RoMa v2: Harder Better Faster Denser Feature Matching

Johan Edstedt[1], David Nordström[2], Yushan Zhang[1], Georg Bökman[3]
Jonathan Astermark[4], Anders Heyden[4], Viktor Larsson[4]
Mårten Wadenbäck[1], Michael Felsberg[1], and Fredrik Kahl[2]

[1]Linköping University [2]Chalmers University of Technology [3]University of Amsterdam [4]Centre for Mathematical Sciences, Lund University

Abstract. Dense feature matching aims to estimate all correspondences between two images of a 3D scene and has recently been established as the gold standard due to its high accuracy and robustness. However, existing dense matchers still fail or perform poorly for many hard real-world scenarios, and high-precision models are often slow, limiting their applicability. In this paper, we attack these weaknesses on a wide front through a series of systematic improvements that together yield a significantly better model. In particular, we construct a novel matching architecture and loss, which, combined with a curated diverse training distribution, enables our model to solve many complex matching tasks. We further make training faster through a decoupled two-stage matching-then-refinement pipeline, and at the same time, significantly reduce refinement memory usage through a custom CUDA kernel. Finally, we leverage the recent DINOv3 foundation model along with multiple other insights to make the model more robust and unbiased. In our extensive set of experiments, we show that the resulting novel matcher sets a new state-of-the-art, being significantly more accurate than its predecessors.[⋆]

Keywords: Dense Feature Matching · Visual Localization · 3D Vision

1 Introduction

Dense matching is the task of matching every pixel of image $\mathbf{I}^A \in \mathbb{R}^{H^A \times W^A \times 3}$ with image $\mathbf{I}^B \in \mathbb{R}^{H^B \times W^B \times 3}$ in terms of a warp $\mathbf{W}^{A \mapsto B} \in \mathbb{R}^{H^A \times W^A \times 2}$ and a confidence mask $\mathbf{p}^{A \mapsto B} \in [0,1]^{H^A \times W^A \times 1}$. In dense *feature* matching, the assumption is that the pixels in both images are observations of 3D points from the *same scene*. In this case, for a perfect matcher, the confidence $\mathbf{p}^{A \mapsto B}$ is 1 for pixels corresponding to a 3D point in the scene that is observable from both views, i.e., that are co-visible, and 0 for occluded pixels.

Feature matching is a fundamental task in Computer Vision, as many downstream tasks, e.g., visual localization [31, 37, 46] and 3D reconstruction [19, 20, 26, 39, 42], rely on precise and trustworthy correspondences in order to function robustly. Traditionally, these methods relied on sparse matches established

⋆ Code is available publicly at https://github.com/Parskatt/RoMaV2.

purely through descriptor similarity. In the last couple of years, these detector-descriptor methods have been gradually replaced by learning-based matchers that consider pairs of images when establishing the correspondences, allowing the networks to not only consider visual similarity but also the spatial context.

(a) Radar chart of performance on benchmarks. Further details on these experiments can be found in Section 4.

(b) Qualitative results. Below each image pair we visualize the dense warp by coloring each pixel by the RGB value from its estimated corresponding location in the opposite image. Brighter values mean lower warp confidence as output by the model.

Fig. 1: Quantitative and Qualitative results. RoMa v2 outperforms previous dense matchers on a wide range of pose estimation and dense matching tasks, as shown in Fig. 1a. We show a snapshot of results across different benchmarks in Fig. 1b.

This development in feature matching has been driven by the introduction of several challenging benchmarks, such as MegaDepth-1500 [22,45], ScanNet-1500 [9,36], WxBS [28] and the recurring Image Matching Challenge at CVPR [15]. These benchmarks are currently dominated by detector-free methods, such as dense feature matchers [11] and feed-forward reconstruction models [51].

One notable dense matcher is RoMa [12], which has proven robust to extreme photometric changes, including different modalities, due to using features from a frozen foundation model for the matching instead of learning features from scratch. However, RoMa still struggles in many challenging scenarios. For example, the recent RUBIK benchmark [25] highlights its weakness under extreme viewpoint changes. Additionally, RoMa has a significant runtime and memory footprint, limiting its applicability for large-scale tasks or resource-constrained settings. Recently, UFM [56] showed that dense matching can be made significantly faster than in RoMa. However, UFM requires finetuning of the pretrained feature extracting backbone, which leads to worse performance on datasets with extreme appearance changes such as WxBS. Furthermore, UFM performs worse than RoMa on benchmarks that require subpixel precision, such as MegaDepth-1500 [22,45].

Motivated by the different trade-offs in RoMa and UFM, in this paper we address the challenge of combining their respective strengths, i.e., developing a dense feature matcher that is both robust to extreme changes in viewpoint and

appearance, applicable to a wide range of real-world scenarios, all while maintaining subpixel precision and a practical runtime and memory footprint. To this end, we introduce RoMa v2, which builds on RoMa and features several improvements to increase robustness while simultaneously reducing the computational cost. RoMa v2 achieves state-of-the-art results on a wide range of benchmarks, as seen in Figure 1a. Qualitative examples from the benchmarks are visualized in Figure 1b.

In summary, our main contributions are:

1. A novel matching objective, combining warp and correlation-based losses, which enables multi-view context to be learned in the coarse matcher of RoMa, described in Section 3.2.
2. Faster and less memory intensive refiners than in RoMa, described in Section 3.3 and ablated in Table 8.
3. A custom mixture of wide- and small-baseline datasets in the training data, that helps balance robustness to extreme viewpoints while maintaining subpixel performance across a wide range of difficult matching tasks, detailed in Section 3.4.
4. Prediction of pixel-wise error covariance that can be used downstream in refinement of estimated geometry, as demonstrated in Section 4.8.
5. We experimentally verify that these improvements significantly reduce the runtime compared to baseline RoMa, while matching or outperforming both RoMa and UFM on their respective strong-points across a wide range of benchmarks in Section 4.

2 Related Work

Feature Matching. Traditionally, feature matching has been dominated by the sparse paradigm, where keypoints are first detected separately in each image, and then matched by a sparse feature matcher. While early approaches relied on similarity between local descriptors, the recent trend is instead to rely on a learned matcher that jointly considers the keypoints and descriptors in each image. Notable recent works in this area include SuperGlue [36], LightGlue [24] and LoMa [29], which all use an attention-based approach for solving the optimal assignment. Common among the sparse methods is the reliance on salient image regions, where repeatable keypoints can be detected. In contrast, LoFTR [45] takes a detector-free approach by matching through attention on learned features, resulting in a semi-dense matching where even non-salient points can be matched. In DKM [11], the matching is performed on a pyramid of feature maps, enabling pixel-dense matches. In a follow-up work, RoMa [12] further improves on this method by using a frozen foundation model to encode the coarse matching features, making it significantly more robust to extreme appearance changes. Recently, UFM [56] was introduced as a more lightweight dense matcher, where the training of wide-baseline dense matching is unified with the related task of optical flow.

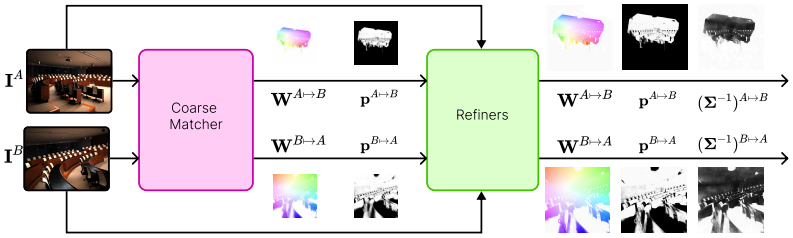


Fig. 2: Overview of RoMa v2. We estimate bidirectional dense image warps $\mathbf{W} = \{\mathbf{W}^{A\mapsto B} \in \mathbb{R}^{H\times W\times 2}, \mathbf{W}^{B\mapsto A} \in \mathbb{R}^{H\times W\times 2}\}$ and warp confidences $\mathbf{p} = \{\mathbf{p}^{A\mapsto B} \in \mathbb{R}^{H\times W\times 1}, \mathbf{p}^{B\mapsto A} \in \mathbb{R}^{H\times W\times 1}\}$ between two input images using a two-stage pipeline consisting of a matching and refinement stage. Unlike recent SotA dense matchers, we additionally predict a precision matrix $\mathbf{\Sigma}^{-1} = \{(\mathbf{\Sigma}^{-1})^{A\mapsto B} \in \mathbb{R}^{H\times W\times 2\times 2}, (\mathbf{\Sigma}^{-1})^{B\mapsto A} \in \mathbb{R}^{H\times W\times 2\times 2}\}$. The **coarse matcher** is a Multi-view Transformer that takes in frozen DINOv3 [41] foundation model features from image $\mathbf{I}^A \in \mathbb{R}^{H\times W\times 3}$ and $\mathbf{I}^B \in \mathbb{R}^{H\times W\times 3}$. Its internals are further illustrated in Figure 3, and explained in detail in Section 3.2. The **refiners** are fine-grained UNet-like CNN models that, conditioned on the previous warp and confidence, produce displacements and delta confidences. Additionally, they predict a full 2×2 precision matrix per pixel, which is visualized as $\left|\mathbf{\Sigma}^{-1}\right|^{-1/4}$. The refiners are further illustrated in Figure 4 and explained in more detail in Section 3.3.

Feed-forward Reconstruction. Recovering 3D structure and camera parameters from images, or Structure-from-Motion (SfM) [14], has traditionally relied on sequential pipelines such as Bundler [42] and COLMAP [39], where point correspondences obtained through image matching play a central role. Recently, learning-based SfM methods have emerged, and many now incorporate matching within a feed-forward architecture. DUSt3R [51] and MASt3R [21] directly regress point maps from image pairs; and VGGT [50] and MapAnything [18] extend this paradigm to longer sequences. While these models can produce coarse correspondences, they struggle to yield accurate high-resolution dense matches. While our architecture is loosely inspired by these approaches, we retain an explicit dense matching formulation that provides subpixel-accurate correspondences.

3 Method

In this section, we outline our proposed method. We begin by discussing our two-stage *matching-then-refinement* architecture in Section 3.1 and proceed to discuss its two parts in Sections 3.2 and 3.3, respectively. In Section 3.4, we describe the training data used and in Section 3.5 we explain the method used to make RoMa v2 more robust to changes in image resolution.

3.1 Architecture

We take inspiration from previous works [12, 56] and divide the dense matching task into a *matching* step and a *refinement* step. Intuitively, these two tasks entail

first finding an approximate or *coarse* match for each pixel, and, conditioned on this, refining the matching to sub-pixel accuracy. An overview of the architecture is shown in Figure 2, and the respective components in Figure 3 and Figure 4.

While some previous works, e.g. [11,12], decouple gradients between matchers and refiners but still train both jointly, we instead opt for a two-stage training paradigm inspired by UFM [56]. This enables rapid experimentation. We next go into detail on the matcher in Section 3.2, followed by the refinement in Section 3.3.

3.2 Matcher

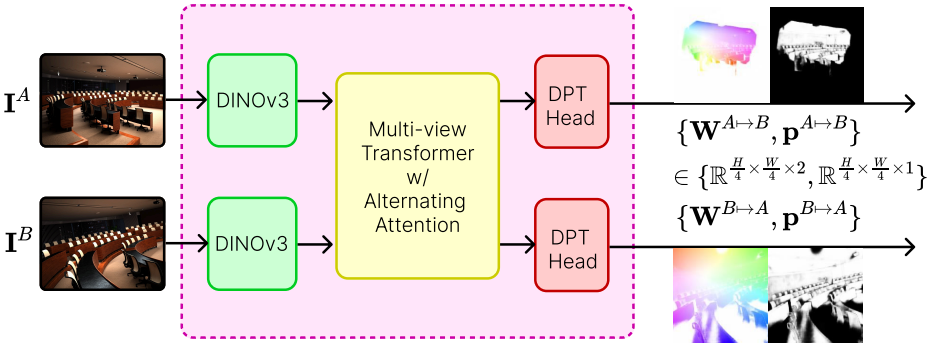


Fig. 3: Coarse matcher. We use a frozen DINOv3 feature extractor in the coarse matching stage. DINOv3 features from both input images are input to a Multi-view Transformer utilizing alternating Attention. Dense Prediction Transformer (DPT) [32] heads output coarse warps $\mathbf{W}$ between the images and confidences $\mathbf{p}$ for 4x downsampled resolution.

An overview of the coarse matcher is shown in Figure 3. We upgrade the DINOv2 [30] encoder used in RoMa to the newer DINOv3 [41]. Inspired by RoMa [12], we compare the encoders (frozen) by training a single linear layer on the features followed by a kernel nearest neighbor matcher. As is shown in Table 1, we find that DINOv3 is more robust than its predecessor despite its slightly larger patch size (16 vs. 14). This is also supported by Table 10. As in RoMa, but unlike UFM, we freeze the encoder weights.

While RoMa is robust and generalizes well, one of its main weaknesses is the lack of multi-view context in the matcher, which relies solely on Gaussian Process (GP) [33] regression combined with a single-view Transformer decoder to classify warp bins. A naive approach would be to add a Multi-view Transformer to RoMa before the GP, however, we found that in practice the gradients through the GP were not sufficiently informative to yield improvements, and caused stability issues during training.

To remedy this, we replace the Gaussian Process with a simple single-headed Attention mechanism. Additionally, we add an auxiliary target, $\mathcal{L}_{\text{NLL}}$, to mini-

mize the negative log-likelihood of the best matching patch in image B for each patch in image A. We first compute the similarity between all patches in image A and image B to form a similarity matrix $\mathcal{S} \in \mathbb{R}^{M \times N}$, where M and N are the number of patches in image A and B respectively. The loss $\mathcal{L}_{\text{NLL}}$ is computed by first applying Softmax over the second dimension of $\mathcal{S}$ and then selecting the most similar pairs for all patches in A, i.e.

$$\mathcal{L}_{\text{NLL}} = \sum_{m=1}^{M} -\log(\text{Softmax}(\mathcal{S}_m)_{n^*}) \tag{1}$$

where n^* is the index of the patch closest to the GT warp for patch m. This approach can be seen as a dense directional version of, e.g. LoFTR [45]. However, we still use a regression head, which is trained to minimize the robust regression loss between its predicted warp and the ground truth warp. The full coarse matching pipeline independently tokenizes the input images using DINOv3 ViT-L, and then applies a ViT-B Multi-view Transformer. Following VGGT [50], we alternate between frame-wise and global Attention. In contrast to VGGT [50], we only use RoPE [44] for the frame-wise Attention. The final token embeddings, $\text{Softmax}(\mathcal{S})x^B$, where x^B are position embeddings as in RoMa, and DINOv3 features, are jointly processed by a Dense Prediction Transformer (DPT) [32] head to predict the warp and confidence. Further details on the matcher architecture are given in the supplementary material.

Matching Loss: We use the same overlap loss $\mathcal{L}_{\text{overlap}}$, and weighting factor ($\lambda = 0.1$), as in RoMa. However, we replace the classification-by-regression term from the matching loss for the robust regression term $\mathcal{L}_{\text{warp}}$ used in the refinement loss. Finally, we add our proposed $\mathcal{L}_{\text{NLL}}$ and obtain:

$$\mathcal{L}_{\text{matcher}} = \mathcal{L}_{\text{NLL}} + \mathcal{L}_{\text{warp}}(\mathbf{r}_{\theta_{\text{matcher}}}, \mathbf{p}_{\text{GT}}) + \lambda \mathcal{L}_{\text{overlap}}(\mathbf{p}_{\theta_{\text{matcher}}}, \mathbf{p}_{\text{GT}}). \tag{2}$$

In contrast to UFM, our matching objective incorporates the auxiliary target $\mathcal{L}_{\text{NLL}}$. We compare these architectures in Table 2 by training on a subset of the data using the training setup outlined below and evaluating on holdout scenes from Hypersim [34]. We find that the RoMa v2 setup works significantly better.

Table 1: Robustness of frozen features. End-point-error (EPE) for a linear probe on MegaDepth. Robustness is the share of matches with error below 32px.

Method	EPE ↓	Robustness % ↑
DINOv2	27.1	77.0
DINOv3	**19.0**	**86.4**

Table 2: Comparing RoMa v2 and UFM matching architectures on Hypersim [34]. Measured in PCK (higher is better).

Method ↓	PCK@ →	1px ↑	3px ↑	5px ↑
UFM		11.2	48.3	67.4
RoMa v2		**30.5**	**76.7**	**86.7**

Coarse Matching Training: We start by training the coarse matcher for 300k steps of batch size 128, resulting in approximately 38M pairs seen throughout training. We use a learning rate of $4 \cdot 10^{-4}$.

3.3 Refiners

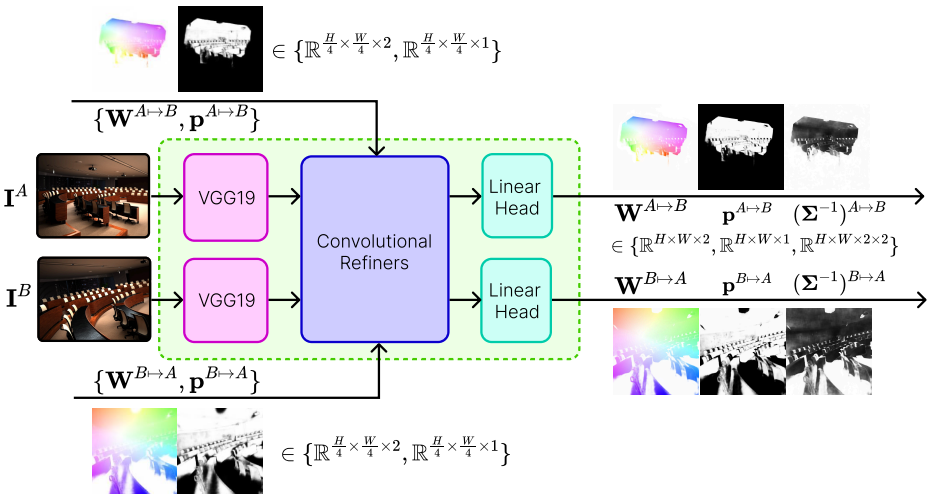


Fig. 4: Refiner internals. The coarse matcher predicts at a resolution 4x smaller than the original image size. The refiners output at the original resolution.

After the matcher has finished training, we freeze it and run it in inference mode, producing a coarse warp for the refiners to refine. We train the refiners for 300k steps of batch size 64, resulting in a total of approximately 19M pairs. Like the matcher, we use a learning rate of $4 \cdot 10^{-4}$.

Architecture: As the matcher predicts at stride 4, in contrast to RoMa's stride 14, we only need to refine at strides ≤ 4. We thus construct three refiners at strides $\{4, 2, 1\}$, respectively. These follow a similar architecture as in RoMa [12] with some efficiency improvements. First, we find that the local correlation implementation used in RoMa uses a large amount of memory, especially at high resolution. To remedy this we write a custom CUDA kernel as a PyTorch extension, which significantly reduces the memory consumption (cf. Table 8). We further change all channel dimensions to be powers of two, which further boosts performance. Further details about the refiners are given in the appendix.

Predictive Covariance: In addition to predicted overlap, it is often useful to have access to a numerical estimate of the expected error. While predictive uncertainty has been previously studied [16, 38, 40, 48, 55, 59], state-of-the-art matchers such as RoMa [12] or UFM [56] do not provide any such estimate. To

remedy this, we predict a pixel-wise Gaussian uncertainty of the 2D residuals,

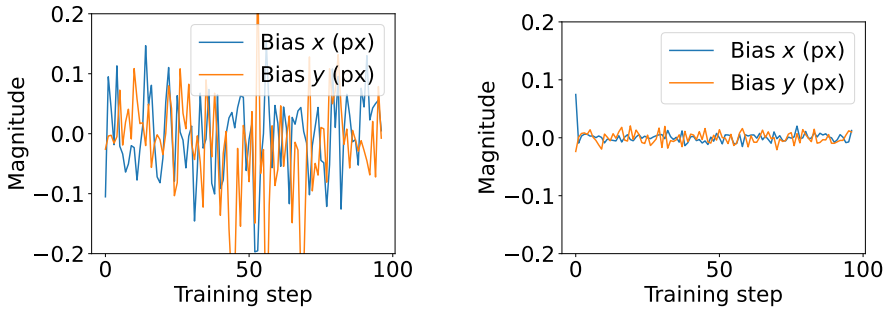
$$\mathbf{r}_\theta := \mathbf{W}_\theta^{A\mapsto B} - \mathbf{W}_{\text{GT}}^{A\mapsto B} \in \mathbb{R}^{H\times W\times 2}, \tag{3}$$

through a 2×2 precision matrix $\mathbf{P}_\theta \in \mathbb{R}^{H\times W\times 2\times 2}$ where $\mathbf{\Sigma}_\theta^{-1}(h,w) \succ 0$, i.e., $\mathbf{\Sigma}_\theta^{-1}(h,w)$ is positive definite. We ensure this by constraining the network to predict the three elements z_{11}, z_{21}, z_{22} and mapping these to Cholesky factors as $l_{11} = \text{Softplus}(z_{11}) + 10^{-6}$, $l_{21} = z_{21}$, $l_{22} = \text{Softplus}(z_{22}) + 10^{-6}$, where $\text{Softplus}(\cdot) = \ln(1+\exp(\cdot))$. The lower triangular matrix is composed from the factors as $L = \begin{pmatrix} l_{11} & 0 \\ l_{21} & l_{22} \end{pmatrix}$ and then the covariance matrix is formed from this as $\Sigma^{-1} = LL^\top$. To learn z_{11}, z_{21}, z_{22} we directly train the model to minimize the negative log-likelihood

$$\mathcal{L}_{\text{prec}}(\mathbf{r}) = -\log \mathcal{N}(\mathbf{r}|0,\Sigma) = \frac{1}{2}\mathbf{r}^\top \Sigma^{-1}\mathbf{r} - \frac{1}{2}\log\det(\Sigma^{-1}) + \log(2\pi) \tag{4}$$

To ensure stability, we only train the model to predict this covariance for covisible regions where $\|\mathbf{r}\| < 8$ pixels. We detach the residuals $\mathbf{r}_\theta$ before the loss.

We predict the precision in a hierarchical fashion from stride 4 up to stride 1, and use the fact that information is additive in the precision parameterization to predict our final precision matrix as $\mathbf{\Sigma}_{\theta_i}^{-1} = \sum_{j\geq i} \Delta\mathbf{\Sigma}_{\theta_j}^{-1}$. We find empirically that our covariance improves performance in downstream tasks (cf. Table 12), and that it qualitatively behaves as one would expect in Figure 7.



(a) Before EMA **(b)** After EMA

Fig. 5: Subpixel bias of refinement. We observe that models exhibit subpixel fluctuations in their predictions throughout training, leading to bias. We propose a simple remedy through storing an exponential moving average (EMA).

EMA to remedy bias: During training, we empirically observed that predictions tend to have a small, but noticeable, sub-pixel bias (typically around ± 0.1 pixels in resolution 640×640). At first, this seemed like a data issue, but after plotting this bias over the course of training we found that it appears almost

random, see Figure 5a. As the bias is seemingly uncorrelated over the course of training, a simple way to fix it is to use an Exponential Moving Average (EMA)[**]. We found a decay factor of $\alpha = 0.999$ to work well empirically. After applying this remedy, we find that the bias in both orientations is substantially diminished, see Figure 5b. We also find improved accuracy, see Table 9.

Refinement Loss: We train the refiners using a combination of three losses. Following RoMa we use a generalized Charbonnier loss [2] which for each refiner reads $\mathcal{L}_{\text{warp}} = (ic)^\alpha \left(\frac{\|\mathbf{r}\|^2}{(ic)^2} + 1^2 \right)^{\alpha/2}$, where we follow RoMa and set $\alpha = 0.5$, $c = 10^{-3}$ and $i \in \{4, 2, 1\}$ is the stride.

For estimating the overlap we follow UFM and RoMa and use a pixel-wise binary cross-entropy loss as $\mathcal{L}_{\text{ov}}$, with the ground truth overlap $\mathbf{p}_{\text{GT}} \in \{0, 1\}$ being derived from either consistent depth (for MVS style datasets) or from warp cycle consistency (for flow datasets). Further details on computing ground truth warps and overlaps are given in the suppl. material. The total refinement loss is

$$\mathcal{L}_{\text{refiners}} = \sum_{i \in \mathcal{S}} \mathcal{L}_{\text{warp}}(\mathbf{r}_{\theta_i}, \mathbf{p}_{\text{GT}}) + \lambda_{\text{ov}} \mathcal{L}_{\text{ov}}(\mathbf{p}_{\theta_i}, \mathbf{p}_{\text{GT}}) + \lambda_{\text{prec}} \mathcal{L}_{\text{prec}}(\mathbf{\Sigma}_{\theta_i}^{-1}, \text{detach}(\mathbf{r}_{\theta_i})), \tag{5}$$

where $\mathcal{S} = \{1, 2, 4\}$, $\lambda_{\text{ov}} = 10^{-2}$, $\lambda_{\text{prec}} = 10^{-3}$.

3.4 Data

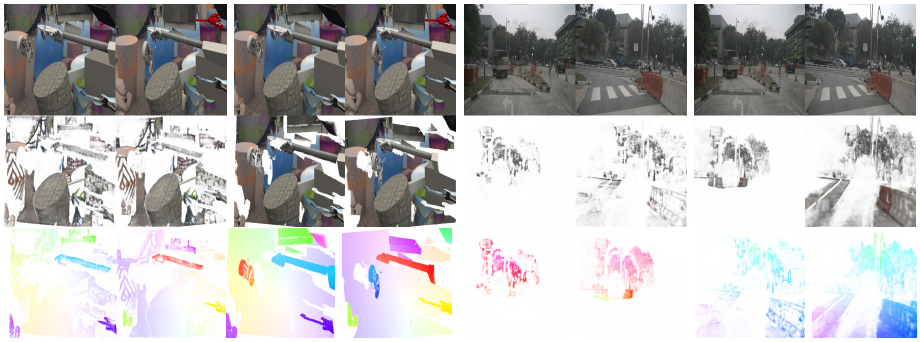
We train RoMa v2 on a mix of wide and small baseline two-view datasets, a summary of which is presented in Table 3. Our mix is inspired by UFM [56], and significantly more diverse than RoMa [12], which is only trained on MegaDepth.

In particular, the inclusion of the Aerial datasets AerialMD [49] and BlendedMVS [53], enable our proposed model to be significantly more robust to large rotations and air-to-ground viewpoint changes. The inclusion of small-baseline datasets, like FlyingThings3D [27], makes RoMa v2 significantly better at predicting fine-grained details. We qualitatively compare RoMa v2 to RoMa on fine-grained prediction on the FlyingThings3D dataset in Figure 6a. We also find that our data mixture enables us to predict textureless surfaces significantly better than RoMa, particularly in Autonomous Driving (AD) scenarios, despite only training on the very small-scale dataset VKITTI2 [6, 13]. We demonstrate this for a randomly selected pair from the NuScenes dataset [7] in Figure 6b.

3.5 Resolution

We find that some elementary key changes make matching and refinement robust to the choice of resolution.

[**] See [17] for discussion regarding different variants.



(a) Left: RoMa warp for small baseline pair. Note the missing warp for the guitar in the bottom right. **Right:** RoMa v2 warp.

(b) Left: RoMa warp for large-baseline pair with texture-poor road surfaces, with warp missing for almost the entire road. **Right:** RoMa v2 warp.

Fig. 6: Qualitative Comparison. RoMa v2 is significantly better than RoMa at capturing small objects with dynamic motion **(a)**, and for texture-less scenes **(b)**

Table 3: Dataset mixture for RoMa v2. The top part contains wide-baseline datasets, while the bottom part contains small-baseline datasets. The weight is proportional to the probability of sampling an image pair from the respective dataset. Further details about the datasets are provided in the supplementary.

Datasets	Type / GT Source	Weight	№Scenes
MegaDepth [22]	Outdoor / MVS	1	169
AerialMD [49]	Aerial / MVS	1	124
BlendedMVS [53]	Aerial / Mesh	1	493
Hypersim [34]	Indoor / Graphics	1	393
TartanAir v2 [52]	Outdoor / Graphics	1	46
Map-Free [1]	Object-centric / MVS	1	397
ScanNet++ v2 [54]	Indoor / Mesh	1	856
UnrealStereo4k [47]	Outdoor / Graphics	0.01	8
Virtual KITTI 2 [6, 13]	Outdoor / Graphics	0.01	5
FlyingThings3D [27]	Outdoor / Graphics	0.5	2239
Total			**5069**

Coarse Matching: Following DINOv3 [41] we use RoPE [44] on a *normalized* grid, rather than a pixel grid. This ensures that distances are always in distribution, even when changing resolution significantly.

Secondly, we find that the absolute position encodings used for the match embeddings need to be of low enough frequency to ensure that their interpolation is unproblematic [57]. Compared to RoMa, which initializes the scale to $\omega = 8$ and lets it be trained, we set it fixed to $\omega = 1$. It is possible that the high frequency of the position embeddings is the cause of the issue that requires UFM to be run at a fixed resolution of 420×560 during inference.

Refinement: Ensuring resolution robustness for the refiners is non-trivial, as convolution is tied to the pixel grid. We find that the approach used in RoMa, whereby the input displacement is rescaled relative to a canonical resolution generalizes best.

Training: We train the coarse matcher on a mix of resolutions and aspect ratios, specifically: $\{512 \times 512, 592 \times 448, 624 \times 416, 688 \times 384, 448 \times 592, 416 \times 624, 384 \times 688\}$. The refiners are trained exclusively with size 640×640.

4 Experiments

The qualitative improvements of RoMa v2 as shown in Figures 1b, 6a and 6b are confirmed by the quantitative results from extensive experiments, listed in this section.

4.1 Relative Pose Estimation

We compare RoMa v2 to state-of-the-art matchers and feed-forward 3D reconstruction methods on relative pose estimation. For sampling correspondences we follow RoMa and compute bidirectional warps from which we sample correspondences from a thresholded distribution where we set

$$\hat{\mathbf{p}}^{A\mapsto B} = \max(\mathbb{1}_{\mathbf{p}^{A\mapsto B}>0.05}, \mathbf{p}^{A\mapsto B}). \tag{6}$$

We use a coarse resolution of 800×800 and a fine resolution of 1024×1024. Similarly, we sample a balanced subset of matches using their kernel density estimate approach.

We report results on MegaDepth-1500 [22,45] and ScanNet-1500 [9,36] in Table 4. RoMa v2 consistently outperforms all prior matchers on both benchmarks. On MegaDepth, which demands accurate sub-pixel correspondences, RoMa v2 also surpasses all 3D reconstruction methods. On ScanNet, RoMa v2 achieves performance that is on par with VGGT and MASt3R.

4.2 Visual Localization

We evaluate visual localization on Map-free [1] and InLoc [46] in Table 5. For Map-free, we use the validation set and metric monocular depth from DA3 [23]. We use the HLoc [35] pipeline to evaluate on InLoc [46]. RoMa v2 significantly outperforms RoMa on both benchmarks. In fact, RoMa v2 tops the public leaderboard for InLoc setting a new SotA.

4.3 Dense Matching

We evaluate dense matching performance in Table 6 and Table 7 on a wide array of datasets and compare with state-of-the-art dense matchers RoMa and UFM. For RoMa and RoMa v2, we directly feed the 640×640 images into the model, while for UFM we first resize the image to their suggested inference resolution 560×420 and then bilinearly upsample the predictions back to 640×640, as their precision degrades significantly for higher resolutions.

RoMa v2 consistently outperforms previous methods across all 6 datasets. Notably, we simultaneously beat UFM on its own benchmark, TA-WB, and RoMa on MegaDepth, on which it is exclusively trained, while having 84% lower EPE compared to RoMa on the challenging AerialMegaDepth benchmark.

Table 4: SotA comparison on MegaDepth-1500 [22,45] and ScanNet-1500 [9, 36]. The top part contains feed-forward 3D reconstruction models, while the bottom part contains feature matchers.

Method	MegaDepth			ScanNet		
AUC@ →	5°	10°	20°	5°	10°	20°
MASt3R [21] ECCV'24	42.4	61.5	76.9	33.6	56.8	74.1
VGGT† [50] CVPR'25	33.5	52.9	70.0	33.9	55.2	73.4
Reloc3r [10] CVPR'25	**49.6**	**67.9**	**81.2**	**34.8**	**58.4**	**75.6**
LoFTR [45] CVPR'21	52.8	69.2	81.2	22.1	40.8	57.6
DKM [11] CVPR'23	60.4	74.9	85.1	29.4	50.7	68.3
LightGlue [24] ICCV'23	51.0	68.1	80.7	17.8	34.0	52.0
RoMa [12] CVPR'24	62.6	76.7	86.3	31.8	53.4	70.9
UFM† [56] NeurIPS'25	41.5	57.9	72.4	31.3	54.1	72.0
RoMa v2	**62.8**	**77.0**	**86.6**	**33.6**	**56.2**	**73.8**

†Our reproduced numbers.

Table 5: Visual Localization. Comparison to RoMa on Map-free [1] and InLoc [46].

Method	Map-free		InLoc	
	Prec.	AUC	DUC1	DUC2
RoMa	59.7	84.4	60.6/79.3/89.9	66.4/83.2/87.8
RoMa v2	**73.3**	**89.6**	**67.7/84.3/94.4**	**73.3/86.3/90.8**

4.4 Ablations and Runtime Comparisons

In Table 8 we compare the runtime of RoMa v2 with RoMa and UFM. As can be seen from the table, we improve the throughput significantly compared to RoMa, running 1.7× faster. Compared to UFM our model is slightly slower, however with a much smaller memory footprint. In Tab. 9, we ablate the refiner EMA using the same metrics as in Tab. 4; in Tab. 10 we ablate the backbone.

4.5 Multi-Modal Matching on WxBS

We evaluate the robustness of RoMa v2 on the extremely challenging WxBS benchmark [28]. This benchmark consists of hand-labeled correspondences be-

Table 6: Dense matching performance. Images are resized to 640 × 640.

Method	TA-WB				MegaDepth				ScanNet++ v2			
	EPE ↓	1px ↑	3px ↑	5px ↑	EPE ↓	1px ↑	3px ↑	5px ↑	EPE ↓	1px ↑	3px ↑	5px ↑
RoMa	60.61	35.1	52.6	56.2	2.34	74.8	93.7	96.4	27.52	20.2	42.8	53.6
UFM	15.85	31.3	65.5	75.1	3.15	55.3	88.0	93.7	6.93	31.4	67.7	80.0
RoMa v2	**13.82**	**67.7**	**81.8**	**85.8**	**1.47**	**79.6**	**94.7**	**96.7**	**4.00**	**45.5**	**77.3**	**86.6**

Table 7: Further dense matching performance. Images are resized to 640×640.

Method	FlyingThings3D				AerialMegaDepth				MapFree			
	EPE ↓	1px ↑	3px ↑	5px ↑	EPE ↓	1px ↑	3px ↑	5px ↑	EPE ↓	1px ↑	3px ↑	5px ↑
RoMa	5.68	78.0	86.6	89.2	25.05	39.0	65.0	73.9	8.55	45.8	72.3	80.9
UFM	1.33	83.4	93.9	96.1	17.44	29.3	61.6	73.8	3.59	31.6	66.7	81.7
RoMa v2	**0.93**	**89.4**	**95.2**	**96.8**	**4.12**	**55.9**	**81.1**	**87.9**	**2.03**	**55.4**	**84.9**	**92.7**

Table 8: Runtime and memory. Benchmarking on $640 \times 640^{\dagger}$ images with a batch size of 8 on an H200. RoMa v2 is 1.7× faster than RoMa with similar memory footprint. $\mathcal{K}$ indicates the custom CUDA kernel for the local correlation operation.

Method	Throughput (pairs/s) ↑	Mem. (GB) ↓
UFM	43.0	16.2
RoMa	18.5	4.7
RoMa v2 (w/o $\mathcal{K}$)	30.3	5.6
RoMa v2 (w/ $\mathcal{K}$)	30.9	4.8

†We use 644×644 for RoMa and UFM due to patch size 14.

tween images taken with extreme changes in either viewpoint, illumination, modality, or all three, which measures the generalizability of the matcher to out-of-distribution downstream tasks. Results are presented in Table 11. We observe that the performance of RoMa v2 is significantly higher than UFM, but slightly lower than RoMa. Investigating the cause of this we found that RoMa v2 and UFM both struggle with the IR-to-RGB multi-modal subset of WxBS.

4.6 Astronaut to Satellite Image Matching

We introduce a new benchmark, SatAst, for matching satellite images to images taken by astronauts from the International Space Station. Prior work on this modality has focused on the retrieval task of searching a database of satellite images for the image content of a given astronaut image [3–5, 43]. We take 39 corresponding image pairs from EarthMatch [3] to create SatAst and hand-annotate 10 accurate correspondences for each pair. Further, we include 90 degree rotated copies of the satellite images, yielding a total of 156 image pairs. Given estimated correspondences from a model, we use RANSAC to obtain a homography and compute AUC@10px of the reprojection error of the annotated ground-truth correspondences using this homography.

SatAst is difficult, the most challenging aspects being i) satellite images are OOD for most matchers (including RoMa v2), ii) large scale changes and iii) large in-plane rotations. We compare RoMa v2 against previous dense methods and present results in Table 11. More information about the benchmark is found in the supplementary material.

Table 9: EMA ablation. Refiner EMA improves results specifically on benchmarks that require subpixel precision, such as Mega-1500.

	Mega-1500 ↑	SN-1500 ↑
w/o EMA	(61.4, 75.8, 85.7)	(**33.6**, 56.1, 73.5)
EMA	(**62.8**, **77.0**, **86.6**)	(**33.6**, **56.2**, **73.8**)

Table 10: Backbone ablation. We compare using DINOv2 vs DINOv3 in the coarse matcher, skipping the refiners.

Backbone	WxBS	Hypersim
DINOv2	**35.6**	78.1
DINOv3	34.2	**79.2**

Table 11: SotA comparison on the WxBS [28], SatAst, and RUBIK [25] benchmarks. mAA, AUC at 10px, and success ratio, respectively (higher is better).

Method	WxBS (mAA@10px)	SatAst (AUC@10px)	RUBIK (Success %)
RoMa	**60.8**	23.5	47.3
UFM	42.3	1.8	53.1
RoMa v2	55.4	**37.0**	**57.3**

4.7 Matching across Geometric Challenges on RUBIK

We evaluate RoMa v2 on RUBIK [25], a subset of NuScenes [7], and report the success ratio (rotation error less than 5° and translation error less than 2m) in Tab. 11. We set a new SotA of 57.3, beating the previous leader DUSt3R (54.8).

4.8 Covariance Estimate

While most robust pose estimation pipelines assume identically distributed residuals, our predictive covariance can be used to reweight residuals. To highlight the usefulness, we perform an experiment leveraging the covariance-weighted residuals. First, only as post-processing, refining the output of a classic point-based RANSAC. Second, we compare with using it to reweight the residuals used for scoring inside RANSAC. For the experiment, we consider image pairs sampled from the HyperSim [34] dataset. In Table 12 we show that we gain significant improvements on 3D pose error metrics. In particular, we improve by ≈ 20 points on AUC@1. Further experimental details can be found in the supplementary material. In Figure 7 we show a qualitative example of the predicted covariances. In the right image, we have applied a linear kernel to simulate motion blur, yielding larger covariances, especially in the blur direction.

4.9 Importance of Subpixel Accuracy for MegaDepth-1500

In Section 4.1 we claimed that subpixel accuracy is critical for high benchmark results on MegaDepth-1500. To give evidence for this, we compute epipolar error histograms for RoMa v2 and UFM. Using these, we get a deterministic quantile mapping as

$$\tilde{r}_{\text{RoMaV2} \to \text{UFM}} = Q_{\text{UFM}}(F_{\text{RoMa v2}}(r_{\text{RoMa v2}})) \tag{7}$$

where Q is the residual quantile function of UFM and F is the CDF of RoMa v2. We then simulate a less precise RoMa v2 by perturbing its predictions using the mapping. Results are shown in Table 13.

Table 12: Impact of predictive covariance on Hypersim [34]. Measured in AUC (higher is better). We use the predicted covariance to weight the residuals in the model refinement.

Method ↓ AUC@ →	$1^\circ \uparrow$	$3^\circ \uparrow$	$5^\circ \uparrow$
RoMa v2 (w/o Σ^{-1})	54.9	79.5	85.9
RoMa v2 (w/ Σ^{-1} Refine)	75.8	89.0	92.6
RoMa v2 (w/ Σ^{-1} RANSAC + Refine)	**76.4**	**89.3**	**92.8**

Fig. 7: Qualitative example of covariances. Predicted covariance for randomly sampled keypoints with simulated motion blur in the right image.

Table 13: Impact of subpixel accuracy.

Method	MegaDepth-1500		
UFM	41.5	57.9	72.4
RoMa v2	**62.8**	**77.0**	**86.6**
RoMa v2 (Perturb.)	46.3	63.7	77.4

5 Limitations and Future Work

Compared to RoMa, our model is slightly less robust to extreme changes in modality, such as in WxBS. However, we are significantly more robust to these changes than UFM, as indicated by Table 11. Exploring the trade-offs between generalization and performance is an interesting direction for future work.

6 Conclusion

We have introduced RoMa v2, a new dense feature matcher capable of matching *harder* pairs, with *better* (i.e., more precise) predictions, with *faster* runtime than its predecessor, leading to *denser* matches (i.e., *more* correct matches).

Acknowledgements

This work was supported by the Wallenberg Artificial Intelligence, Autonomous Systems and Software Program (WASP), funded by the Knut and Alice Wallenberg Foundation, and by the strategic research environment ELLIIT, funded by the Swedish government. The computational resources were provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS) at C3SE, partially funded by the Swedish Research Council through grant agreement no. 2022-06725, and by the Berzelius resource, provided by the Knut and Alice Wallenberg Foundation at the National Supercomputer Centre.

References

1. Arnold, E., Wynn, J., Vicente, S., Garcia-Hernando, G., Monszpart, A., Prisacariu, V., Turmukhambetov, D., Brachmann, E.: Map-free visual relocalization: Metric pose relative to a single image. In: European Conf. Computer Vision (ECCV) (2022)
2. Barron, J.T.: A general and adaptive robust loss function. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2019)
3. Berton, G., Goletto, G., Trivigno, G., Stoken, A., Caputo, B., Masone, C.: Earthmatch: Iterative coregistration for fine-grained localization of astronaut photography. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2024)
4. Berton, G., Stoken, A., Caputo, B., Masone, C.: Earthloc: Astronaut photography localization by indexing earth from space. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2024)
5. Bökman, G., Edstedt, J., Felsberg, M., Kahl, F.: Steerers: A framework for rotation equivariant keypoint descriptors. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2024)
6. Cabon, Y., Murray, N., Humenberger, M.: Virtual kitti 2. arXiv preprint arXiv:2001.10773 (2020)
7. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuscenes: A multimodal dataset for autonomous driving. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2020)
8. Chojnacki, W., Brooks, M.J., Van Den Hengel, A., Gawley, D.: On the fitting of surfaces to data with covariances. IEEE Trans. Pattern Analysis and Machine Intelligence (T-PAMI) **22**(11), 1294–1303 (2000)
9. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2017)
10. Dong, S., Wang, S., Liu, S., Cai, L., Fan, Q., Kannala, J., Yang, Y.: Reloc3r: Large-scale training of relative camera pose regression for generalizable, fast, and accurate visual localization. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2025)
11. Edstedt, J., Athanasiadis, I., Wadenbäck, M., Felsberg, M.: DKM: Dense kernelized feature matching for geometry estimation. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2023)
12. Edstedt, J., Sun, Q., Bökman, G., Wadenbäck, M., Felsberg, M.: RoMa: Robust dense feature matching. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2024)
13. Gaidon, A., Wang, Q., Cabon, Y., Vig, E.: Virtual worlds as proxy for multi-object tracking analysis. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2016)
14. Hartley, R., Zisserman, A.: Multiple view geometry in computer vision. Cambridge university press (2003)
15. Howard, A., Trulls, E., Yi, K.M., Mishkin, D., Dane, S., Jin, Y.: Image matching challenge 2022 (2022), https://kaggle.com/competitions/image-matching-challenge-2022
16. Ilg, E., Cicek, O., Galesso, S., Klein, A., Makansi, O., Hutter, F., Brox, T.: Uncertainty estimates and multi-hypotheses networks for optical flow. In: European Conf. Computer Vision (ECCV) (2018)

17. Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D., Wilson, A.G.: Averaging weights leads to wider optima and better generalization. arXiv preprint arXiv:1803.05407 (2018)
18. Keetha, N., Müller, N., Schönberger, J., Porzi, L., Zhang, Y., Fischer, T., Knapitsch, A., Zauss, D., Weber, E., Antunes, N., Luiten, J., Lopez-Antequera, M., Bulò, S.R., Richardt, C., Ramanan, D., Scherer, S., Kontschieder, P.: MapAnything: Universal feed-forward metric 3D reconstruction (2025), arXiv preprint arXiv:2509.13414
19. Kotovenko, D., Grebenkova, O., Ommer, B.: Edgs: Eliminating densification for efficient convergence of 3dgs. arXiv preprint arXiv:2504.13204 (2025)
20. Lee, J., Yoo, S.: Dense-sfm: Structure from motion with dense consistent matching. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2025)
21. Leroy, V., Cabon, Y., Revaud, J.: Grounding image matching in 3d with mast3r. In: European Conf. Computer Vision (ECCV) (2024)
22. Li, Z., Snavely, N.: Megadepth: Learning single-view depth prediction from internet photos. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2018)
23. Lin, H., Chen, S., Liew, J.H., Chen, D.Y., Li, Z., Shi, G., Feng, J., Kang, B.: Depth anything 3: Recovering the visual space from any views. arXiv preprint arXiv:2511.10647 (2025)
24. Lindenberger, P., Sarlin, P.E., Pollefeys, M.: LightGlue: Local Feature Matching at Light Speed. In: IEEE Int'l Conf. Computer Vision (ICCV) (2023)
25. Loiseau, T., Bourmaud, G.: Rubik: A structured benchmark for image matching across geometric challenges. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2025)
26. Mapillary: Opensfm. https://github.com/mapillary/OpenSfM
27. Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T.: A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2016)
28. Mishkin, D., Matas, J., Perdoch, M., Lenc, K.: WxBS: Wide baseline stereo generalizations. In: British Machine Vision Conference (BMVC) (2015)
29. Nordström, D., Edstedt, J., Bökman, G., Astermark, J., Heyden, A., Larsson, V., Wadenbäck, M., Felsberg, M., Kahl, F.: LoMa: Local feature matching revisited. In: European Conf. Computer Vision (ECCV) (2026)
30. Oquab, M., Darcet, T., Moutakanni, T., Vo, H.V., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., Howes, R., Huang, P.Y., Xu, H., Sharma, V., Li, S.W., Galuba, W., Rabbat, M., Assran, M., Ballas, N., Synnaeve, G., Misra, I., Jegou, H., Mairal, J., Labatut, P., Joulin, A., Bojanowski, P.: DINOv2: Learning robust visual features without supervision. Transactions on Machine Learning Research (TLMR) (2023)
31. Panek, V., Zhou, Q., Ding, Y., Agostinho, S., Kukelova, Z., Sattler, T., Leal-Taixé, L.: A guide to structureless visual localization. arXiv preprint arXiv:2504.17636 (2025)
32. Ranftl, R., Bochkovskiy, A., Koltun, V.: Vision transformers for dense prediction. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2021)
33. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning). The MIT Press (2005)
34. Roberts, M., Ramapuram, J., Ranjan, A., Kumar, A., Bautista, M.A., Paczan, N., Webb, R., Susskind, J.M.: Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. In: IEEE Int'l Conf. Computer Vision (ICCV) (2021)

35. Sarlin, P.E., Cadena, C., Siegwart, R., Dymczyk, M.: From coarse to fine: Robust hierarchical localization at large scale. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2019)
36. Sarlin, P.E., DeTone, D., Malisiewicz, T., Rabinovich, A.: Superglue: Learning feature matching with graph neural networks. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2020)
37. Sattler, T., Maddern, W., Toft, C., Torii, A., Hammarstrand, L., Stenborg, E., Safari, D., Okutomi, M., Pollefeys, M., Sivic, J., Kahl, F., Pajdla, T.: Benchmarking 6dof outdoor visual localization in changing conditions. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2018)
38. Schröppel, P., Bechtold, J., Amiranashvili, A., Brox, T.: A benchmark and a baseline for robust multi-view depth estimation. In: Int'l Conf. 3D Vision (3DV) (2022)
39. Schönberger, J.L., Frahm, J.M.: Structure-from-Motion Revisited. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2016)
40. Shenoi, A., Lindenberger, P., Sarlin, P.E., Pollefeys, M.: Raco: Ranking and covariance for practical learned keypoints. In: Int'l Conf. 3D Vision (3DV) (2026)
41. Siméoni, O., Vo, H.V., Seitzer, M., Baldassarre, F., Oquab, M., Jose, C., Khalidov, V., Szafraniec, M., Yi, S., Ramamonjisoa, M., Massa, F., Haziza, D., Wehrstedt, L., Wang, J., Darcet, T., Moutakanni, T., Sentana, L., Roberts, C., Vedaldi, A., Tolan, J., Brandt, J., Couprie, C., Mairal, J., Jégou, H., Labatut, P., Bojanowski, P.: Dinov3. arXiv preprint arXiv:2508.10104 (2025)
42. Snavely, N., Seitz, S.M., Szeliski, R.: Modeling the world from internet photo collections. Int'l J. Computer Vision (IJCV) **80**(2) (2008)
43. Stoken, A., Fisher, K.: Find my astronaut photo: Automated localization and georectification of astronaut photography. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) Workshops (2023)
44. Su, J., Lu, Y., Pan, S., Murtadha, A., Wen, B., Liu, Y.: Roformer: Enhanced transformer with rotary position embedding (2023), https://arxiv.org/abs/2104.09864
45. Sun, J., Shen, Z., Wang, Y., Bao, H., Zhou, X.: LoFTR: Detector-free local feature matching with transformers. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2021)
46. Taira, H., Okutomi, M., Sattler, T., Cimpoi, M., Pollefeys, M., Sivic, J., Pajdla, T., Torii, A.: Inloc: Indoor visual localization with dense matching and view synthesis. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2018)
47. Tosi, F., Liao, Y., Schmitt, C., Geiger, A.: Smd-nets: Stereo mixture density networks. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2021)
48. Truong, P., Danelljan, M., Timofte, R., Van Gool, L.: PDC-Net+: Enhanced Probabilistic Dense Correspondence Network. IEEE Trans. Pattern Analysis and Machine Intelligence (T-PAMI) (2023)
49. Vuong, K., Ghosh, A., Ramanan, D., Narasimhan, S., Tulsiani, S.: Aerialmegadepth: Learning aerial-ground reconstruction and view synthesis. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2025)
50. Wang, J., Chen, M., Karaev, N., Vedaldi, A., Rupprecht, C., Novotny, D.: Vggt: Visual geometry grounded transformer. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2025)
51. Wang, S., Leroy, V., Cabon, Y., Chidlovskii, B., Revaud, J.: Dust3r: Geometric 3d vision made easy. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2024)

52. Wang, W., Zhu, D., Wang, X., Hu, Y., Qiu, Y., Wang, C., Hu, Y., Kapoor, A., Scherer, S.: Tartanair: A dataset to push the limits of visual slam. In: IEEE/RSJ Int'l Conf. Intelligent Robots and Systems (IROS) (2020)
53. Yao, Y., Luo, Z., Li, S., Zhang, J., Ren, Y., Zhou, L., Fang, T., Quan, L.: Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2020)
54. Yeshwanth, C., Liu, Y.C., Nießner, M., Dai, A.: Scannet++: A high-fidelity dataset of 3d indoor scenes. In: IEEE Int'l Conf. Computer Vision (ICCV) (2023)
55. Yin, Z., Darrell, T., Yu, F.: Hierarchical discrete distribution decomposition for match density estimation. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2019)
56. Zhang, Y., Keetha, N., Lyu, C., Jhamb, B., Chen, Y., Qiu, Y., Karhade, J., Jha, S., Hu, Y., Ramanan, D., Scherer, S., Wang, W.: Ufm: A simple path towards unified dense correspondence with flow. In: Advances in Neural Information Processing Systems (NeurIPS) (2025)
57. Zhao, J., Chen, X., Yuan, Y., Felsberg, M., Wang, D., Lu, H.: Efficient motion prompt learning for robust visual tracking. In: Int'l Conf. Machine learning (ICML) (2025)
58. Zhao, X., Wu, X., Chen, W., Chen, P.C.Y., Xu, Q., Li, Z.: Aliked: A lighter keypoint and descriptor extraction network via deformable transformation. IEEE Transactions on Instrumentation & Measurement **72** (2023)
59. Zhou, H., Ummenhofer, B., Brox, T.: Deeptam: Deep tracking and mapping. In: European Conf. Computer Vision (ECCV) (2018)