EDM: Efficient Deep Feature Matching

Supplementary Material

6. Implementation Details

6.1. Training Details

We adopt a multi-step training strategy to train our EDM model on the MegaDepth dataset for a total of 30 epochs. The training begins with an initial learning rate of $2e$-3, which undergoes a linear warmup phase lasting 3 epochs. Following this, the learning rate is reduced by half every 4 epochs, starting from the eighth epoch. The learning rate curve is depicted in Fig. 6.

Additionally, the supervision of fine matching relies on the predictions derived from coarse matching. However, in the early training stages, the accuracy of these coarse matching predictions may be unsatisfactory. In order to avoid distributed data parallel deadlocks and enhance the supervision of fine matching, we pad the training samples with an additional 32 ground truth coarse matches for those with inadequate coarse matching predictions.

7. More Experiments

7.1. CNN Backbone

We utilize a simple variant of ResNet18 [21] as our backbone, which achieves a minimum resolution of $\frac{1}{32}$. To investigate the impact of various channel configurations, we conducted experiments on the MegaDepth dataset, as shown in Tab. 8. The final channel configuration we selected, $[32, 64, 128, 256, 256]$, offers an optimal balance between efficiency and performance.

7.2. Fine Matching Selection

As presented in Eq. (7), during the training process, on the one hand, it is desirable for the fine-level loss function $\mathcal{L}_f$ to constrain the value of σ in term $\log \sigma$ to be as small as possible. On the other hand, in term $\log \mathcal{Q}_\phi(\hat{x})$, when the distance between the predicted mean μ and the ground truth mean μ^{gt} is large, the standard deviation σ of the predict distribution tends to increase in order to mitigate the overall loss. As demonstrated in Fig. 7, the matches that exhibit higher confidence $\mathcal{P}_f$ (indicated by a smaller σ) are frequently found in image regions that contain abundant local details. This observation suggests that the model leverages these detailed areas to make more precise and confident predictions.

7.3. SCC Bins

The Soft Coordinate Classification (SCC) bins number N in the Axis-Based Regression Head (ABRHead) is set to 16

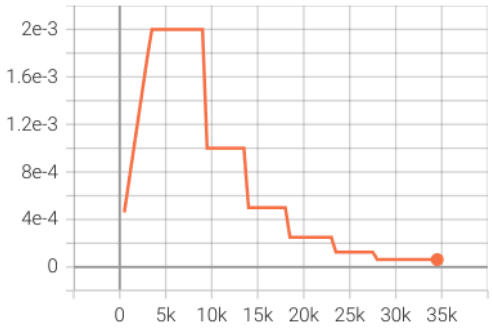

Figure 6. Learning rate curve over iterations.

backbone channels	Pose Estimation AUC			Time (ms)
	@5°	@10°	@20°	
$[16, 32, 64, 128, 256]$	55.9	71.8	83.1	77.6
$[32, 64, 128, 128, 256]$	57.3	73.0	84.0	84.2
$[32, 64, 128, 256, 256]$	57.5	73.2	84.2	86.0
$[32, 64, 128, 256, 512]$	57.8	73.1	84.0	96.0
$[64, 128, 256, 256, 256]$	58.0	73.5	84.3	109.3

Table 8. The results of varying backbone channel numbers, from $\frac{1}{2}$ to $\frac{1}{32}$ scale, on the MegaDepth dataset.

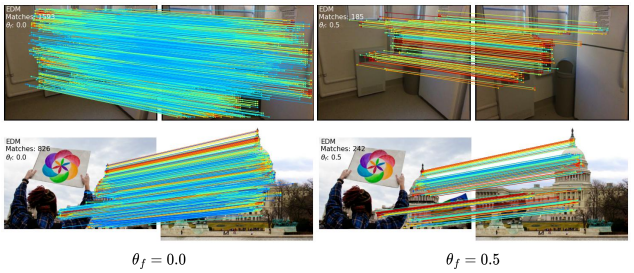

Figure 7. Impact of θ_f on fine-level matching filtering. Matches in areas with obvious details tend to have smaller σ, indicating higher confidence in these results.

in our EDM. Considering the X-axis as an illustrative example, during the fine-level matching, each 8×8 grid along the X-axis is divided into N bins, so each pixel within this grid is mapped to $\frac{N}{8}$ bins. The Fig. 8 demonstrates a trend in matching accuracy as N varies. Initially, as N increases, the accuracy of matching improves, benefiting from the finer segmentation of pixels into bins. However, as N continues to grow, the complexity of the learning task also increases, leading to a gradual decline in matching accuracy. Detailed experimental results supporting this observation are provided in Tab. 9. Additionally, the differences in network

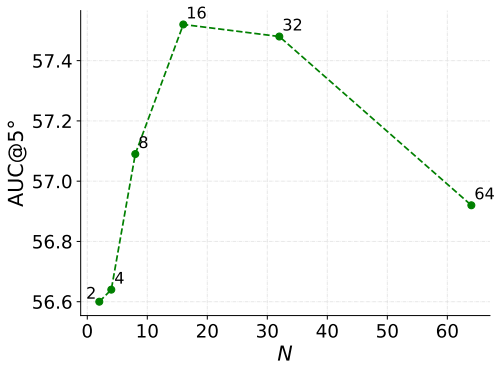

Figure 8. The impact of N on matching accuracy.

N	Pose Estimation AUC		
	@5°	@10°	@20°
2	56.60	72.34	83.56
4	56.64	72.81	83.89
8	57.09	72.75	83.75
16	**57.53**	**73.21**	**84.20**
32	57.48	73.02	84.19
64	56.92	72.56	83.60

Table 9. The results of different N on the MegaDepth dataset.

Platform	Time (ms)
PyTorch	16.74
TensorRT	**6.72**

Table 10. Comparison of inference time on different platforms. The running times for an image pair with 640×480 resolution are measured on a single NVIDIA 3090 GPU.

parameters caused by variation N are relatively small, rendering the comparison of efficiency between different settings unnecessary, as they exhibit similar performance in terms of computational cost.

7.4. TensorRT Runtime

To further demonstrate the potential of our method in industrial applications, we deployed the EDM with float32 precision based on ONNX Runtime with TensorRT engine, and compared its runtime with that of the native PyTorch model. The inference times for the same image pair on the identical hardware are presented in Tab. 10. With the acceleration provided by the TensorRT platform, our deployment-friendly model can achieve a higher efficiency. Besides, due to the sensitivity of feature matching tasks to precision, we do not recommend reducing the numerical precision of a well-trained model.

Batch Size	1	2	4	8	16	32
Runtime (ms)	16.7	20.7	37.4	70.1	136.5	270.8

Table 11. Comparison of inference time on different batch size.

Method	Parameters(M)	FLOPs(G)	Memory(MB)	Runtime(ms)	AUC@5°
RoMa [16]	111.3	2586.3	2791.1	312.9	**28.9**
SP [11] + LG [34]	**8.9**	290.5	646.1	29.2	14.8
LoFTR [60]	11.6	783.6	1029.6	71.8	16.9
ELoFTR [66]	15.1	420.9	985.3	39.0	19.2
EDM (ours)	10.2	**72.6**	**493.0**	**16.7**	19.8

Table 12. More Efficiency Comparisons on ScanNet dataset.

Method	Pose Estimation AUC (LO-RANSAC)			Time (ms)
	AUC@5°	AUC@10°	AUC@20°	
LoFTR [60]	62.1	75.5	84.9	134.8
ELoFTR [66]	63.7	77.0	86.4	82.5
JamMa [37]	64.1	77.4	86.5	84.3
Ours	**64.7**	**77.8**	**86.8**	**38.5**

Table 13. Results of Relative Pose Estimation on MegaDepth Dataset following JamMa's setting.

7.5. Batch Inference

Our design prioritizes efficiency and deployment flexibility. Our method enables data to be grouped into mini-batches for batch inference, thereby reducing average computational resource consumption overall. As shown in Tab. 11, inference latency measurements across batch sizes are benchmarked on a single NVIDIA 3090 GPU with 640×480 resolution.

7.6. Other Efficiency Comparisons

More efficiency comparisons in as Tab. 12 shown, the results further validate the effectiveness of our method.

7.7. Additional Results on other RANSAC setting

Recent semi-dense method JamMa [37] introducing Mamba [20] to enhance matching performance and efficiency, employs more advanced poselib LO-RANSAC [28] for evaluating relative pose estimation. We follow the same setting as JamMa to further evaluate our method on the MegaDepth dataset. Specifically, test images are resized and padded to 832×832, and the inlier pixel threshold of LO-RANSAC is set to 0.5. The results are shown in Tab. 13, our method outperforms all previous semi-dense methods in terms of accuracy and efficiency.

8. More Visualizations

8.1. More Intuitive Explanation of Fine Matching

We further explained our bidirectional axis-based matching and regression pipeline. Fig. 9 shows our thinking and im-

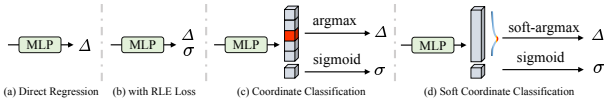

Figure 9. Regression Paradigms.

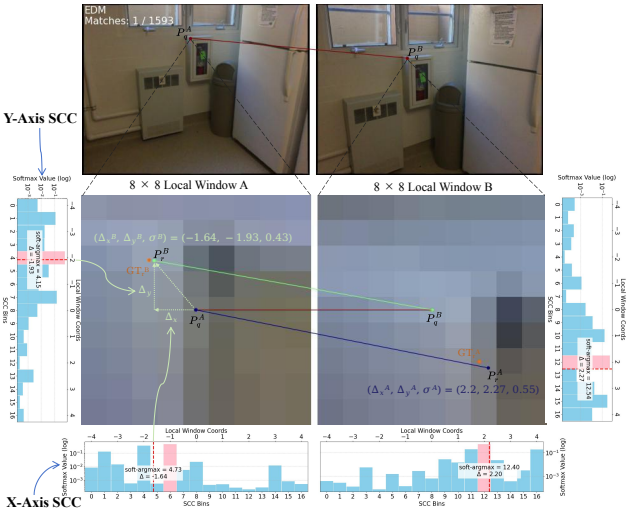

Figure 10. Explanation of Bidirectional Axis-Based Matching.

provement process regarding different regression paradigms in the task of implicit matching coordinate estimation.

Furthermore, as shown in Fig. 10, we illustrate a correctly predicted coarse match from real inference data by visualizing its corresponding 8×8 image patch with both network predictions and ground truth overlaid. This provides a realistic and intuitive explanation to highlight the differences from previous methods. It can be observed that the bidirectional axis-based regression head operates as expected. Specifically, it can be summarized as follows: (a) Distinguishing the implicit encoding of local coordinates for axes reduces the optimization difficulty. (b) Inherent bounding within the window eliminates regression outliers. (c) Multimodal distribution facilitates the correction of imprecise maximum response values. (d) In the two bidirectional fine matching of a pair of coarse correspondences, the one with lower standard deviation σ (higher confidence score) typically has a smaller discrepancy with the ground truth, indicating higher accuracy.

8.2. Failure Cases

As shown in Fig. 11, EDM's failure cases typically occur in scenarios with extreme scale and viewpoint variations or textureless regions.

Figure 11. Failure Cases.

9. Future Work

Although we have made improvements to each step of the detector-free matching pipeline, the efficiency improvement of feature transformation is the least significant. Even though the number of tokens has been significantly reduced, the efficiency issue still persists due to the inherent characteristics of the Transformer. In future work, we consider experimenting and replacing some components in our pipeline, including more efficient backbones and feature transform mechanisms to further improve the accuracy and speed.