S³-Match: Common-View Aligned Image Matching via Self-Supervised Keypoint Selection

Supplementary Material

6.1 Linear Transformer

The Transformer is a model architecture designed for sequence modeling, originally implemented primarily in natural language processing tasks, but also applicable across various domains. The self-attention mechanism is a key component of the Transformer. In this mechanism, the input sequence is utilized as query, key, and value vectors. An attention weight matrix is obtained by computing the dot product of the query and key, which is then used for weighted averaging of the values. Consequently, the output vector at each position can consider all positions in the sequence, thereby capturing global dependencies. The self-attention mechanism is capable of effectively capturing long-range dependencies without the reliance on a fixed-sized sliding window.

In the Transformer, the attention mechanism can be expressed as:

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{12}$$

where Q represents the query matrix, K the key matrix, V the value matrix, and d_k the dimension of the query and key. The application of distinct linear transformations to the query, key, and value separately allows the realization of a multi-head attention mechanism, enabling the model to focus on different semantic information.

Assuming the lengths of Q and K are N, and the feature dimension is D, the dot product operation between Q and K in the Transformer introduces a quadratic computation complexity ($O(N^2)$) with the length of the input sequence. Even when we partition the image into larger patchs effectively reducing the length N, applying a traditional Transformer model directly would still lead to significant computational complexity. To address this issue, we adopt an efficient linear Transformer variant proposed in LoFTR to replace the standard attention layer in the Transformer.

The Linear Transformer introduces a method to replace the kernel function, replacing the exponential kernel function used in the original attention layer with another kernel function:

$$\text{sim}(Q,K) = \phi(Q)\cdot\phi(K)^T, \phi(\cdot) = \text{elu}(\cdot)+1. \tag{13}$$

By leveraging the associativity property of matrix multiplication, we can perform multiplication between $\phi(K)^T$ and V first. Given that $D \ll N$, the computational complexity is reduced to $O(N)$.

6.2 Supervision of Common-View Aligned Bottleneck Features

S³-Match performs a flattening operation on the deepest bottleneck feature maps of the U-Net, represented as F^A and F^B, to execute the cross-attention mechanism. For a feature vector F_i^A from the set of feature points F^A in image I^A, it represents the feature information of a pixel patch in the original image. The goal at this stage is to make the features contained within this patch as similar as possible to the features of the corresponding patch in I^B. To achieve this matching, the following steps are undertaken to compute a matching probability matrix $\mathcal{P}$ for I^A and I^B:

$$\mathcal{C}(i,j) = \frac{1}{\tau} \cdot \langle F^A(i), F^B(j) \rangle, \tag{14}$$

$$\mathcal{P}^1(i,j) = \text{softmax}_1\left(\mathcal{C}(i,\cdot)\right)_j, \tag{15}$$

$$\mathcal{P}^2(i,j) = \text{softmax}_1\left(\mathcal{C}(\cdot,j)\right)_i. \tag{16}$$

Here, $\mathcal{C}(i,j)$ denotes the feature similarity between point i from F^A and point j in F^B, which is scaled using a temperature parameter τ.

To effectively supervise the generation of cross-image aligned features, we introduces two symmetrical Focal loss functions, defined as follows:

$$\begin{aligned}\mathcal{L}_{align} =& \frac{1}{N_{\text{pos}}} \sum_{i:\widetilde{\mathcal{P}^1}_i=1} \left(-(1-\mathcal{P}^1_i)^2 \log(\mathcal{P}^1_i)\right) + \frac{1}{N_{\text{neg}}} \sum_{i:\widetilde{\mathcal{P}^1}_i=0} \left(-(\mathcal{P}^1_i)^2 \log(1-\mathcal{P}^1_i)\right) + \\ & \frac{1}{N_{\text{pos}}} \sum_{i:\widetilde{\mathcal{P}^2}_i=1} \left(-(1-\mathcal{P}^2_i)^2 \log(\mathcal{P}^2_i)\right) + \frac{1}{N_{\text{neg}}} \sum_{i:\widetilde{\mathcal{P}^2}_i=0} \left(-(\mathcal{P}^2_i)^2 \log(1-\mathcal{P}^2_i)\right).\end{aligned} \tag{17}$$

Here, $\widetilde{\mathcal{P}^1}$ and $\widetilde{\mathcal{P}^2}$ represent the ground truth of the matching matrices, with the variable $i = \{x,y\}$ succinctly indicating the two-dimensional coordinate positions within the matrices. As we iterate through the blocks in I^A, when a block a in I^A is the closest match to block b in I^B, $\widetilde{\mathcal{P}^1}_{\{a,b\}} = 1$; the logic for $\widetilde{\mathcal{P}^2}$ involves iterating the blocks in I^B.

6.3 Supervision of Multi-Scale Fusion Descriptors

Similar to the aforementioned loss functions, we employ Focal loss to supervise the mutual nearest-neighbor matching of feature points:

$$\mathcal{L}_{desc} = \frac{1}{N_{\text{pos}}} \sum_{i:\widetilde{\mathcal{Q}}_i=1} \left(-(1-\mathcal{Q}_i)^2 \log(\mathcal{Q}_i)\right) + \frac{1}{10 \cdot N_{\text{neg}}} \sum_{i:\widetilde{\mathcal{Q}}_i=0} \left(-\mathcal{Q}_i^2 \log(1-\mathcal{Q}_i)\right). \tag{18}$$

In this context, $\widetilde{\mathcal{Q}}$ represents the ground truth matching matrix for the feature points. When the ath feature point in I^A and the bth feature point in I^B are mutual nearest neighbors with a discrepancy of no more than $2px$, then $\widetilde{\mathcal{Q}_{\{a,b\}}} = 1$. Due to the potential absence of ground truth data for depth maps in some edge areas, some positive labels might inadvertently be treated as negative. To address this, we assigns a lower weight to negative labels to mitigate any potential bias.

6.4 More Qualitative Analysis of Keypoint Selection

Figure 7 illustrates the results of feature point detection. This figure displays the effectiveness of keypoint selection in images I_A and I_B. Utilizing the camera pose and depth information available in the dataset, keypoints from I_B are projected in red onto I_A and displayed in the lower left corner of the image. When the same geometric locations in both images are identified as keypoints, the overlay of red and green appears yellow. For comparison, the lower right corner shows the repeatability of keypoints detected by the SuperPoint algorithm. Compared to SuperPoint, the S³-Match algorithm focuses more on detecting keypoints in the common-view areas of the images and makes more effective use of image texture information, resulting in higher detection rates and better distribution uniformity of keypoints. Additionally, S³-Match significantly outperforms the SuperPoint algorithm in terms of keypoint repeatability.

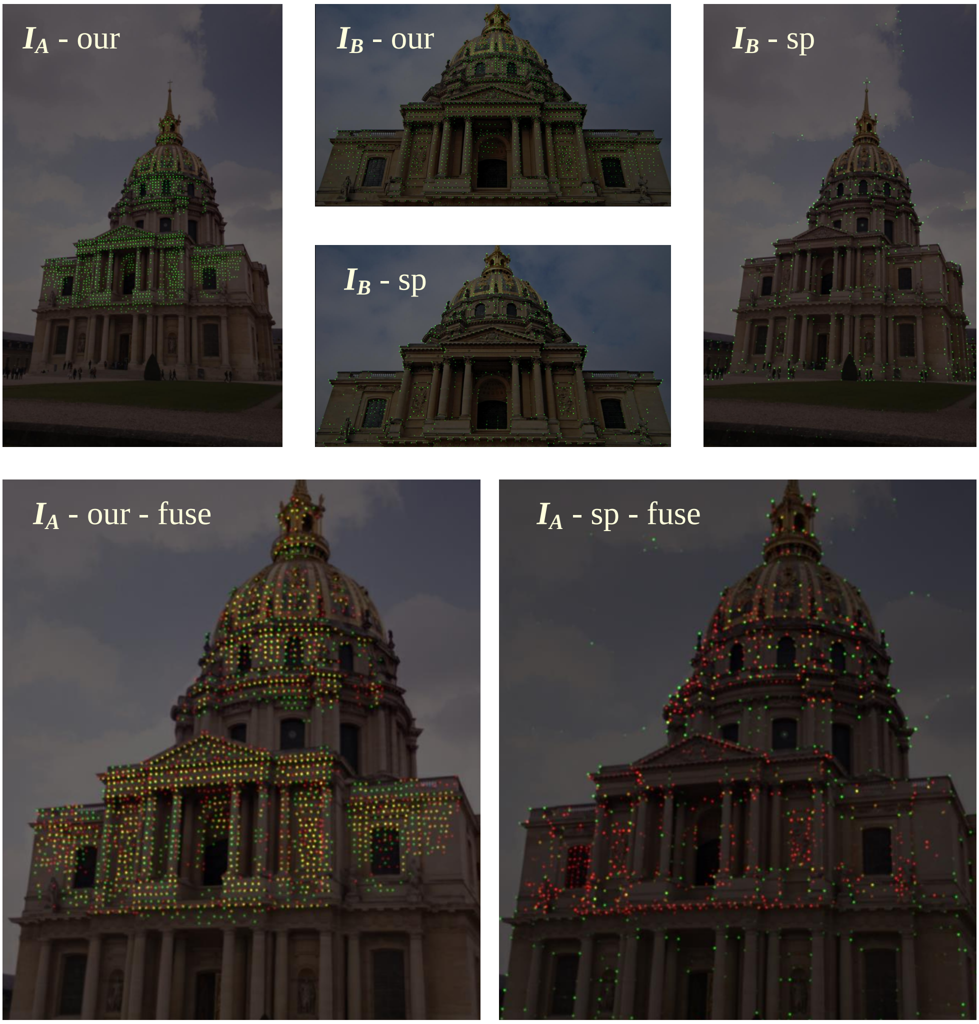

Figure 7: **Feature point detection results in images with significant perspective changes.** The figure illustrates the keypoint detection results for S³-Match on images A (I_A-our) and B (I_B-our), and for SuperPoint [5] on images A (I_A-sp) and B (I_B-sp). Additionally, it shows an integrated result where keypoints from image A are projected onto image B in red (I_B-our-fuse), alongside a comparative integration result from SuperPoint (I_B-sp-fuse).

Figure 8 showcases another set of keypoint detection results. S³-Match is able to extract

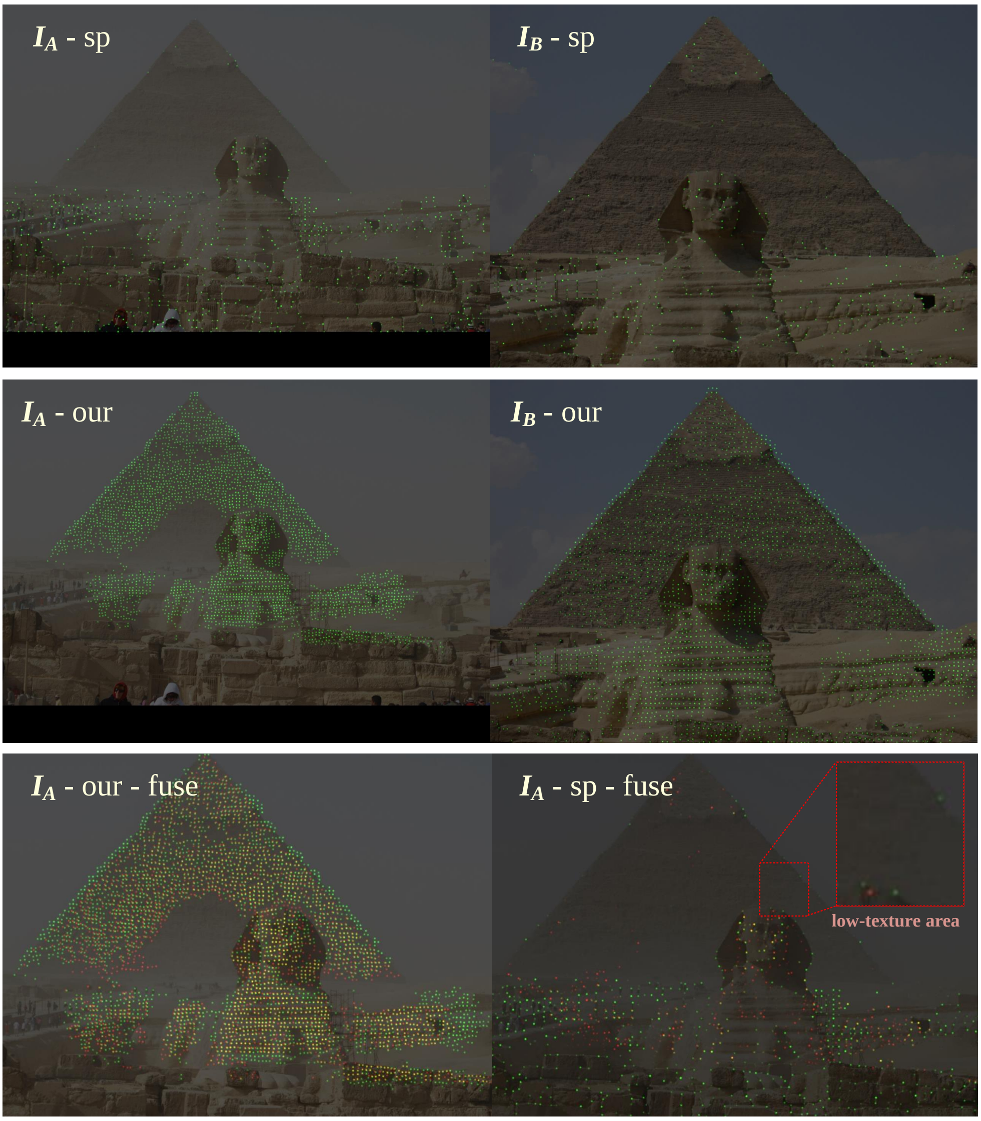

Figure 8: Feature point detection results in images with weak textures.

reliable and highly repeatable keypoints in areas of weak texture (marked with red boxes), leveraging the various subtle texture details contained within the image more fully than traditional corner detection methods. In Figure 9, additional results of keypoint detection are presented.

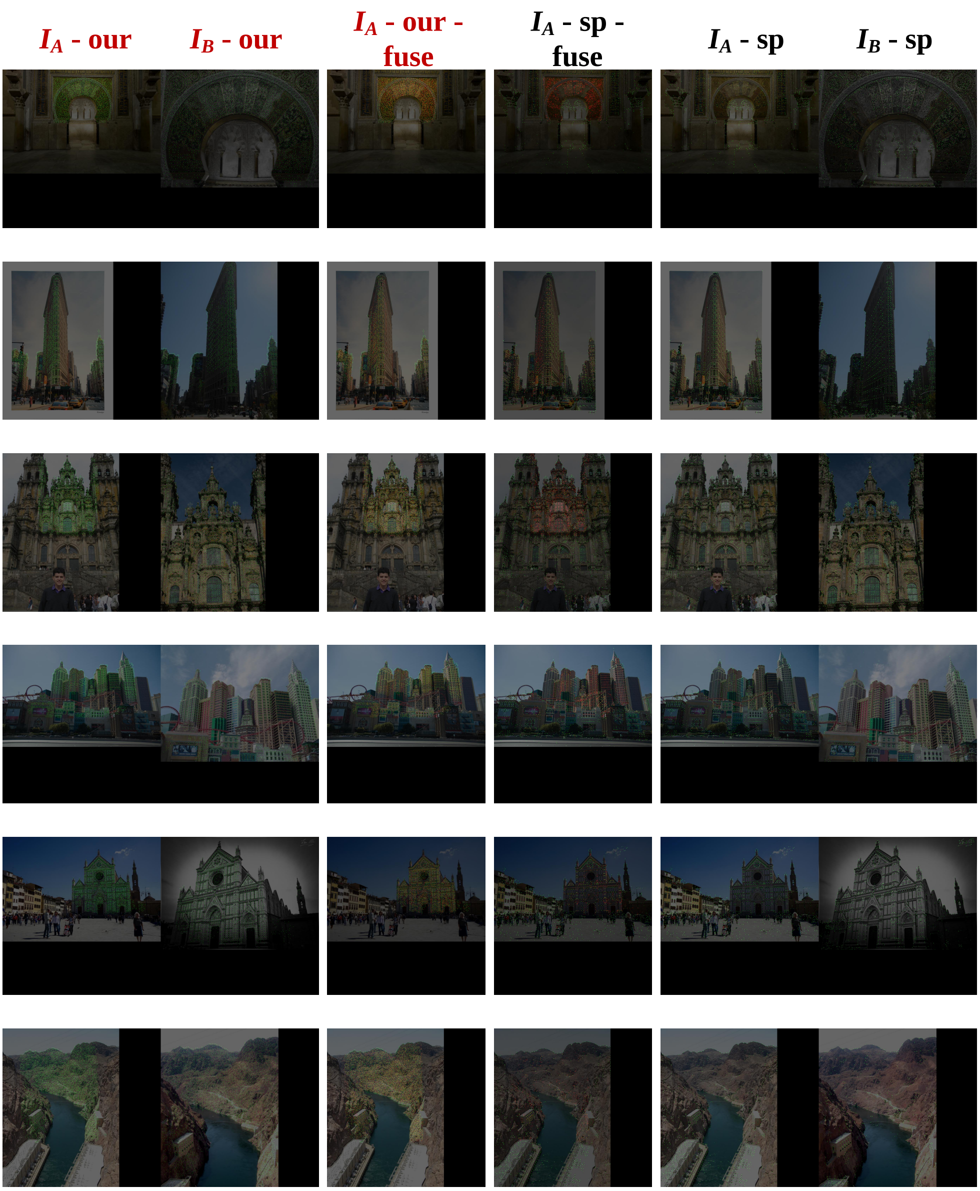

Figure 9: Additional feature point detection results.