Response Letter

Manuscript ID: remotesensing-2453652

We highly appreciate the reviewers' comments and suggestions. We have carefully addressed all the review comments (highlighted in yellow) and improved the quality of this manuscript accordingly. Please find below our response to the review comments.

Response to comments of Reviewer 3

■ **Comment 1**

The English of paper is poor, please polish it again.

Response: Thank you for your comments. We have utilized the English editing services provided by MDPI to revise and polish the language across the entire manuscript. This revision has addressed the language issues and conveyed the research content more effectively. We hope this revision meets your requirements.

■ **Comment 2**

Each variable in the equation should be given the meaning.

Response: Thank you for your comments. We have explained the meaning of each variable in detail after all equations. These explanations will enhance the reader's understanding of the proposed method and provide better clarity. Thank you again for your suggestion, as it helps improve our manuscript's readability and comprehensibility. Our explanation of each variable in Equation 1 to Equation 10 is as follows (see details in lines 252-254, 270-271, 331-332, 367-371, 376-379, 382-383, 386-388, 392-395, 401-405 and 408-410 of the revised manuscript):

2.2. FCTrans Structure

Then, we applied two FCTrans blocks with cross-image attention to F_v^0, F_r^0, and F_c^0. In the l-th FCTrans block, we regard F_v^l as the query feature map (source feature map), F_c^l as the key/value feature map (projected target feature map), and F_r^l as the reference feature map (unprojected target feature map). In addition, the cross-image attention operation in each FCTrans block requires F_v^{l-1} and F_c^{l-1} as inputs to obtain the projected target feature map F_v^l, as shown at the top of Figure 2. F_v^{l-1} and F_r^{l-1} are regarded as the query image and the reference image, respectively, and do not need to be projected; therefore, F_v^l and F_r^l are obtained through FCTrans block without cross-image attention, respectively. The computations in the FCTrans block are as follows:

$$\begin{aligned} F_k^l &= MLP\left(LN\left(LN\left(F_k^{l-1}\right)\right)\right) + LN\left(F_k^{l-1}\right), \qquad k = v, r \\ \hat{F}_c^l &= f_c^{l-1} + F_c^{l-1} \\ F_c^l &= MLP\left(LN\left(\hat{F}_c^l\right)\right) + \hat{F}_c^l \end{aligned} \tag{1}$$

where $LN(\cdot)$ denotes the operation of the LayerNorm layer; $MLP(\cdot)$ denotes the operation of MLP; F_k^l indicates the feature map output by the l-th FCTrans block, where F_v^l, F_c^l, and F_r^l

denote the source feature map, the projected target feature map and the unprojected target feature map, respectively; f_c^{l-1} represents the feature map obtained with $LN(F_v^{l-1})$ and $LN(F_c^{l-1})$ as the input of cross-image attention; $\hat{F}_c^l$ stands for F_c^{l-1} the output feature map in the S(W)-CIA module.

To generate a hierarchical representation, we halved the feature map size and doubled the number of channels using the patch merging module. The two FCTrans blocks, together with a patch merging module, are called "Stage 1". Similarly, "Stage 2" and "Stage 3" adopt a similar scheme. However, their FCTrans block numbers are 2 and 6, respectively, and "Stage 3" does not have a patch merging module. After three stages, each feature patch in F_c^{10} implies a correlation with all the feature patches in the corresponding window of the source feature map at different scales, thus achieving the goal of projecting feature information from the source image into the target image.

Finally, we concatenated F_r^{10} and F_c^{10} to build $[\mathrm{F_r^{10}},\mathrm{F_c^{10}}]$, and then input it to the homography prediction layer (including the LayerNorm layer, global pooling layer, and fully connected layer) to output 4 offset vectors (8 values). With the 4 offset vectors, we obtained the homography matrix, H_{vr}, by solving the DLT [19]. We use $\mathrm{h}(\cdot)$ to represent the whole process, i.e.,

$$H_{vr} = h([F_r^{10}, F_c^{10}]) \tag{2}$$

where F_r^{10} represents the unprojected target feature map outputted by the 10th FCTrans block; F_c^{10} indicates the projected target feature map outputted by the 10th FCTrans block.

2.2.2. Cross-image Attention

First, we take F_v^{l-1} and F_c^{l-1} of size $\frac{H}{2^k} \times \frac{W}{2^k}$ (where k denotes the number of stages) processed by the LayerNorm layer as the query feature map and key/value feature map. We adopt a (shifted) window partitioning scheme and a feature patch partitioning scheme to partition them into windows of size $M \times M$ containing $\frac{M}{2} \times \frac{M}{2}$ feature patches. Next, we flatten these windows in the feature patch dimension, thus reshaping the window size to $N \times D$, where N denotes the number of feature patch ($\frac{M}{2} \times \frac{M}{2}$) and D represents the number of pixels in the feature patch (2×2). Then, the window of F_v^{l-1} passes through the fully connected layer to obtain the Query matrix, and the window of F_c^{l-1} passes through two different fully connected layers to obtain the Key matrix and the Value matrix, respectively. We compute the similarity between the Query matrix and all Key matrices to assign weights to each Value matrix. The similarity matrix is usually computed using the dot product and then normalized to a probability distribution via the softmax function. In this way, we can query the similarity between each feature in F_v^{l-1} (represented by feature patch) and all features in F_c^{l-1} within the corresponding windows of F_v^{l-1} and F_c^{l-1}, thus achieving the effect of explicit feature matching. Finally, we multiply the Value matrix and the similarity matrix to obtain the final output matrix, y_c^{l-1}, after obtaining the weighted similarity matrix. Each feature patch in this output matrix, y_c^{l-1}, implies the correlation between all the feature patches in the window corresponding to the source feature map, thus achieving a mapping from the source image to the target image in the feature dimension. This implementation process can be described as follows:

$$y_c^{l-1} = softmax(\frac{QK^T}{\sqrt{d}} + B)V \tag{3}$$

where Q, K, and V represent the Query, Key, and Value matrixes, respectively; d stands for the Q/K dimension, which is 2×2 in the experiment; and B represents the relative position bias. We used a feature patch as the unit of computation; therefore, the relative positions along each axis were in the range $[-\frac{M}{2}+1, \frac{M}{2}+1]$. We parameterized a bias matrix, $\hat{B} \in \mathbb{R}^{(M-1)\times(M-1)}$, and the values in B were taken from $\hat{B}$. We rescaled the output matrix y_c^{l-1}

of size $N \times D$ to match the size of the original feature map, i.e., $\frac{H}{2^k} \times \frac{W}{2^k}$. This adjustment could facilitate subsequent convolution operations or other image processing steps. In addition, we performed residual concatenation by adding the output feature map and the original feature map, F_c^{l-1}, to obtain the feature map, f_c^{l-1}, thus alleviating the gradient disappearance.

2.3.1. Loss Function of Generator

To solve the problem of the network having difficulty adequately capturing the feature relationship between infrared and visible images, we propose a constraint called "Feature Correlation Loss" (FCL). FCL aims to minimize the distance between the projected target feature map, F_c^l, and the source feature map, F_v^l, while maintaining a large distance between the unprojected target feature map, F_r^l, and the source feature map, F_v^l. This scheme encourages the network to continuously learn the feature correlation between the projected target feature map (F_c^l) and the source feature map (F_v^l) within the window, and then continuously weight the projected target feature map under multiple stages to achieve better feature matching with the source feature map. Our FCL constraint is defined as follows:

$$\begin{aligned} L_{fc}^l(F_v^l, F_c^l, F_r^l) &= \max\left(\left\|F_c^l - F_v^l\right\|_1 - \left\|F_r^l - F_v^l\right\|_1 + 1{,}0\right) \\ L_{fc}(F_v, F_r) &= \sum_{l=1}^{10} L_{fc}^l\,(F_v^l, F_c^l, F_r^l) \end{aligned} \tag{4}$$

where F_v^l, F_c^l, and F_r^l represent the source feature map, the projected target feature map, and the unprojected target feature map output by the l-th FCTrans block, respectively. $L_{fc}^l(F_v^l, F_c^l, F_r^l)$ denotes the loss generated by the l-th FCTrans block. F_v and F_r stand for the visible shallow feature map and infrared shallow feature map, respectively. Our FCL is the sum of the losses generated by all FCTrans blocks, i.e., $L_{fc}(F_v, F_r)$.

To perform unsupervised learning, we minimize three other losses in addition to constraining the FCL of FCTrans network training. The first one is the feature loss, which is used to encourage the feature maps between the warped and target images to have similar data distributions [47], written as:

$$L_f(I_r, I_v) = \max\left(\|F_r' - F_v\|_1 - \|F_r - F_v\|_1 + 1{,}0\right) \tag{5}$$

where I_v and I_r represent the visible image patch and the infrared image patch, respectively. F_v and F_r indicate the visible shallow feature map and the infrared shallow feature map, respectively. F_r' denotes the warped infrared shallow feature map obtained by warping F_r with the homography matrix, H_{rv}.

The second term is the homography loss, which is used to force H_{rv} and H_{vr} to be mutually inverse matrices [47], and is computed by:

$$L_{hom} = \|H_{vr}H_{rv} - E\|_2^2 \tag{6}$$

where E denotes the third-order identity matrix. H_{vr} represents the homography matrix from I_v to I_r. H_{rv} denotes the homography matrix from I_r to I_v.

The third term is the adversarial loss, which is used to force the feature map of the warped image to be closer to that of the target image [47], i.e.,

$$L_{adv}(F_r') = \sum_{n=1}^{N}(1 - logD_{\theta_D}(F_r')) \tag{7}$$

where $logD_{\theta_D}(\cdot)$ indicates the probability of the warped shallow feature map like a target shallow feature map, N represents the size of the batch, and F_r' stands for warped infrared shallow feature map.

In practice, we can derive the losses $L_f(I_v, I_r)$, $L_{adv}(F_v')$, and $L_{fc}(F_r, F_v)$ by exchanging the order of image patches I_v and I_r. Thus, the total loss function of the generator can be written as:

$$L_G = L_f(I_r, I_v) + L_f(I_v, I_r) + \lambda L_{hom} + \mu(L_{adv}(F_r')) + L_{adv}(F_v'))) + \xi(L_{fc}(F_v, F_r) + L_{fc}(F_r, F_v)) \quad (8)$$

where I_v and I_r stand for the visible image patch and infrared image patch, respectively. F_v and F_r indicate the visible shallow feature map and infrared shallow feature map, respectively. F_v' and F_r' represent warped visible shallow feature map and warped infrared shallow feature map, respectively. λ, μ, and ξ are the weights of each term set as 0.01, 0.005, and 0.05, respectively. We provide an analysis of parameter ξ in Appendix A.

2.3.1. Loss Function of the Discriminator

The discriminator aims to distinguish the feature maps between the warped image and the target image. According to [47], the loss between the feature map of the infrared image and the warped feature map of the visible image is calculated by:

$$L_D(F_r, F_v') = \sum_{n=1}^{N}(a - logD_{\theta_D}(F_r)) + \sum_{n=1}^{N}(b - logD_{\theta_D}(F_v')) \quad (9)$$

where F_r indicates the infrared shallow feature map; F_v' represents the warped visible shallow feature map; N represents the size of the batch; a and b represent the labels of the shallow feature maps F_r and F_v', which are set as random numbers from 0.95 to 1 and 0 to 0.05, respectively; and $logD_{\theta_D}(\cdot)$ indicates the probability of the warped shallow feature map to be similar to the target shallow feature map.

In practice, we can obtain the loss $L_D(F_v, F_r')$ by swapping the order of I_v and I_r. Thus, the total loss function of the discriminator can be defined as follows:

$$L_D = L_D(F_r, F_v') + L_D(F_v, F_r') \quad (10)$$

where F_v and F_r indicate the visible shallow feature map and infrared shallow feature map, respectively; F_v' and F_r' represent the warped visible shallow feature map and warped infrared shallow feature map, respectively.

■ **Comment 3 and Comment 4**

Response: Thank you very much for your constructive comments. We note that your comments 3 and 4 are both revised for section 3.2. To avoid double-pasting the revised manuscript and ensure you can easily track and understand our revisions, we decided first to answer the two comments separately and then provide the revised section 3.2 at the end of the response to comment 4. We hope that this response helps you understand our revisions and find relevant revisions. Next, we will specifically respond to your two comments.

■ **Comment 3**

Please give all parameters of the proposed method.

Response: Thank you very much for this point. We have added Table 2 in Section 3.2 to list all relevant parameters of the proposed method. Thank you again for your suggestion, as this addition makes our parameter selection clearer. Our revisions to Section 3.2 are at the end of the response to Comment 4 (see details in lines 446 and 459 of the revised manuscript):

■ **Comment 4**

How to train the model?

Response: Thank you very much for this comment. We have added more details about our model training process in Section 3.2. First, we describe our specific approach to data preprocessing, including image resizing, random cropping, normalization, and grayscaling. Second, we introduce our network's training environment, and emphasize our utilization of the Adam optimizer and the implementation of a learning rate decay strategy to effective network optimization. Finally, we detail the order and approach we use to update the discriminator and generator during each iteration and how we optimize the loss function. Meanwhile, we also describe the preservation of the training model. Our additions in Section 3.2 are as follows (see details in lines 439-457 of the revised manuscript):

3.2. Implementation Details

Our experimental environment parameters are shown in Table 1. During data preprocessing, we resized the image pairs to a uniform size of 150×150 and then randomly cropped them to image patches of size 128×128 to increase the amount of data. In addition, we normalized and grayscaled images to obtain patches I_v and I_r as the input of the model. Our network was trained under the PyTorch framework. To optimize the network, we employed the adaptive moment estimation (Adam) [53] optimizer with the initial value of the learning rate set to 0.0001 and adjusted by the decay strategy during the training process. All parameters of the proposed method are shown in Table 2. In each iteration of the model training, we first updated the discriminator (D) parameters and then the generator (FCTrans). Its loss function is optimized by backpropagation in each iteration step. Specifically, we first utilized the generator to generate a homography matrix by which the source image is warped to a warped image. Thus, we trained the discriminator using the warped and target images. We calculated the loss function of the discriminator using Equation (8) and then updated the discriminator's parameters by backpropagation. Next, we trained the generator. We computed the loss function of the generator using Equation (10) and updated the generator's parameters by backpropagation. We make the network continuously tuned to the homography matrix through the adversarial game between the generator and the discriminator. Meanwhile, we periodically saved the model state during the training process for subsequent analysis and evaluation.

Table 1. Experiment environmental parameters.

Parameter	Experimental Environment
Operating System	Windows 10
GPU	NVIDIA GeForce RTX 3090
Memory	64GB
Python	3.6.13
Deep Learning Framework	Pytorch 1.10.0/ CUDA 11.3

Table 2. Network parameters of the proposed method.

Parameter	Value
Image Size	150×150
Image Patch Size	128×128
Initial Learning Rate	0.0001
Optimizer	Adam
Weight Decay	0.0001
Learning Rate Decay Factor	0.8
Batch Size	32
Epoch	50
Window Size (M)	16
Feature Patch Size	2

Channel Number (C)	18
Block Numbers	{2,2,6}

■ **Comment 5**

More experiments should be given to show the advantage of the proposed method.

Response: Thank you for your comments. We have expanded the experimental section in Section 3 by adding experiments on real-world dataset. These expansions provide a more comprehensive validation from multiple perspectives and further demonstrate the proposed method's effectiveness and superiority in practical scenarios. Specifically, we have made the following revisions to the manuscript:

First, we have added an introduction to the real-world dataset in Section 3.1, including the source of the dataset, data characteristics, and some samples. We hope that adding this section can help readers better understand our dataset.

Second, we add an evaluation metric (point matching error) for the real-world dataset in Section 3.3. Since the real-world dataset does not contain a ground-truth homography matrix, we cannot use the evaluation metric (corner error) adopted for the synthetic benchmark dataset. We add an evaluation metric calculated using manually labeled matched point pairs to assess the performance of our method on real-world data.

Then, we restructured Section 3 by moving the introduction of the comparison method in Section 3.4 to the beginning of Section 3 to ensure the overall logical coherence and structural fluency of the manuscript. The main reason is that the experiments on both the synthetic benchmark dataset and the real-world dataset adopt the same comparison methods, so it is more reasonable to place it at the beginning of Section 3. Meanwhile, we also redescribe the structure of the section at the start of Section 3.4.

Finally, we provide a quantitative comparison of our method and other comparison methods on the real-world dataset in Section 3.5 and analyze the results for the comparison. The addition of this section can demonstrate that our method has excellent performance not only on synthetic datasets but also has good robustness and accuracy when dealing with real-world data.

We look forward to these revisions meeting your expectations and thank you again for your constructive comments. Our additions in Section 3, Section 3.1, Section 3.3, Section 3.4, and Section 3.5 are as follows (see details in lines 415-422, 432-437, 470-476, 478-480, 559-572, and 587-588 of the revised manuscript):

3. Experimental Results

In this section, we first briefly introduce the synthetic benchmark dataset and the real-world dataset, and then describe some implementation details of the proposed method. Next, we briefly present the evaluation metrics used in the synthetic benchmark dataset and the real-world dataset. Second, we perform comparisons with existing methods on synthetic benchmark datasets and real-world datasets to demonstrate the performance of our method. We compare our method with traditional feature-based methods and deep-learning-based methods. The traditional feature-based methods include eight methods that are combined by four feature descriptors (SIFT [20], ORB [22], BRISK [23], and AKAZE [24]) and two outlier rejection algorithms (RANSAC [36] and MAGSAC++ [38]). The deep learning-based methods include three methods (CADHN [43], DADHN [46], and HomoMGAN [47]). Finally, we also performed some ablation experiments to demonstrate the effectiveness of all the newly proposed components.

3.1. Dataset

Furthermore, we utilized the CVC Multimodal Stereo Dataset [52] as our real-world dataset. This collection includes 100 pairs of long-wave infrared and visible images, primarily taken on city streets, each with a resolution of 506×408. Figure 6 displays four representative image pairs from the dataset.

Figure 6. Some samples from the real-world dataset. Row 1 shows the visible images; row 2 shows the infrared images.

3.3. Evaluation Metrics

The point matching error [43, 44] is a measure of the average l_2 distance between pairs of manually labeled matching points. Lower values of this metric indicate superior performance in homography estimation. The calculation of the point matching error [43, 44] is performed as follows:

$$\mathcal{L}_p = \frac{1}{N}\sum_{i=1}^{N} \|x_i - y_i\|_2 \tag{12}$$

where x_i denotes point i transformed by the estimated homography, y_i denotes the matching point corresponding to point i, and N represents the number of manually labeled matching point pairs.

3.4. Comparison on Synthetic Benchmark Datasets

We conducted qualitative and quantitative comparisons between our method and all the comparative methods on synthetic benchmark dataset to demonstrate the performance of our method.

3.5. Comparison on Real-World Dataset

We performed a quantitative comparison with 11 methods on the real-world dataset to demonstrate the effectiveness of our method, as shown in Table 4. The evaluation results of the feature-based methods on the real-world dataset are similar to the results on the synthetic benchmark dataset, and both show varying degrees of algorithm failure and poor algorithm performance. In contrast, the deep learning-based methods perform significantly better than the feature-based methods, and no algorithm failures are observed. The proposed algorithm achieves the best performance among the deep learning-based methods; the performance of CADHN [43] and DADHN [46] is comparable with the average point matching errors of 3.46 and 3.47, respectively. Notably, our algorithm significantly improves the performance by explicitly guiding feature matching in the regression network compared to HomoMGAN [47], and the average point matching error is significantly reduced from 3.36 to 2.79. This fully

illustrates the superiority of explicitly guided feature matching compared to implicitly guided feature matching.

Table 4. Comparison of point matching error between the proposed algorithm and all other methods on the real-world dataset.

1)	**Method**	**Easy**	**Moderate**	**Hard**	**Average**	**Failure rate**
2)	$I_{3\times3}$	2.36	3.63	4.99	3.79	Nan
3)	SIFT [20] + RANSAC [36]	135.43	Nan	Nan	135.43	96%
4)	SIFT [20] + MAGSAC++ [38]	165.54	Nan	Nan	165.54	96%
5)	ORB [22] + RANSAC [36]	40.05	63.23	159.70	76.57	22%
6)	ORB [22] + MAGSAC++ [38]	61.69	109.96	496.02	158.87	27%
7)	BRISAK [23] + RANSAC [36]	44.22	81.51	483.76	151.47	24%
8)	BRISAK [23] +MAGSAC++ [38]	66.09	129.58	350.06	142.75	27%
9)	AKAZE [24] + RANSAC [36]	71.77	170.03	Nan	83.33	66%
10)	AKAZE [24] + MAGSAC++ [38]	122.64	Nan	Nan	122.64	71%
11)	CADHN [43]	2.07	3.27	4.65	3.46	0%
12)	DADHN [46]	2.10	3.27	4.66	3.47	0%
13)	HomoMGAN [47]	2.00	3.15	4.54	3.36	0%
14)	Proposed algorithm	**1.69**	**2.55**	**3.79**	**2.79**	0%