Get better 1 pixel PCK: ladder scales correspondence flow networks for remote sensing image matching in higher resolution

Weitao Chen
Alibaba Group
hillskyxm@gmail.com

Zhibin Wang
Alibaba Group
zhibin.waz@alibaba-inc.com

Hao Li
Alibaba Group
lihao.lh@alibaba-inc.com

Abstract

Recently, remote sensing image matching by deep learning reaches competitive performance evaluated by Probability of Correct Keypoints(PCK). The percentage of image size is often used as the threshold of PCK. Even though it can achieve a good 1% *PCK in high resolution by regression of transformer parameters, the value will be reduced by using the absolute 1 pixel as threshold in the higher resolution. Inspired by the flow-based methods used in natural image matching tasks, we convert the transformer to correspondence flow and propose ladder scales correspondence flow networks(LSCFN) to get better 1 pixel PCK in higher resolution. Input images are resized to multi scales and then sent to network backbone to generate multi feature pyramids. These pyramids are linked and effectively pull up the highest resolution of original backbone just like a ladder when the global correlation scale is fixed. LSCFN regress correspondence flow in ladder scales in a dense cascade way. We build LSCFN-b and LSCFN-s based on the degree of semantic change between compared images. One with only global correlation is used for the big change, another with global and local correlation is used for the opposite one. The proposed LSCFN achieve state-of-the-art performance evaluated by* 1% *of image size PCK and absolute 1 pixel PCK on google earth dataset[25].*

1. Introduction

Remote sensing image matching is a fundamental problem in remote sensing image processing. This is due to its many important applications, including change detection[42], damage assessment[5], remote sensing image fusion[9] and mosaic[43]. As is known, remote sensing images from a scene are often from different environments, such as time,sensors and viewpoints. These uncontrollable factors may cause uncontrollable changes. Semantic change and geometric change are the most common two.

Figure 1. images with slight semantic change and big geometric change.

Different applications also suffer varying degrees of semantic or geometric changes. For example,change detection often suffers the big semantic changes and slight geometric changes, remote sensing image fusion often suffers the slight semantic changes and big geometric changes. Compared with natural images, contents of remote sensing images with these changes are more difficult to understand and their features are not obvious. Although deep learning methods such as two-streams network[25] were proposed to get more obvious features and got a good 1% PCK results, their performances are limited by the sparse regression target, although they also considered the semantic changes and tried to process these changes, they did not consider the varying degrees of different changes, and although the resolution from nature images is high, it is still low for remote sensing image.

Contributions: In this paper, we propose novel ladder scales correspondence flow networks(LSCFN). It will not only get a better relative 1% PCK but also get a better absolute 1 pixel PCK. The contributions can be summarized as follows:

We convert the sparse transformer parameters to dense correspondence flow. So the sparse regression targets are converted to dense targets which are more reasonable and easier to learn. As far as we know, it's the first time that deep dense correspondence flow in this very high resolution is used for remote sensing image matching.

Figure 2. images with big semantic change and big geometric change.

We build more than one feature pyramids on basic backbone and construct a new ladder scales feature pyramid. We use this ladder to pull up the highest resolution of feature maps when the resolution of global correlation is fixed. Compared to the simple upsampling, it's a more efficient way to improve absolute 1 pixel PCK.

We consider the varying degrees of semantic changes. We propose a special network to process slight semantic changes and another to process big semantic changes. So the slight one can make use of the local correlation to get a efficient refinement for global correlation and the big one can avoid the degradation of inaccurate local correlation from local areas with different land covers caused by big semantic changes.

The proposed methods achieve the best performance on the google earth dataset used by Park et al[25].

2. Related Work

Image matching is the task to find pixel-to-pixel correspondence between images. Geometric matching and semantic matching are the two most typical sub tasks of natural image matching.Geometric matching focuses on large geometric displacements under the same scene. It suffers big geometric changes and slight semantic changes. Semantic matching poses additional challenges due to intra-class appearance and shape variations among different instances from the same object or scene category. It suffers big semantic changes. As a sub task of image matching, remote image matching may be simlar with geometric or semantic matching of natural image. But it often may suffer more complex changes as well. As showed in figure 2, the remote sensing images from different time and viewpoints suffer different land covers in the most areas under the same scene.

The common pipeline for image matching can be summed up as three stages briefly, (1) feature extraction, (2) feature matching, and (3) transformation model estimation. Remote image matching also can be solved by this pipeline. We will take a brief look at every stage below.

2.1. Feature extraction

Hand-craft features such as SIFT[21], ASIFT[24], HOG[4] and SURF[1] are used by classical methods before deep learning methods. SIFT[21] well-designed based on natural image is the most representative one, it has good performance in natural image processing tasks, because the generated feature descriptors have rotation, scaling, and translation invariance. Beyond SIFT[21], ASIFT[24] was proposed to improve fully affine invariant image comparison. However, these hand-craft features may not continue to maintain good performance in a remote sensing images task because of the different and complex imaging mechanism. To imporve these hand-craft features, methods without deep learning such as PSO-SIFT[22] were proposed. However, it's very hard to reach the performance of deep learning.

With the rapid development of deep learning and its outstanding performance in the field of computer vision, features extracted by deep neural network are more robust, powerful and transferable. In recent years, convolution neural networks such as VGGNet[33], ResNet[11] and ResNeXt[39] have been commonly used in correspondence tasks. For example, VGGNet[33] was used by CNNGeo[29], PARN[15], SAM-Net[18], DGC-Net[23] and GLU-Net[37]; ResNet[11] was used by RTNs[17], NC-Net[30] and DCCNet[12].

For remote sensing images, Ye et al[41] integrated the depth features extracted by CNN and the local features, and fused the obtained features into the PSO-SIFT[22] algorithm. Wang et al[38]transfered the pretrained CNN features in natural images by mapping function to adapt remote sensing image. Quan et al[27] used GAN[10] to automatically create more training data without manually standardizing data. Dong et al[6] designed a DescNet network extract the depth features of the image, they replaced the maxpooling operation by increasing the stride size of the convolution filter. Yang et al[40] used VGGNet[33] as the feature extractor. Kim et al[16] used ResNet[11] as their best feature extractor. Park et al[25] used ResNeXt[39] as their best feature extractor.

2.2. Feature matching

Feature matching is also an important part of the whole pipeline, the accuracy of matching features has a great impact on the entire task. KNN[8] is a representative method for feature matching and commonly works with SIFT features. Outer product is commonly used as global correlation with the features extracted by convolution neural networks. For natural images, CNNGeo[29], DGC-Net[23] and DCCNet[12] adopted this global correlation for feature matching. Based on CNNGeo[29], methods proposed by Yang et al[41], Kim et[16] and Park et al[25] for remote sensing images also adopted outer product as global correlation. Due to the big computational cost of global cor-

relation, it's often limited in a low resolution and hard to process small displacements. Prune Truong[37] combined the global correlation and local crrelation to process both large and small displacements.

Feature matching results by methods mentioned above are still coarse. In claasic methods, RANSAC[7] was proposed to filter the wrong matching results. A geometric method to evaluate the homography matrix was proposed by Song et al[34] to filter out wrong matching results. For deep matching, feature normalization[29] was used to filter coarse matching results, this normlization way also used by Yang et al[40], Park et al[25]. Furthermore, Ignacio Rocco[30] proposed a soft mutual nearest neighbor module and a neighborhood consensus network with 4D convolutions to get more accurate matching results, Prune Truong also used these methods in their GLUNET[37] and GoCOR[36]. Beyond the RANSAC[7], RANSAC-Flow[32] was proposed to do a better filter by a flow-based way.

2.3. Transformation model estimation

As is known, it's expensive to get the ground-truth of dense pixel correspondence. So classic methods estimated the transformation model by unsupervised learning. After getting the matching points from matching features by filtering and ranking, transformation parameters were learned from these matching points by least squares[2]. Methods based on spare annotations were also proposed. CNNGeo(W)[28] and NC-Net[30] were proposed to learn from sparse correspondence annotations. Shuda et al[19] improved them with an adaptive method. The optimal transport is also a good method to process sparse correspondence annotations. Liu et al[20] transported the coarse matchings with optimal to make sure an one-to-one matching. Sarlin et al[31] also used optimal transport to get a better correspondence estimation.

However, the spare annotations are still hard to get.Self-supervised learning was proposed to solve this problem efficiently. Ignacio Rocco proposed a geometric matching network[29] to regress parameters of transformations. The training datas are all generated by random affine and TPS transformations and without annotations. Beyond CNNGeo[29], Iaroslav Melekhov converted the transformation to dense correspondence flow and used DGC-Net[23] to regress the correspondence flow. Prune Truong proposed GLUNet[37] and converted the correspondence flow to optical flow to break through the limit of small resolution. For remote sensing image, self-supervised learning is also be adopted by Park et al[25], they assumed that the transformation is affine in a local area and tried to learn from a sparse global affine transformation parameters generated randomly.

3. Proposed methods

In this section, we introduce our ladder scales correspondence flow networks(LSCFN) for remote sensing image matching. We also assume that a transformations is affine in local areas.An affine transformation is represented by a vector:$[a_1, a_2, t_x, a_3, a_4, ty]$.$a_1 \sim a_4$ represent the scale, rotated angle and tilted angle, (t_x, t_y) denotes the (x-axis, y-axis) translation. We use the same datas and parameters used by Park et al[25]. To generate the dense target correspondence flow from these affine transformations, we fisrt convert the parameters into the homogeneous form:

$$[a_1, a_2, t_x, a_3, a_4, t_y] \Longrightarrow \begin{bmatrix} a_1 & a_2 & t_x \\ a_3 & a_4 & t_y \\ 0 & 0 & 1 \end{bmatrix}.$$

Figure 4. transformation of homogeneous form.

Then we convert this homogeneous matrix to correspondence flow in the resolution of the first feature map by affine grid which is introduced in STN[14]. For a pyramid structure, correspondence flow will be converted to special scales by interpolation. So we get a dense correspondence flow pyramid. Inspired by GLUNet[37], we resize the images to different resolution and get multi pyramids in each resolution. These pyramids will be linked like a ladder.

As is mentioned above, the degrees of semantic or geometric changes between the compared image are different. As is showed in figur1, the correspondence pixels are similar in the local areas when the semantic changes are slight. As is showed in figure2, the correspondence pixels represent different land covers in the most local areas. We design our networks for these differences.

3.1. ladder scale network for slight semantic change

Our ladder scale network for slight semantic change(LSCFN-s) uses VGG-16[33] as basic feature extractor backbone and builds two pyramids on this backbone to get a whole ladder feature pyramid. It is consisted of four level correlations. The coarsest level is based on a global correlation layer, followed by a mapping decoder estimating the correspondence map. Due to the similarity of correspondence pixels in local area, we can make use of the local correlation and the next three levels instead rely on local correlation layers. Firstly, correspondence flow from global correlation is converted to optical flow by un-normalization for refinemnet by local correlation. The dense optical flow is then estimated by flow decoders, taking as input the correspondence volumes resulting from the local correlation. By this coarse-to-fine way, we can get a better local correlation for small displacements. However, the coarse errors will be passed to fine layer and get no any

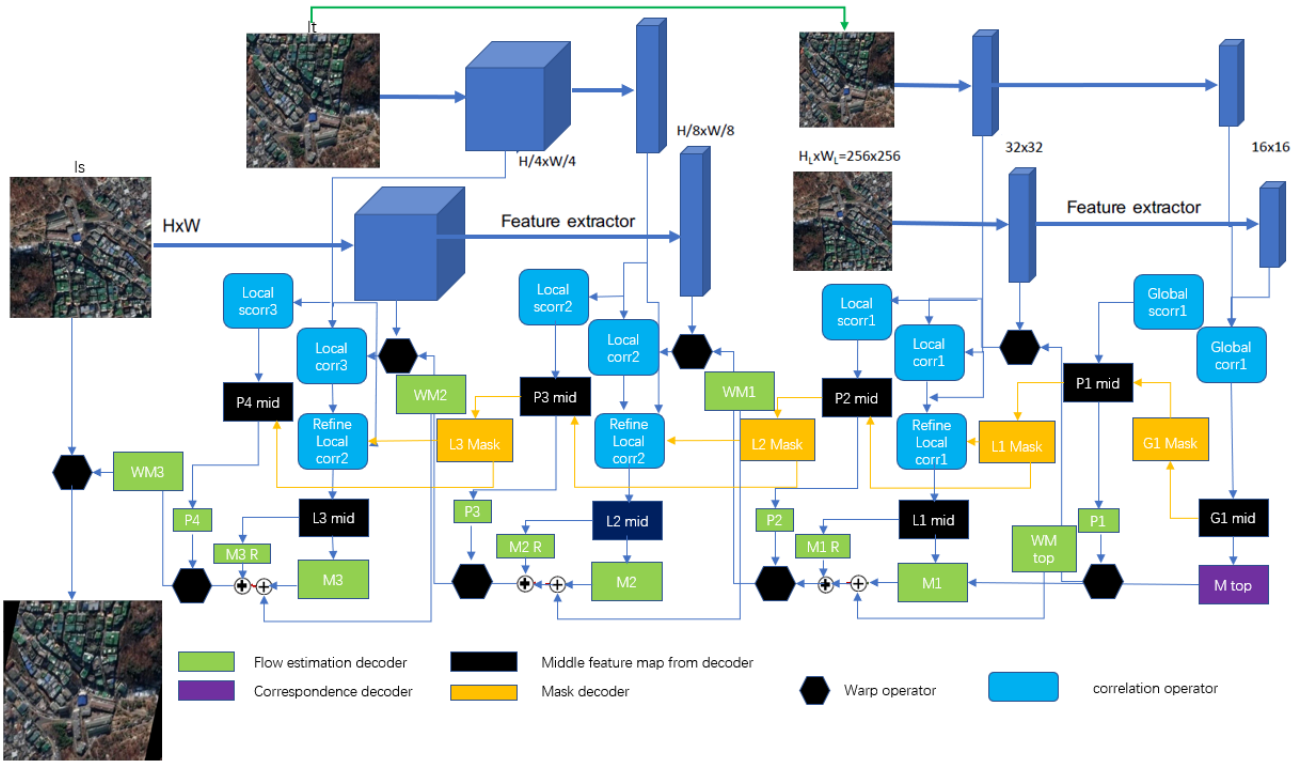

Figure 3. Architectural details of our ladder scale network for slight semantic change in remote sensing images, Yellow pipline is the mask pipline. The upsamping operator exists from coarse layer to fine layer, Inside the pyramid, we use deconvolution operator to upsample. We use the bilinear upsampling operator to upsample the flows from one pyramid to another pyramid and nearest upsampling operator with 1x1 convolution to upsample the mask from one pyramid to another pyramid, $scorr$ is the self-correlation operator.

refinement to correct. So we also build a mask pipeline to estimate the errors of correlation and optical flow. Now we introduce these parts in detail.

Global correspondence by global correlation: It is based on the global correlation which is formulated as:

$$C^l_{global}(F^l_t, F^l_s) = F^l_t(x)^T F^l_s(x^{'}) \tag{1}$$

F^l_s is the source feature map from l level and $x^{'}$ is coordinates of F^l_s, F^l_t is the target feature map from l level and x is coordinates of F^l_t. As shown in figure 3, a global mask decoder and correspondence flow decoder will be pulled from this global correlation.

Local feature matching module by local correlation: It is based on the current local correlation and last mask prediction. The original local correlation is formulated as:

$$\begin{aligned} C^{l+1}_{local} &= C^{l+1}_{local}(F_t(x); F_s(x+d)) \\ &= F^{l+1}_t(x)^T F^{l+1}_s(x+d), ||d||_\infty \leq R \end{aligned} \tag{2}$$

F^{l+1}_s is the source feature map at $l+1$ level warpped by the flow from l level, x is a coordinate in the target feature map, R is the search radius for correlation, d is the displacement from x. Furthermore, the wrong correlation results will be corrected in mask pipeline.

Flow pipeline: The correspondence flow decoder receives global correlation and an initial zero correspondence flow as inputs. It outputs a coarse correspondence flow.This correspondence flow is then converted to optical flow by unnormalization. After upsampling, it warps the next source feature map. The next warped source feature map and original target feature map will be sent to a local correlation layer. Then we get a local correlation from this correlation layer. The next optical flow decoder receives this local correlation and outputs a optical flow. This optical flow will be refined by an auxiliary optical flow decoder from the same temporary middle feature map. The refined optical flow will be added to the corase optical flow. We repeat these steps by three times. The last optical flow will be upsampled to the resolution of origianl source image and warp the source image to get final matching result.

Mask pipeline: The errors from coarse layer will be passed to fine layers in this coarse-to-fine structure. Inspired by Liteflownet3[13], we build a mask pipeline to correct these errors. In parallel with flow decoders, mask decoders are pulled to estimate the errors from coarse layer to fine layers. At first, a global mask decoder is pulled from global correlation. It shares all the same parameters with flow decoders

except for the last mask output layer. The output channel number of this layer is set to 1. After upsampling, mask at global level will mask the self-correlation of the next target feature map. The self-correlation of next target feature map is formulated as:

$$selfC^{l}_{local} = F^{l+1}_{t}(x)^{T} F^{l+1}_{t}(x + df) \tag{3}$$

We mask it as below:

$$selfC^{l}_{local} = (selfC^{l}_{local}, mask^{l}) \tag{4}$$

F^{l}_{t} is the target feature at l level, $selfC^{l}_{local}$ is the self-correlation of target feature at l level. df is the displacement from x for self-correlation, $mask^{l}$ is the prediction from mask decoder at l level, the $()$ is the concat operator.

The next mask and deformable displacement of current flow are generated from this masked self-correlation. we pull two heads from it. One predicts the next mask $mask^{(l+1)}$ and one predicts the current flow deformable displacement $disp^{l}$. The two heads share the most parameters except for the last one for effcient computation. Current level flow after upsampling will be warped to by $disp^{l}$:

$$F^{l} = disp^{l} o F^{l} \tag{5}$$

o means warp operator and this new refined flow will warp the next source feature map. This new mask pipeline combined with the original flow pipeline is showed in figure 5:

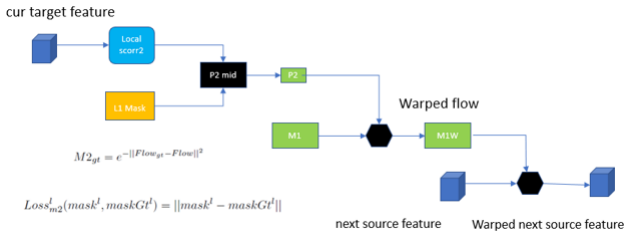

Figure 5. mask pipeline combined with flow pipeline

Next mask will firstly mask next local correlation and modulate the original correlation based on the masked correlation. Next correlation will be masked as below:

$$C^{l+1}_{mask} = (mask^{l+1}, C^{l+1}, Ft^{l+1}) \tag{6}$$

The final new correlation will be got similar with deformable convolution[3] as below :

$$C^{l+1}_{refine} = m * C^{l+1}_{mask} + p \tag{7}$$

Then it will mask the next self-correlation and refine the next flow in the same way as current level mask.

The m and p are learned from the masked correlation C^{l+1}_{mask}. This vernier module is showed in figure 6. they also share the most parameters except the last one. Except for the global correlation, all the local correlation will be masked and modulated by this way. The first local module with flow and mask pipe line is showed in figure 7. The whole LSCFN-s with flow and mask pipeline is showed in figure 3.

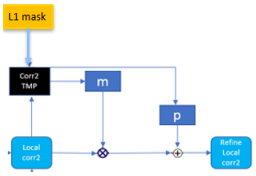

Figure 6. vernier module for correlation

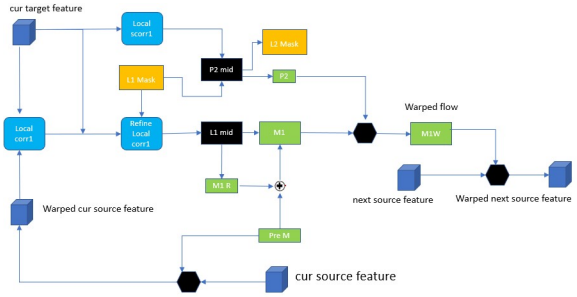

Figure 7. first local flow and mask module

3.2. ladder scale network for big semantic change

When we try our LSCFN-s on the images with big semantic change, we find it will get big degradation of PCK. Compared with the images with slight semantic change, the local correlation is hard to get the correspondence because of the land cover in the most local area is changed, local correlation will bring new wrong correspondence. So we propose a special ladder scale network for big semantic change(LSCFN-b). We also use the same backbone and build a ladder feature pyramid.

Flow pipeline: Compared with LSCFN-s, LSCFN-b adopts a different flow pipeline. All local correlation layers are removed, global correlation result will be refined by multi level correspondence flow decoders.We adopt six correspondence flow decoders. The first decoder also recives global correlation and an initial zero correspondence flow as inputs. Unlike LSCFN-s, it will not be converted to optical flow by un-normalization. After upsampling, it warps the next source feature map and is sent to next flow decoder as an input of next flow decoder. The next warped source feature map and target feature map are also sent to next flow decoder as inputs. We repeat this step from the second decoder to the last decoder. The last decoder is in the resolution of original input images. We can get a better correspondence in a higher resolution by this cascade way. Every decoder is consist of 5 convolutional blocks (Conv-BN-ReLU) and every convolutional block has different dila-

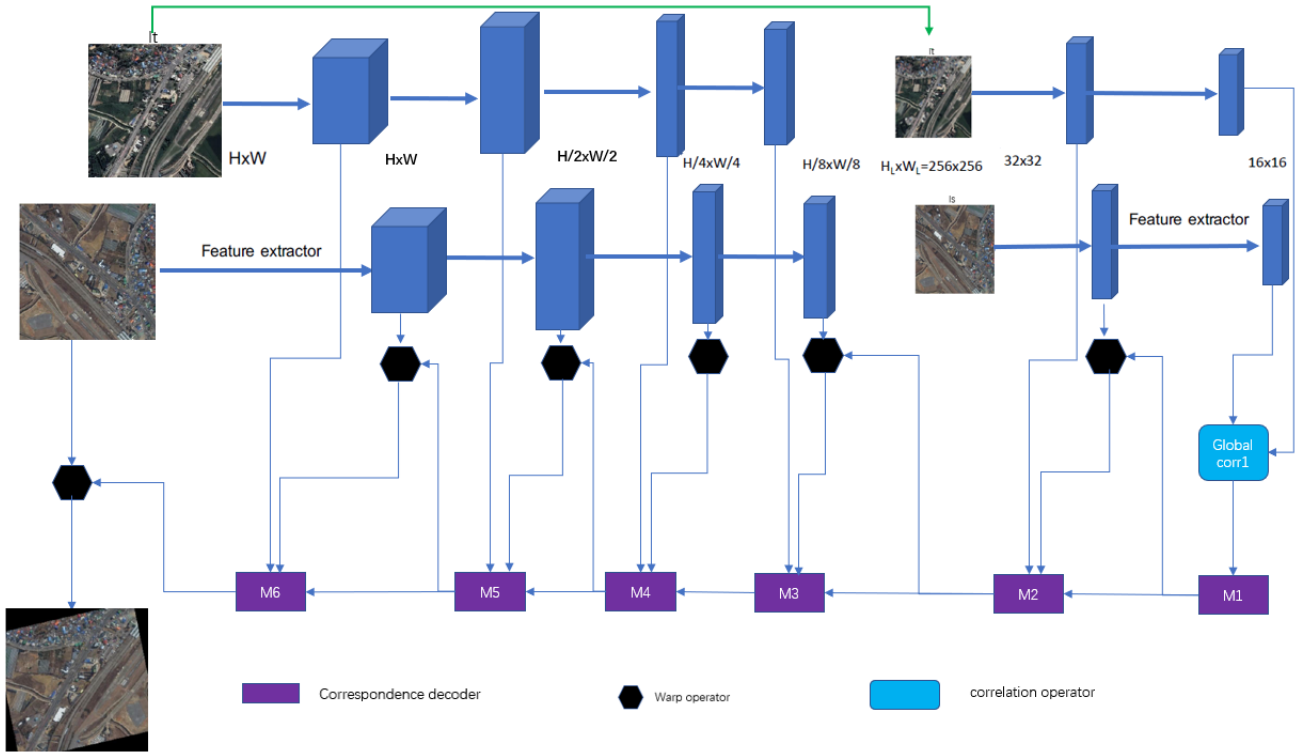

Figure 8. Architectural details of our ladder scale network flow pipeline for big semantic change in remote sensing images. We use the bilinear upsampling and concat operator to link different levels.

tion parameters for different receptive fields. All the inputs will be concated before convolution. The whole flow pipe line of LSCFN-b is showed in figure 8.

Mask pipeline: We keep the most mask pipeline used in LSCFN-s and just remove the vernier module for correlation.

3.3. Training

Loss function: Loss function of LSCFN-s is consist of optical flow loss and mask loss. We apply supervision at every pyramid level using the endpoint error (EPE) loss as flow loss. It is formulated as:

$$Loss_{of} = \sum_{l=1}^{4} (w_f^l * \sum_x ||f_{target}^l(x) - f_{pred}^l(x)||) \quad (8)$$

f_{target}^l is the ground-truth of optical flow at level l, f_{pred}^l is the prediction from flow decoder at level l, w_f^l is the flow weight at level l. x indexes over valid pixel locations.

The ground-truth of mask is generated by flow, it is formulated as:

$$M_{target}^l = e^{-||f_{target}^l - f_{pred}^l||^2} \quad (9)$$

We use the weighted L2 as the loss function of mask, it is formulated as:

$$Loss_m = \sum_{l=1}^{4} (w_m^l * ||M_{target}^l - M_{pred}^l||) \quad (10)$$

M_{pred}^l is the prediction from mask decoder at level l, w_m^l is the mask weight at level l. The final loss of LSCFN-s is formulated as:

$$Loss_{LSCFNs} = Loss_{of} + Loss_m \quad (11)$$

Loss function of LSCFN-b is consist of the correspondence flow loss and mask loss. We use the same mask loss with LSCFN-s. For correspondence flow loss, we apply supervision at every pyramid level using the L1 distance loss. It is formulated as:

$$Loss_{LSCFNb} = Loss_{cf} = \sum_{l=1}^{6} (w_f^l * \sum_x ||f_{target}^l(x) - f_{pred}^l(x)||_1) \quad (12)$$

From the ground-truth of flow, we can also get a mask pyramid estimating the existence of correspondence as below:

$$M1x_{target} = fx_{target} + X \quad (13)$$

$$M1y_{target} = Fy_{target} + Y \quad (14)$$

$$\begin{cases} M1x_{target} = 1 & 0 < M1x_{target} < w \\ M1x_{target} = 0 & M1x_{target} <= 0 \quad M1x_{target} >= w \end{cases} \tag{15}$$

$$\begin{cases} M1y_{target} = 1 & 0 < M1y_{target} < h \\ M1y_{target} = 0 & M1y_{target} <= 0 \quad M1y_{target} >= h, \end{cases} \tag{16}$$

$$M1_{target} = M1x_{target} \cap M1y_{target} \tag{17}$$

The fx_{target} and fy_{target} are flow ground-truth on x and y axis. X and Y are the sorted range of width and height. w and h are the width and height of the current level resolution. We mask the original flow loss by this mask. The new flow loss are formulated as:

$$Loss_{of} = \sum_{l=1}^{4} (w_f^l * \sum_x M1_{target}^l * ||f_{target}^l(x) - f_{pred}^l(x)||) \tag{18}$$

$$Loss_{cf} = \sum_{l=1}^{6} (w_f^l * \sum_x M1_{target}^l * ||f_{target}^l(x) - f_{pred}^l(x)||_1) \tag{19}$$

Common training settings: Our model is implemented by Pytorch[26]. We freeze the weights of backbone when training on natural images and unfreeze the weights of backbone when training on remote sensing images. We employ a search radius R=4 for local correlation. For self-correlation of LSCFN-s, we employ differents search radius. At global level, we empoly R=4. At first local level, we employ R=3. At second local level, we employ R=2. For self-correlation of LSCFN-b, we adopt $[4, 4, 3, 3, 2, 2]$ as search radius set from first level to last level. We use a batch size of 16 for LSCFN-s and LSCFN-b. The initial learning rate for LSCFN-s is 0.0001, we change it by three steps. The initial learning rate for LSCFN-s is 0.00002, we change it by two steps. We train LSCFN-s 100 epoches and LSCFN-b 200 epoches in total. We use one V100 GPU card for LSCFN-s and four V100 GPU cards for LSCFN-b.

4. experiments

In this section, we evaluate the LSCFN-s and LSCFN-b on the dataset used by Park et al[25]. This dataset is from the synthetic affine transformation, when a image is fixed, the synthetic affine transformation is applied on another corresponding image. We use PCK as the evaluation metrics. It is formulated as:

$$PCK = \frac{\sum_{i=0}^{n} 1(D(\tilde{p_i}, p_i) < \sigma)}{\sum_{i=0}^{n} |\tilde{p_i}|} \tag{20}$$

$\tilde{p_i}$ is the point warped by flow, p_i is the correspondence ground-truth. D is the distance of $\tilde{p_i}$ and p_i. L2 distance is often used as this distance, σ is the threshold. Percent of the max image size is often used as this threshold:

$$\sigma = \tau * max(h, w) \tag{21}$$

When $\tau = 0.01$, we denote PCK as PCK-01, when $\tau = 0.03$, we denote PCK as PCK-03, when $\tau = 0.05$, we denote PCK as PCK-05, when $\sigma = 1$, we denote PCK as PCK-1px. The higher τ means the higher σ, the higher σ means more correct points. When resolution changes, we can also get that relative PCK will be easier to keep than the absolute one from equation 21. So PCK-1px is the strictest metric. In our experiments on google earth dataset, we adopt these four settings for PCK.

4.1. experiments of LSCFN-s

The slight semantic changes are not considered before, so we build a new dataset for training and evalution based on the dataset used for big semantic changes. This new dataset shares the target images and transformations with the original google dataset[25]. We just change the original source images and let the original target images be the new original source images. The transformations are applied on these new original source images to get the final source images.

As is showed in table 1, we get a very high PCK on this new dataset. The height and width of image are both 520.

Methods	backbone	PCK-01	PCK-03	PCK-05	PCK-1px
CNNGeo+Int.Aug.+Bi-En[25].	SE-ResNeXt101	73.05	99.0	99.91	2.35
LSCFN-s	VGG-16	99.963	99.989	99.997	98.593

Table 1. PCK [%] obtained by LSCFN-s on google earth dataset.

We compare LSCFN-s with CNNGeo+Int.Aug.+Bi-En[25] and use the best pretrained model with SE-ResNeXt101 backbone. Our method shows powerful results with stricter distances. Furthermore, we try our LSCFN-s on TSS[35] about semantic correspondence task of natural image, we also get a competitive result without using inefficient neighborhood consensus network with 4D convolution. The detail results are showed on table 2.

Methods	Feature backbone	FG3DCAR	JOBS	PASCAL	AVG
CNNGeo(w)[28]	ResNet-101	90.3	76.4	56.5	74.4
RTNs[17]	ResNet-101	90.1	78.2	63.3	77.2
PARN[15]	VGG-16	87.6	71.6	68.8	76.0
PARN[15]	ResNet-101	89.5	75.9	71.2	78.8
NC-Net[30]	ResNet-101	94.5	81.4	57.1	77.7
DCCNet[12]	ResNet-101	93.5	82.6	57.6	77.9
SAM-Net[18]	VGG-19	96.1	82.2	67.2	81.8
GLU-Net[37]	VGG-16	93.2	73.3	71.1	79.2
Semantic-GLU-Net[37]	VGG-16	94.4	75.5	78.3	82.8
LSCFN-s(ours)	VGG-16	93.0	76.6	77.0	82.2

Table 2. PCK [%] obtained by different state-of-the-art methods on TSS for the task of semantic matching.

4.2. experiments of LSCFN-b

We first prove the effectiveness of this cascade way in the resolution of 240. We build sub networks in sub scale pyramids. The sub scale pyramids are showed in table 3.

The resolution below 240 will be upsamped to 240. Because LSCFN-b is slow for convergence. So for fair and efficient comparison, we choose the epoches with similar average loss when the networks get some enough convergence. In practice, the average loss is 0.232 for sub nets and the average loss is 0.235 for LSCFN-b. The results in

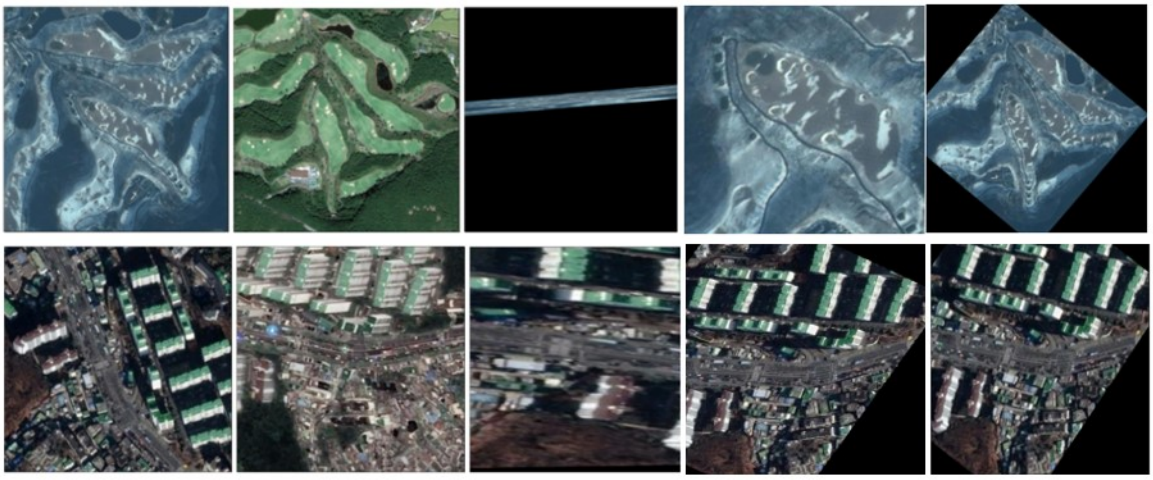

Figure 9. the images showed in first column are source images, the images showed in second column are target images, the image showed in third column are the results of SIFT with RANSAC, the images showed in forth column are the results of CNNGeo+Int.Aug.+Bi-En, the images showed in fifth column are the results of LSCFN-b.

scale	scale pyramid
15	[15]
30	[15, 30]
60	[15, 30, 60]
120	[15, 30, 60, 120]
240	[15, 30, 60, 120, 240]

Table 3. scale and its pyraimd.

detail are showed in table 4. It is proved that the higher

scale	PCK-01	PCK-03	PCK-05	PCK-1px
15	16.463	62.091	82.917	3.313
30	37.982	88.922	94.647	8.084
60	73.101	93.758	96.232	26.229
120	76.146	95.073	96.967	29.809
240	90.076	96.678	97.603	61.449

Table 4. PCK [%] obtained by sub networks of LSCFN-b in 240 resolution on google earth dataset.

PCK is from the higher scale, PCK-1px is lower than PCK-01. Furthermore, we try LSCFN-b in 520 resolution and upsample the results in 240 resolution to 520 resolution to compare. The results in detail are showed in table 5.

scale	PCK-01	PCK-03	PCK-05	PCK-1px
240 up	90.385	96.517	97.517	22.665
520	89.509	99.205	99.718	26.343

Table 5. PCK [%] obtained by sub networks of LSCFN-b in 520 resolution on google earth dataset.

PCK-01 upsampled from 240 with a slightly lower average loss is a little higher than PCK-01 in 520 resolution, but PCK-03, PCK-05 and PCK-1px are lower. Based on this experiment, we keep on training LSCFN-b in 520 resolution for a better convergence. Finally We compare it with other methods. The results are showed in table 6.

method	backbone	PCK-01	PCK-03	PCK-05	PCK-1px
SURF[1]	-	15.3	23.1	26.7	-
SIFT[21]	-	33.7	45.9	51.2	-
ASIFT[24]	-	37.9	57.9	64.8	-
OA-Match[34]	-	38.2	57.8	64.9	-
CNNGeo[29]	ResNet101	27.6	76.2	90.6	-
CNNGeo+Int.Aug.+Bi-En[25].	ResNet101	35.1	82.5	93.8	2.2
CNNGeo+Int.Aug.+Bi-En[25].	SE-ResNeXt101	48.0,	91.1,	97.1	3.7
LSCFN-b(ours)	VGG16	95.7	99.7	99.9	49.0

Table 6. PCK [%] obtained by different methods on google earth dataset.

Beyond all the former methods, LSCFN-b gets a very high PCK-01 and gets a better PCK-1px in higher resolution. So it's more practical. The limit by learning from sparse annotation is broke through. We also show a powerful result demo of LSCFN-b in figure 9, most other methods fails on the pair of images showed in first row.

5. Conclusion

In this work, we propose ladder scales correspondence flow networks to learn from dense ground-truth of flow by self-supervised learning in high resolution. We consider the different degree of semantic changes. LSCFN-s is used for slight semantic change and LSCFN-b is used for big semantic change. We build a new dataset for slight semantic change. The experimental results on TSS show our methods are universal methods for image matching. The experimental results on google earth dataset show our methods achieve the best comprehensive performance for remote sensing image matching.

References

[1] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *In ECCV*, pages 404–417, 2006.

[2] A. Charnes, E. L. Frome, and P. L. Yu. The equivalence of generalized least squares and maximum likelihood estimates in the exponential family. *Journal of the American Statistical Association*, 71(353):169–171, 1976.

[3] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, pages 764–773. IEEE Computer Society, 2017.

[4] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893 vol. 1, 2005.

[5] F. Dell'Acqua and P. Gamba. Remote sensing and earthquake damage assessment: Experiences, limits, and perspectives. *Proceedings of the IEEE*, 100(10):2876–2890, 2012.

[6] Yunyun Dong, Weili Jiao, Tengfei Long, Lanfa Liu, Guojin He, Chengjuan Gong, and Yantao Guo. Local deep descriptor for remote sensing image feature matching. *Remote. Sens.*, 11(4):430, 2019.

[7] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[8] E. Fix and J.L. Hodges. *Discriminatory Analysis: Nonparametric Discrimination: Consistency Properties*. USAF School of Aviation Medicine, 1951.

[9] Hassan Ghassemian. A review of remote sensing image fusion methods. *Information Fusion*, 32:75–89, 2016.

[10] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *NIPS*, pages 2672–2680, 2014.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778. IEEE Computer Society, 2016.

[12] Shuaiyi Huang, Qiuyue Wang, Songyang Zhang, Shipeng Yan, and Xuming He. Dynamic context correspondence network for semantic alignment. In *ICCV*, pages 2010–2019. IEEE, 2019.

[13] Tak-Wai Hui and Chen Change Loy. Liteflownet3: Resolving correspondence ambiguity for more accurate optical flow estimation. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *ECCV (20)*, volume 12365 of *Lecture Notes in Computer Science*, pages 169–184. Springer, 2020.

[14] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *NIPS*, pages 2017–2025, 2015.

[15] Sangryul Jeon, Seungryong Kim, Dongbo Min, and Kwanghoon Sohn. Parn: Pyramidal affine regression networks for dense semantic correspondence. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *ECCV (6)*, volume 11210 of *Lecture Notes in Computer Science*, pages 355–371. Springer, 2018.

[16] Dong-Geon Kim, Woo-Jeoung Nam, and Seong-Whan Lee. A robust matching network for gradually estimating geometric transformation on remote sensing imagery. In *SMC*, pages 3889–3894. IEEE, 2019.

[17] Seungryong Kim, Stephen Lin, Sangryul Jeon, Dongbo Min, and Kwanghoon Sohn. Recurrent transformer networks for semantic correspondence. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *NeurIPS*, pages 6129–6139, 2018.

[18] Seungryong Kim, Dongbo Min, Somi Jeong, Sunok Kim, Sangryul Jeon, and Kwanghoon Sohn. Semantic attribute matching networks. In *CVPR*, pages 12339–12348. Computer Vision Foundation / IEEE, 2019.

[19] Shuda Li, Kai Han, Theo W. Costain, Henry Howard-Jenkins, and Victor Prisacariu. Correspondence networks with adaptive neighbourhood consensus. In *CVPR*, pages 10193–10202. IEEE, 2020.

[20] Yanbin Liu, Linchao Zhu, Makoto Yamada, and Yi Yang. Semantic correspondence as an optimal transport problem. In *CVPR*, pages 4462–4471. IEEE, 2020.

[21] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, Nov. 2004.

[22] Wenping Ma, Zelian Wen, Yue Wu, Licheng Jiao, Maoguo Gong, Yafei Zheng, and Liang Liu. Remote sensing image registration with modified sift and enhanced feature matching. *IEEE Geosci. Remote. Sens. Lett.*, 14(1):3–7, 2017.

[23] Iaroslav Melekhov, Aleksei Tiulpin, Torsten Sattler, Marc Pollefeys, Esa Rahtu, and Juho Kannala. Dgc-net: Dense geometric correspondence network. In *WACV*, pages 1034–1042. IEEE, 2019.

[24] Jean-Michel Morel and Guoshen Yu. Asift: A new framework for fully affine invariant image comparison. *SIAM J. Imaging Sci.*, 2(2):438–469, 2009.

[25] Jae-Hyun Park, Woo-Jeoung Nam, and Seong-Whan Lee. A two-stream symmetric network with bidirectional ensemble for aerial image matching. *CoRR*, abs/2002.01325, 2020.

[26] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[27] Dou Quan, Shuang Wang, Xuefeng Liang, Ruojing Wang, Shuai Fang, Biao Hou, and Licheng Jiao. Deep generative

matching network for optical and sar image registration. In *IGARSS*, pages 6215–6218. IEEE, 2018.

[28] Ignacio Rocco, Relja Arandjelovic, and Josef Sivic. End-to-end weakly-supervised semantic alignment. In *CVPR*, pages 6917–6925. IEEE Computer Society, 2018.

[29] Ignacio Rocco, Relja Arandjelovic, and Josef Sivic. Convolutional neural network architecture for geometric matching. *IEEE Trans. Pattern Anal. Mach. Intell.*, 41(11):2553–2567, 2019.

[30] Ignacio Rocco, Mircea Cimpoi, Relja Arandjelovic, Akihiko Torii, Tomás Pajdla, and Josef Sivic. Neighbourhood consensus networks. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *NeurIPS*, pages 1658–1669, 2018.

[31] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. *CoRR*, abs/1911.11763, 2019.

[32] Xi Shen, François Darmon, Alexei A Efros, and Mathieu Aubry. Ransac-flow: generic two-stage image alignment. In *16th European Conference on Computer Vision*.

[33] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2014. cite arxiv:1409.1556.

[34] Woo-Hyuck Song, Honggyu Jung, In-Youb Gwak, and Seong-Whan Lee. Oblique aerial image matching based on iterative simulation and homography evaluation. *Pattern Recognit.*, 87:317–331, 2019.

[35] Tatsunori Taniai, Sudipta N. Sinha, and Yoichi Sato. Joint recovery of dense correspondence and cosegmentation in two images. In *CVPR*, pages 4246–4255. IEEE Computer Society, 2016.

[36] Prune Truong, Martin Danelljan, Luc Van Gool, and Radu Timofte. Gocor: Bringing globally optimized correspondence volumes into your neural network. *CoRR*, abs/2009.07823, 2020.

[37] Prune Truong, Martin Danelljan, and Radu Timofte. Glu-net: Global-local universal network for dense flow and correspondences. In *CVPR*, pages 6257–6267. IEEE, 2020.

[38] Shuang Wang, Dou Quan, Xuefeng Liang, Mengdan Ning, Yanhe Guo, and Licheng Jiao. A deep learning framework for remote sensing image registration. *ISPRS Journal of Photogrammetry and Remote Sensing*, 145:148–164, 2018. Deep Learning RS Data.

[39] Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, pages 5987–5995. IEEE Computer Society, 2017.

[40] Z. Yang, T. Dan, and Y. Yang. Multi-temporal remote sensing image registration using deep convolutional features. *IEEE Access*, 6:38544–38555, 2018.

[41] Famao Ye, Yanfei Su, Hui Xiao, Xuqing Zhao, and Weidong Min. Remote sensing image registration using convolutional neural network features. *IEEE Geosci. Remote. Sens. Lett.*, 15(2):232–236, 2018.

[42] Puzhao Zhang, Maoguo Gong, Linzhi Su, Jia Liu, and Zhizhou Li. Change detection based on deep feature representation and mapping transformation for multi-spatial-resolution remote sensing images. *ISPRS Journal of Photogrammetry and Remote Sensing*, 116:24–41, 2016.

[43] Y. Zhen, Z. Sun, J. Li, and Y. Peng. An airborne remote sensing image mosaic algorithm based on feature points. In *2016 Sixth International Conference on Instrumentation Measurement, Computer, Communication and Control (IMCCC)*, pages 202–205, 2016.