrics. To design a visual odometry based on a learned metric, with the above-mentioned auxiliary strategies, we need to do the following:
For a specific pose initialization method, the difference between the initialized pose and the true value should conform to a certain distribution, and the dataset should be collected on this basis of this distribution to training the network so that the learned metric can adapt the motion mode.

- The patches should be input into the network in batches, to achieve high efficiency of the neural network distributed computing. Adjusting the size and number of the batches and the method in [46] can be used to balance the speed and accuracy of visual odometry.
- The learned metric can also be applied to mature SLAM systems, such as ORB and DSO, to improve its performance.

VI. CONCLUSION

Pose optimization is the basis of geometric vision work such as 3D reconstruction, SFM, and visual odometry. A pose optimization method based on the learned metric is proposed. The process of optimization is uncertain and the distribution of patch samples is difficult to predict. The accuracy of the network output also affects the optimization performance. These difficulties make the network need to have strong generalization and learning ability. The optimization performance also depends on reasonable learned metric applications and corresponding Jacobian design. The main contributions are summarized as follows:

- The neural network was designed and trained by the collected patch datasets. The convergence and generalization of the network were verified.
- The optimization method based on the learned metric was designed using a reasonable Jacobian, and the high precision and robustness were verified by simulation through comparison with intensity- and feature-based methods.
- The affecting factors of the accuracy and robustness of pose optimization were analyzed to reveal the reason for the good performance of the proposed method, which provides a valuable reference for further development in this research field.

The target patch input to network is obtained by projection. The entire pose estimation process does not require feature extraction and matching. Thus, the method can be applied in a wide range of scenarios. The improvement of optimization results brought by the learned metric is verified. The learned metric increases the optimization convexity and robustness with respect to the initial value. No matching needed avoids damage to the system caused by mismatches. Despite the slight error in the network output and the true value $\delta a$, the residuals and Jacobian calculated by a large number of patch samples almost erase the effect of this error. This way of ‘learning + model computing’ provides a good reference for the solution of geometric vision problems.

Different patch sizes and patterns can be also used to learn the metrics for finding the most appropriate results. Apart from the difference from the patch center, additional metrics can also be learned from patches to direct the optimization. The metric can also be learned in a targeted manner according to different application modes to enhance its suitability.

REFERENCES

[1] C. Forster, M. Pizzoli, and D. Scaramuzza, “SVO: Fast semi-direct monocular visual odometry,” in Proc. IEEE Int. Conf. Robot. Autom. (ICRA), May 2014, pp. 15–22.
[2] E. Tola, V. Lepetit, and P. Fua, “DAISY: An efficient dense descriptor applied to wide-baseline stereo,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 32, no. 5, pp. 815–830, May 2010.
[3] T. Whelan, M. Kaess, H. Johansson, M. Fallon, J. J. Leonard, and J. J. Leonard, “Real-time large-scale dense RGB-D SLAM with volumetric fusion,” Int. J. Robot. Res., vol. 34, nos. 4–5, pp. 598–626, Apr. 2015.
[4] J. L. Schonberger and J.-M. Frahm, “Structure-from-Motion revisited,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 2016–4113.
[5] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” Int. J. Comput. Vis., vol. 60, no. 2, pp. 91–110, Nov. 2004.
[6] J. Buy, A. Ess, T. Tuytelaars, and L. Van Gool, “Speeded-up robust features (SURF),” Comput. Vis. Image Understand., vol. 110, no. 3, pp. 346–359, Jun. 2008.
[7] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “ORB: An efficient alternative to SIFT or SURF,” in Proc. Int. Conf. Comput. Vis., Nov. 2011, pp. 2564–2571.
[8] H. Zhou, T. Zhang, and W. Lu, “Vision-based pose estimation from points with unknown correspondences,” IEEE Trans. Image Process., vol. 23, no. 8, pp. 3468–3477, Aug. 2014.
[9] J. Engel, T. Schöps, and D. Cremers, “Lsd-SLAM: Large-scale direct monocular SLAM,” in Proc. Eur. Conf. Comput. Vis., 2014, pp. 834–849.
[10] S. Madhuri G and M. P. Indra Gandhi, “Image registration quality assessment with similarity measures—A research study,” in Proc. Int. Conf. Commun. Signal Process. (ICDSP), Apr. 2015, pp. 0084–0088.
[11] J. Engel, V. Koltun, and D. Cremers, “Direct sparse odometry,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 40, no. 3, pp. 611–625, Mar. 2018.
[12] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 770–778.
[13] H. Rivaz, Z. Karimaghaloo, and D. L. Collins, “Self-similarity weighted mutual information: A new nonrigid image registration metric,” Med. Image Anal., vol. 18, no. 2, pp. 343–358, Feb. 2014.
[14] G. Caron, A. Dami, and E. Marchand, “Direct model based visual tracking and pose estimation using mutual information,” Image Vis. Comput., vol. 32, no. 1, pp. 54–63, Jan. 2014.
[15] Y. Hu, M. Modat, E. Gibson, N. Ghavami, E. Bonmati, C. M. Moore, M. Emberton, J. A. Noble, D. C. Barratt, and T. Vercauteren, “Label-driven weakly-supervised learning for multimodal deformable image registration,” in Proc. IEEE 15th Int. Symp. Biomed. Imag. (ISBI), Apr. 2018, pp. 1070–1074.
[16] H. Sokooti, B. D. Vos, F. Berendsen, B. P. Lelieveldt, I. Išgum, and M. Staring, “Nonrigid image registration using multi-scale 3d convolutional neural networks,” in Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent. 2017, pp. 232–239.
[17] V. Baltsas, S. Li, and V. Prisacariu, “Relocnet: Continuous metric learning relocalisation using neural nets,” in Proc. Eur. Conf. Comput. Vis. (ECCV), 2018, pp. 751–767.
C. Tang and P. Tan, “BA-net: Dense bundle adjustment network,” in Proc. Int. Conf. Adv. Concepts Intell. Syst., vol. 5, no. 4, pp. 287–364, 2013.

B. Harwood, G. Carneiro, I. Reid, and T. Drummond, “Smart mining for deep metric learning,” Space, vol. 9, no. 13, p. 22, 2017.

J.-H. Seo and D.-S. Kwon, “Learning similarity metric for comparing RGB-D image patches by CNN,” in Proc. 14th Int. Conf. Ubiquitous Robot. Ambient Intell. (URAI), Jun. 2017, pp. 490–491.

J. Wang, F. Zhou, S. Wen, X. Liu, and Y. Lin, “Deep metric learning with angular loss,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 2593–2601.

X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg, “MatchNet: Unifying feature and metric learning for patch-based matching,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2015, pp. 3279–3286.

P. Jain, B. Kulis, and K. Grauman, “Fast image search for learned metrics,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2008, pp. 1–8.

X. Wang and M. Liu, “Multi-view deep metric learning for volumetric image recognition,” in Proc. IEEE Int. Conf. Multimedia Expo (ICME), Jul. 2018, pp. 1–6.

J. Zhontar and Y. LeCun, “Computing the stereo matching cost with a convolutional neural network,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2015, pp. 1592–1599.

A. Kendall, M. Grimes, and R. Cipolla, “PoseNet: A convolutional network for real-time 6-DOF camera relocalization,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Dec. 2015, pp. 2938–2946.

A. Kendall and R. Cipolla, “Geometric loss functions for camera pose regression with deep learning,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 5974–5983.

J. Wu, L. Ma, and X. Hu, “Delving deeper into convolutional neural networks for camera relocalization,” in Proc. IEEE Int. Conf. Robot. Autom. (ICRA), May 2017, pp. 5644–5651.

I. Melekhov, J. Ylioinas, J. Kannala, and E. Rahtu, “Relative camera pose estimation using convolutional neural networks,” in Proc. Int. Conf. Adv. Concepts Intell. Syst., 2017, pp. 675–687.

G. Costante, M. Mancini, P. Valigi, and T. A. Ciarfuglia, “Exploring representation learning with CNNs for frame-to-frame ego-motion estimation,” IEEE Robot. Autom. Lett., vol. 1, no. 1, pp. 18–25, Jan. 2016.

S. Wang, R. Clark, H. Wen, and N. Trigoni, “DeepVO: Towards end-to-end visual odometry with deep recurrent convolutional neural networks,” in Proc. IEEE Int. Conf. Robot. Autom. (ICRA), May 2017, pp. 2043–2050.

T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, “Unsupervised learning of depth and ego-motion from video,” in Proc. CVPR, vol. 2, 2017, p. 7.

R. Li, S. Wang, Z. Long, and D. Gu, “UnDeepVO: Monocular visual odometry through unsupervised deep learning,” in Proc. IEEE Int. Conf. Robot. Autom. (ICRA), May 2018, pp. 7268–7279.

Z. Yin and J. Shi, “GeoNet: Unsupervised learning of dense depth, optical flow and camera pose,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 1983–1992.

S. Vijayanarasimhan, S. Ricco, C. Schmid, R. Sukthankar, and K. Fragkiadaki, “SIM-net: Learning of structure and motion from video,” 2017, arXiv:1704.07804. [Online]. Available: http://arxiv.org/abs/1704.07804

B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy, and T. Brox, “DeMoN: Depth and motion network for learning monocular stereo,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 5038–5047.

K. Tateno, F. Tombari, I. Laina, and N. Navab, “CNN-SLAM: Real-time dense monocular SLAM with learned depth prediction,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 6243–6252.

D. DeTone, T. Malisiewicz, and A. Rabinovich, “Self-improving visual odometry,” 2018, arXiv:1812.03245. [Online]. Available: http://arxiv.org/abs/1812.03245

X. Wang, K. Wang, and S. Shen, “FlowNorm: A learning-based method for increasing convergence range of direct alignment,” 2019, arXiv:1910.07217. [Online]. Available: http://arxiv.org/abs/1910.07217

C. Tang and P. Tan, “BA-net: Dense bundle adjustment network,” 2018, arXiv:1806.04807. [Online]. Available: http://arxiv.org/abs/1806.04807

O. Wesenmuller, M. Meyer, and D. Stricker, “CoRBS: Comprehensive RGB-D benchmark for SLAM using kinect V2,” in Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV), Mar. 2016, pp. 1–7.

A. Schmidt, M. Fularz, M. Kraft, A. Kasiński, and M. Nowicki, “An indoor rgb-d dataset for the evaluation of robot navigation algorithms,” in Proc. Int. Conf. Adv. Concepts Intell. Syst., 2013, pp. 321–329.

J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, “A benchmark for the evaluation of RGB-D SLAM systems,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., Oct. 2012, pp. 573–580.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2009, pp. 20–25.

R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, “G2o: A general framework for graph optimization,” in Proc. IEEE Int. Conf. Robot. Autom., pp. 3607–3613, IEEE, 2011.

S.-P. Li, T. Zhang, X. Gao, D. Wang, and Y. Xian, “Semi-direct monocular visual and visual-inertial SLAM with loop closure detection,” Robot. Auto. Syst., vol. 112, pp. 201–210, Feb. 2019.

Z. Lv, F. Dellaert, J. M. Rehg, and A. Geiger, “Taking a deeper look at the inverse compositional algorithm,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 4581–4590.

TAO ZHANG (Senior Member, IEEE) was born in March 1969. He received the B.S., M.S., and Ph.D. degrees from Tsinghua University, Beijing, China, in 1993, 1995, and 1999, respectively, and the Ph.D. degree from Saga University, Saga, Japan, in 2002. He is currently a Professor and the Head of the Department of Automation, School of Information Science and Technology, Tsinghua University. He is also the Laboratory Director of the Beijing National Research Center for Information Science and Technology. He is the author or coauthor of more than 200 articles and six books. His current research interests include robotics, image processing, control theory, artificial intelligent, and navigation and control of spacecraft.
BANGJIE LI received the Ph.D. degree in aircraft design from the High-Tech Institute of Xi’an, China, in 2008. He was a Professor with the High-Tech Institute of Xi’an, in 2015. His current research interests include design and optimization of spacecraft orbit, computer vision, and artificial intelligent.

DAQIAO ZHANG received the Ph.D. degree in aircraft design from the High-Tech Institute of Xi’an, China, in 2011. He is currently a Lecturer with the High-Tech Institute of Xi’an. His current research interests include design and optimization of spacecraft orbit, UAV mission planning, and artificial intelligent.

WEILIN GUO received the B.S. degree from the High-Tech Institute of Xi’an, Xi’an, China, in 2014, where he is currently pursuing the Ph.D. degree. His research interests include aircraft design, integrated navigation, and artificial neural networks.

* * *