Pixel-Perfect Structure-from-Motion with Featuremetric Refinement

Philipp Lindenberger[1*] Paul-Edouard Sarlin[2*] Viktor Larsson[2] Marc Pollefeys[2,3]

Departments of [1]Mathematics and [2]Computer Science, ETH Zurich [3]Microsoft

Supplemental

In the following pages, we present additional details on the experiments conducted in the main paper. We provide in Section A additional detailed results for the tasks of triangulation and camera pose estimation on the ETH3D dataset. In Section B, we analyze how several parameters can impact the accuracy and the computational requirements of the refinement. Section C describes our efficient cost approximation. Finally, in Section D we provide additional implementation details on the experiments presented in the main paper.

A. Additional results on ETH3D

A.1. Triangulation

We refine the triangulation of SuperPoint [2] keypoints for the ETH3D *Courtyard* scene and show in Figure 1 the distribution of triangulation errors for points observed by different numbers of images (track length). Our featuremetric refinement provides the largest improvement for points with low track length, for which the estimates of the traditional geometric BA are dominated by the noise of the keypoint detection. For larger track lengths, the refined point cloud has an accuracy close to the Faro Focus X 330 laser scanner from which the ground truth is computed.

We show in Figure 6 the raw and refined point clouds for SuperPoint and D2-Net. The benefits of our refinement are easily visible in 3D. Planar walls exhibit fewer noisy keypoints and the refined point clouds are more complete.

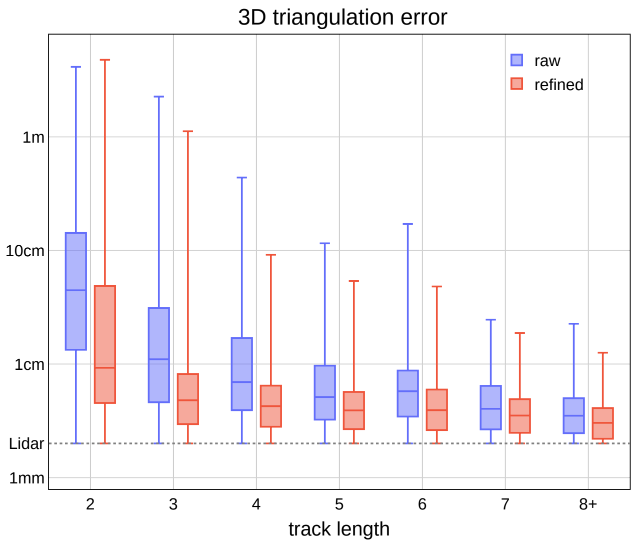

Figure 1: **Triangulation errors vs. track length.** The initial, unrefined output, based on geometric BA, exhibits high errors for 3D points that are observed by few images (low track length). Our refinement significantly reduces these errors and brings the accuracy of the sparse point cloud close to the ground truth acquired by Lidar (2mm accuracy).

SuperPoint ↳ Refinement	KA	BA	qKA	qBA	AUC (%) 1mm	1cm	10cm
unrefined					15.38	51.20	82.33
↳ refined	✓				16.15	53.34	82.49
↳ refined	✓	✓			16.92	54.71	84.08
↳ refined	✓	✓	✓		38.46	70.44	85.28
↳ refined (full)	✓	✓	✓	✓	**40.00**	**71.97**	**86.86**
↳ Patch Flow	✓		✓		28.46	63.04	86.65

Table 1: **Ablation study for pose estimation.** The accuracy of the camera pose is improved by refining the map (KA and BA) and by refining the query keypoints before (qKA) and after (qBA) pose estimation. The largest improvement is brought by qKA. It increases the number of inlier matches and the likelihood of finding the correct pose with RANSAC.

A.2. Camera pose estimation

We analyze in Table 1 how the different kinds of adjustments impact the accuracy of camera localization. The full method presented in the main paper first refines the 3D SfM model with featuremetric keypoint and bundle adjustments. It then refines each keypoint in the query image using its tentative 2D-3D correspondences by minimizing the featuremetric error between its observation in the query and the most similar observation of the respective 3D points. Refining the query keypoints before RANSAC increases the number of inlier matches and stabilizes the pose estimation in challenging scenarios where few 3D points are matches.

Once an initial pose is estimated with PnP+RANSAC, we refine it via a small featuremetric bundle adjustment over the inlier correspondences. This optimizes each query keypoint against the closest descriptor within the matched track. As opposed to refining each query keypoint against all observations in the matched track, this has the benefit of scaling linearly in the number of query keypoints and yields a similar accuracy.

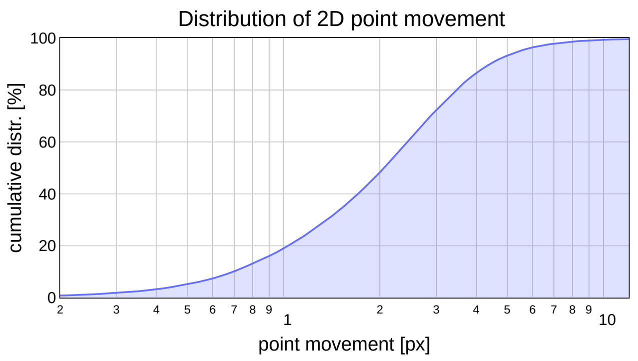

Figure 2: **Distribution of point movements.** We show the cumulative distribution of the distance traveled by the 2D keypoints during the featuremetric refinement of SuperPoint with KA and BA. 60% of the points move by fewer than 2 pixels and 99% remain within 8 pixels of the initial detections.

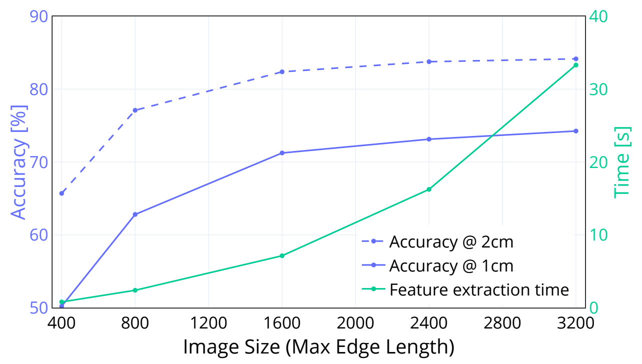

Figure 4: **Impact of the image resolution.** Increasing the image resolution increases the accuracy, but at the cost of longer feature extraction time and higher VRAM requirements. For all experiments on ETH3D, we used a maximum edge length of 1600px, which is very close to saturating the accuracy while providing low run times.

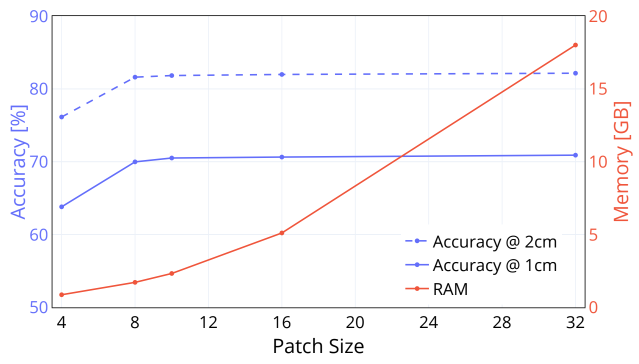

Figure 3: **Impact of the patch size.** Smaller patches for each observation significantly reduce memory requirements but can impair the accuracy of the refinement. Patches of size 10×10 offer a good trade-off with high accuracy and moderate memory consumption.

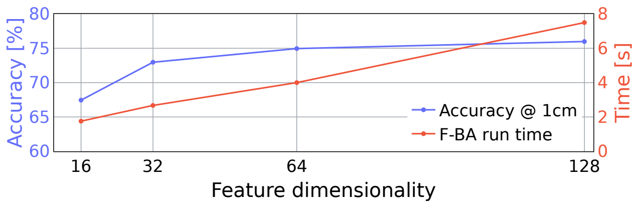

Figure 5: **Impact of the feature dimensionality.** Dense features computed by S2DNet can be naively reduced to accelerate the featuremetric bundle adjustment by 2 while incurring only a minor drop of triangulation accuracy.

B. Impact of various parameters

B.1. Patch size

Figure 2 shows how much our refinement displaces the detected keypoints during the triangulation of SuperPoint on *Courtyard* using dense features extracted from 1600x1066-pixel images. When using full feature maps without any constraints in keypoint adjustment, most points are moved by more than 1 pixel, but most often by less than 8 pixels. This confirms that storing the feature maps as 16×16 patches is sufficient and rather conservative.

We show in Figure 3 the accuracy of the triangulation for various patch sizes. Smaller 10×10 patches achieve sufficient accuracys and require significantly less memory.

B.2. Image resolution

The image resolution at which the dense features are extracted has a large impact on the accuracy of the refinement. In Figure 4 we quantify in the impact on both triangulation accuracy and run time for the ETH3D *Courtyard* scene (38 images). The accuracy drops significantly when the resolution is smaller than 1600×1066px, which amounts to 25% of the full image resolution. Doubling the resolution to 3200×2132px yields noticeable improvements, albeit significantly increases the extraction time and the consumption of GPU VRAM. As a reference, extracting only fine-level S2DNet features (4 convolutions) from 3200×2132px images requires around 10GB of GPU VRAM.

B.3. Reference selection for keypoint adjustment

Selecting some observations as references is necessary to avoid the drift. In a given track, the keypoint adjustment selects the point that is the most connected (topological center), while the bundle adjustment selects the point closest to the robust mean in feature space (feature center). Could we use the feature center for selecting the reference of the keypoint adjustment? By minimizing the feature distance to

	Triangulation ↳ Refinement	Acc. (%) 1cm	Acc. (%) 2cm	Compl. (%) 1cm	Compl. (%) 2cm	Compl. (%) 5cm	track length
SuperPoint	unrefined	18.03	31.97	0.07	0.49	5.03	4.17
	↳ Patch Flow [4]	37.00	55.18	0.15	0.93	7.44	**5.24**
	↳ F-BA	43.65	62.44	0.18	1.06	7.70	4.17
	↳ +F-KA (feat-ref)	45.05	64.84	0.18	1.12	7.76	4.88
	↳ +F-KA (topol-ref)	**46.46**	**65.41**	**0.19**	**1.14**	**8.19**	5.02
D2-Net	unrefined	7.68	13.98	0.02	0.17	2.19	3.29
	↳ Patch Flow [4]	34.64	52.36	0.16	1.00	8.10	**4.99**
	↳ F-BA	39.30	58.59	0.15	0.94	6.99	3.29
	↳ +F-KA (feat-ref)	43.35	62.54	0.19	1.18	8.36	4.49
	↳ +F-KA (topol-ref)	**44.21**	**64.22**	**0.20**	**1.20**	**8.72**	4.63

Table 2: **Additional ablation study on ETH3D - Facade.** i) Featuremetric keypoint adjustment significantly improves the completeness, especially for noisy keypoints as in D2-Net. ii) Keypoint adjustment against the topological center in each tentative track (topol-ref) improves the point cloud in accuracy and completeness over KA towards the robust feature center (feat-ref) because it allows to merge tracks.

this unique reference, we could reduce the number of residuals from quadratic (pairwise constraints) to linear (unary constraints) and thus accelerate the optimization.

Retaining pairwise constraints however allows the optimization to separate tracks that were incorrectly merged by the track separation algorithm. This is not necessary in the bundle adjustment, as tracks are already filtered by the robust geometric estimation and can thus be assumed to be correct, but is common for unverified track. We evaluate the impact of the reference selection in the keypoint adjustment and report the results in Table 2. For both SuperPoint and D2-Net, using the feature center results in lower completeness and accuracy than the topological center. It also results in a lower track length, which confirms that the topological reference allows to retain incorrectly-merged tracks. Since the feature center still performs relatively well, it could be considered in case of tighter computational constraints.

Furthermore, Table 2 highlights the importance of the featuremetric keypoint adjustment. The benefits are larger for D2-Net, which detects very noisy keypoints. As a consequence, many correct albeit noisy matches are rejected by the geometric verification. Our keypoint adjustment not only allows more points to be triangulated, thus increasing the completeness of the model, but also increases the accuracy of the triangulated points.

B.4. Number of feature levels

Using multiple feature levels enlarges the basin of convergence but increases the computational requirements. The radius of convergence that is required depends on the noise of the keypoint detector and on the resolution of the image from which keypoints are detected. When performing detection and refinement at identical image resolutions, the optimal displacement is at most a few pixels for most keypoint detectors. In this case, the fine level of S2DNet feature maps is sufficient. We empirically measured that its radius of convergence is approximately 3 pixels, although the multiview constraints enable to refine over much larger distances.

We thus use a single feature level for all experiments involving SIFT, SuperPoint, and R2D2. D2-Net require a different treatment, as its detection noise is significantly larger. This is partly due to the aggressive downsampling of its CNN backbone and to the low resolution of its output heatmap. As a consequence, we employ both fine and medium feature levels for D2-Net. Both keypoint and bundle adjustments run the optimization successively at the coarser and finer levels.

B.5. Dimensionality of the features

Throughout this paper, we used 128-dimensional dense features extracted by S2DNet [5]. Relying on compact features would easily reduce the memory footprint and the run time of the refinement. To demonstrate these benefits, we show in Figure 5 the relationship between the dimension, the run time of the BA, and the triangulation accuracy when retaining only the first k channels of the S2DNet features. Features with fewer dimensions yield a faster refinement. The accuracy drops moderately but we expect a smaller reduction with features explicitly trained for smaller dimensions.

C. Cost map approximation

We mention in 4.4 that the memory efficiency of the bundle adjustment can be improved by precomputing the featuremetric cost. We provide here more details.

Description: Given D-dimension features, the featuremetric bundle adjustment (Eq. 5) involves residuals and Jacobian matrices of dimension D. Unlike the keypoint adjustment, which can optimize tracks independently, all bundle parameters are updated simultaneously and the memory requirements are thus prohibitive. Given the 2D reprojection $\mathbf{p}_{ij} = \Pi\left(\mathbf{R}_i\mathbf{P}_j + \mathbf{t}_i, \mathbf{C}_i\right)$, this formulation loads in memory the dense features $\mathbf{F}_i$, interpolates them at $\mathbf{p}_{ij}$, and compute the residuals $\mathbf{r}_{ij} = \mathbf{F}_i\left[\mathbf{p}_{ij}\right] - \mathbf{f}^j$ for the cost $E_{ij} = \left\|\mathbf{r}_{ij}\right\|_\gamma$.

To reduce the memory footprint, we can exhaustively precompute patches of feature distances and treat them as one-dimensional residuals $\bar{\mathbf{r}}_{ij} = \left\|\mathbf{F}_i - \mathbf{f}^j\right\|\left[\mathbf{p}_{ij}\right]$. The cost then becomes $\bar{E}_{ij} = \gamma(\bar{\mathbf{r}}_{ij})$. Such distances only need to be computed once since the reference $\mathbf{f}^j$ is kept fixed throughout the optimization. This precomputed cost reduces the peak memory by a factor D, with often D=128. It is similar to the Neural Reprojection Error recently introduced by Germain *et al.* [6] for camera localization.

Analysis: Swapping the distance computation and the sparse interpolation introduces an approximation error. We first write the bilinear or bicubic interpolation as a sum over

SuperPoint ↳ Refinement	Acc. (%) 1cm	Acc. (%) 2cm	Compl. (%) 1cm	Compl. (%) 2cm	Time (s)	Memory (GB)
unrefined	64.27	76.47	0.37	1.44	-	-
↳ ours (exact)	**81.31**	**88.50**	**0.47**	**1.74**	42.22	7.3
↳ ours (cost maps)	80.27	87.81	**0.47**	1.72	**29.86**	**0.15**

Table 3: **Triangulation with cost map approximations.** Using precomputed cost maps increase the efficiency of the bundle adjustment with a marginal loss of accuracy.

features $\mathbf{F}_k$ on the discrete grid:

$$\mathbf{F}[\mathbf{p}] = \sum_k w_k \mathbf{F}_k \quad \text{with} \quad \sum_k w_k = 1 \ . \tag{1}$$

We assume that the features are L2-normalized $\|\mathbf{F}_k\| = 1$, such that $\|\mathbf{F}[\mathbf{p}]\| \approx 1$. For a squared loss function, the approximation error can then be written as:

$$\begin{aligned} &\|\mathbf{F} - \mathbf{f}\|^2 [\mathbf{p}] - \|\mathbf{F}[\mathbf{p}] - \mathbf{f}\|^2 \\ &\approx 1 - \|\mathbf{F}[\mathbf{p}]\|^2 = \frac{1}{2} \sum_k \sum_l w_k w_l \|\mathbf{F}_k - \mathbf{F}_l\|^2 \ . \end{aligned} \tag{2}$$

This error is zero at points on the discrete grid and increases with the roughness of the feature space. This approximation thus displaces the local minimum of the cost by at most 1 pixel but most often by much less.

Improvement: This approximation however degrades the correctness of the approximate Hessian matrix that the Levenberg-Marquardt algorithm [7] relies on for fast convergence. We found that also optimizing the squared spatial derivatives of this cost significantly improves the convergence. This simply amounts to augmenting the scalar residual map with dense derivative maps:

$$\tilde{\mathbf{r}}_{ij} = \begin{pmatrix} \left\|\mathbf{F}_i - \mathbf{f}^j\right\| \\ \frac{\partial \left\|\mathbf{F}_i - \mathbf{f}^j\right\|}{\partial x} \\ \frac{\partial \left\|\mathbf{F}_i - \mathbf{f}^j\right\|}{\partial y} \end{pmatrix} [\mathbf{p}_{ij}] \ . \tag{3}$$

This improvement results in three-dimensional residuals, which is still smaller than D when D=128. Using the spatial derivatives, we can also compute an exact, more accurate bicubic spline interpolation of the cost landscape.

Evaluation: We now show experimentally that this approximation often does not, or only minimally, impairs the accuracy of the refinement. Table 3 reports the results of the triangulation of SuperPoint features on the ETH3D dataset. The approximation reduces the accuracy by less than 1% and does not alter the completeness. It however significantly reduces the memory consumption of the bundle adjustment, allowing it to scale to thousands of images. Note that all experiments in Sections 5.1, 5.2, and 5.3 do not use the cost map approximation as the corresponding scenes are relatively small.

D. Experimental details

D.1. ETH3D - Sections 5.1 and 5.2

For the experiments on ETH3D, we use the evaluation code provided by Dusmanu *et al.* [4]. We use the original implementations of SuperPoint [2], D2-Net [3], and R2D2 [11], and extract root-normalized SIFT [10] features using COLMAP [18]. For both sparse and dense feature extraction, the images are resized so that their longest dimension is equal to 1600 pixels. The tentative matches are filtered according to the recipe described in [4].

D.2. Structure-from-Motion - Section 5.3

We tune the hyperparameters on the training scenes *Temple Nara Japan*, *Trevi Fountain*, and *Brandenburg Gate*. The results in the main paper are computed on the test scenes *Sacre Coeur*, *Saint Peter's Square*, and *Reichstag*, using the data and code provided by the challenge organizers.

For SIFT [10], we use the mutual check, a ratio test with threshold 0.85 for the multi-view and 0.9 for the stereo tasks, and DEGENSAC with an inlier threshold of 0.5px. For D2-Net [3], we use the mutual check and inlier thresholds of 2px and 0.5px for raw and refined keypoints, respectively. For SuperPoint+SuperGlue [2, 14], we do not use additional match filtering and we select an inlier thresholds of 1.1px and 0.5px for raw and refined keypoints, respectively. All sparse local and dense features are extracted at full image resolution, which is generally not larger than 1024px.

D.3. Ablation study - Section 5.4

The triangulation metrics are reported for the ETH3D scene *Facade*, which is the largest with 76 images. We use SuperPoint local features as they perform best in all earlier experiments and we store dense feature maps in every experiment. The localization AUC is measured over all 13 scenes in ETH3D with 10 holdout images per scene. We now detail the different baselines.

Localization is achieved in "F-KA" by first refining the keypoints, triangulating the map and finally performing query keypoint adjustment as described in section A.2. For localization with "F-BA", we refined the triangulated model using featuremetric bundle adjustment and then refined the pose from PnP+RANSAC using qBA.

In the entry "w/ F-BA drift", we use the robust reference (Eq. 7) to select the observation in each track which is most similar to the robust reference as the source frame. The optimizer then minimizes the error between each other observation and the current, moving reference of the source frame. Since only the index of the source frame is fixed during the optimization, this method does not account for drift, which appears to yield higher accuracy but suffers from repeatability problems during localization.

The baseline "PatchFlow + F-BA" uses the keypoint refinement from Dusmanu *et al*. [4] as initialization, and runs our featuremetric bundle adjustment on top of it. We used the exact same parameters for PatchFlow as presented in [4].

The entry "higher resolution" corresponds to input images at double the resolution than all the other experiments, i.e. 3200 pixels in the longest dimension.

For the "photometric" baseline, we use RGB images (while Woodford *et al*. [20] use grayscale images), we warp patches of 4×4 pixels at the featuremap resolution (1600 pixels in the longest dimension) with fronto-parallel assumption, and apply normalized cross correlation (NCC). Identically to our featuremetric BA and to LSPBA [20], the source frame is selected as the observation closest to the robust mean.

We report results for dense features extracted from a VGG-16 CNN, trained on ImageNet [1], at the layer `conv1_2` (64 channels) and for the fine feature map predicted by PixLoc [15] (32 channels). The model of PixLoc, trained on MegaDepth [8], was kindly provided by its authors. In DSIFT [9] (128 channels), we apply a bin size of 4 and a step size of 1 and refer to the VLFeat implementation [19] for more details.

D.4. Scalability

All experiments were conducted on 8 CPU cores (Intel Xeon E5-2630v4) and one NVIDIA RTX 1080 Ti. The subsets from the Aachen Day-Night v1.1 model [16, 17, 21] were selected as the images with the largest visibility overlap, in descending order. To accelerate the feature matching, each image was matched only to its top 20 most covisible reference images in the original Aachen SfM model. We use SuperPoint [2] features and match image pairs with the mutual check and distance thresholding at 0.7. During BA, we apply the sparse Schur solver from Ceres for each linear system in LM, while we use sparse Cholesky in KA, similar to [4]. Featuremetric bundle adjustment is stopped after 30 iterations while KA runs for at most 100 iterations and stops when parameters change by less than 10^{-4}.

To refine the full Aachen Day-Night model, we use SuperPoint features matched with SuperGlue [14] from the Hierarchical Localization toolbox [12, 13]. We refine the keypoints with KA, then triangulate the points with fixed poses from the reference model. Finally, we run a full bundle adjustment of the model with the proposed approximation by cost maps.

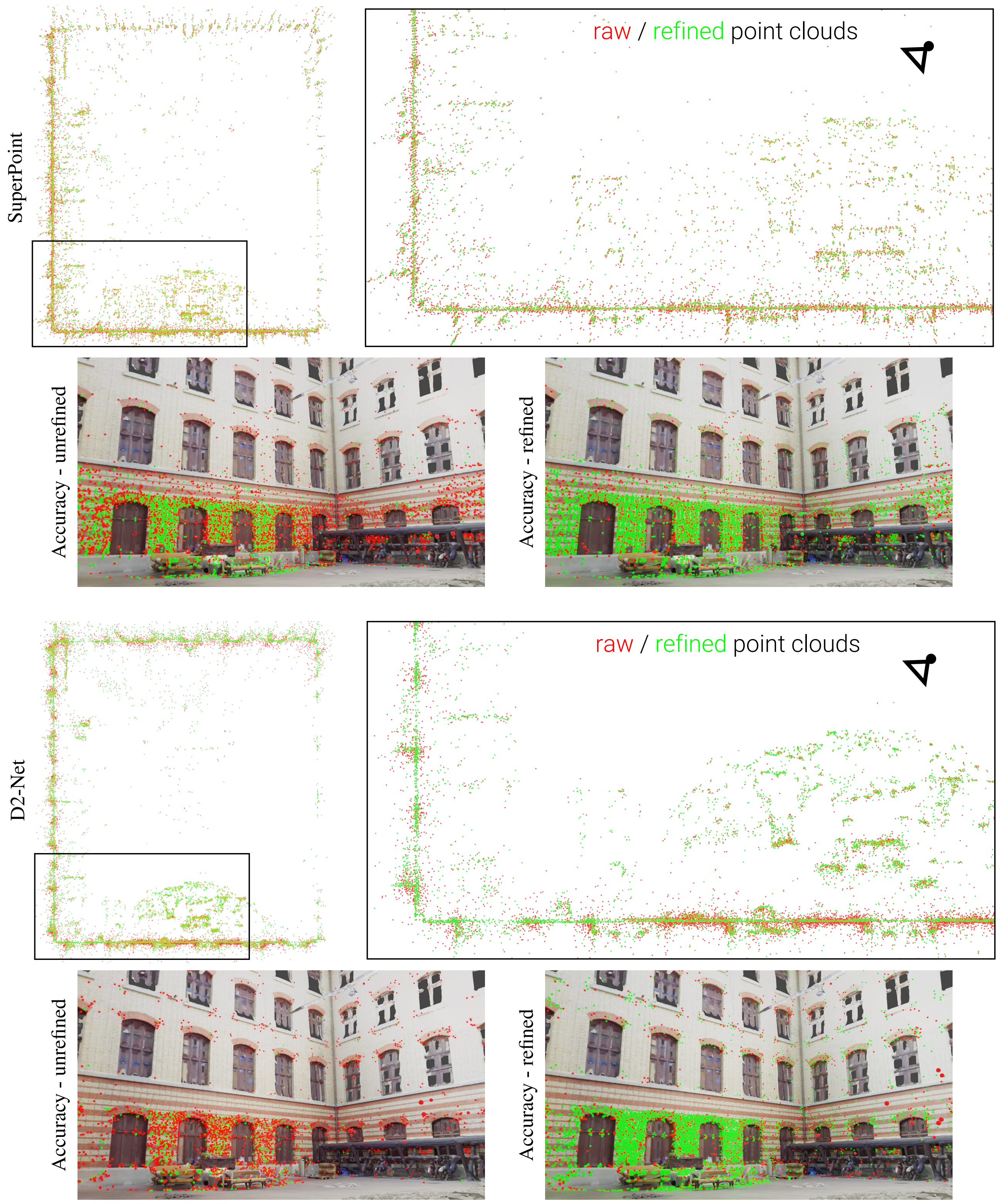

Figure 6: **Refinement on ETH3D *Courtyard*.** In the top parts, we show for both SuperPoint (top) and D2-Net (bottom) top-down views of the sparse point clouds triangulated with raw (in red) and refined (in green) keypoints. The refined point clouds better fit the geometry of the scene, especially on planar walls. In the lower parts, we also show images in which points are colored as accurate (in green) or inaccurate (in red) at 1cm for raw (left) and refined (right) point clouds.

References

[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009. 5

[2] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperPoint: Self-supervised interest point detection and description. In *CVPR Workshop on Deep Learning for Visual SLAM*, 2018. 1, 4, 5

[3] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-Net: A trainable CNN for joint detection and description of local features. In *CVPR*, 2019. 4

[4] Mihai Dusmanu, Johannes L. Schönberger, and Marc Pollefeys. Multi-View Optimization of Local Feature Geometry. In *ECCV*, 2020. 3, 4, 5

[5] Hugo Germain, Guillaume Bourmaud, and Vincent Lepetit. S2DNet: Learning accurate correspondences for sparse-to-dense feature matching. In *ECCV*, 2020. 3

[6] Hugo Germain, Vincent Lepetit, and Guillaume Bourmaud. Neural Reprojection Error: Merging feature learning and camera pose estimation. In *CVPR*, 2021. 3

[7] Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of applied mathematics*, 2(2):164–168, 1944. 4

[8] Zhengqi Li and Noah Snavely. MegaDepth: Learning single-view depth prediction from internet photos. In *CVPR*, 2018. 5

[9] Ce Liu, Jenny Yuen, and Antonio Torralba. SIFT Flow: Dense correspondence across scenes and its applications. *TPAMI*, 2010. 5

[10] David G Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 4

[11] Jerome Revaud, Philippe Weinzaepfel, César De Souza, Noe Pion, Gabriela Csurka, Yohann Cabon, and Martin Humenberger. R2D2: Repeatable and reliable detector and descriptor. In *NeurIPS*, 2019. 4

[12] Paul-Edouard Sarlin. Visual localization made easy with hloc. https://github.com/cvg/Hierarchical-Localization/. 5

[13] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *CVPR*, 2019. 5

[14] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *CVPR*, 2020. 4, 5

[15] Paul-Edouard Sarlin, Ajaykumar Unagar, Måns Larsson, Hugo Germain, Carl Toft, Viktor Larsson, Marc Pollefeys, Vincent Lepetit, Lars Hammarstrand, Fredrik Kahl, and Torsten Sattler. Back to the Feature: Learning robust camera localization from pixels to pose. In *CVPR*, 2021. 5

[16] Torsten Sattler, Will Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, Fredrik Kahl, and Tomas Pajdla. Benchmarking 6DOF outdoor visual localization in changing conditions. In *CVPR*, 2018. 5

[17] Torsten Sattler, Tobias Weyand, Bastian Leibe, and Leif Kobbelt. Image retrieval for image-based localization revisited. In *BMVC*, 2012. 5

[18] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 4

[19] Andrea Vedaldi and Brian Fulkerson. VLFeat: An open and portable library of computer vision algorithms. In *ACM international conference on Multimedia*, 2010. 5

[20] Oliver J Woodford and Edward Rosten. Large scale photometric bundle adjustment. In *BMVC*, 2020. 5

[21] Zichao Zhang, Torsten Sattler, and Davide Scaramuzza. Reference Pose Generation for Long-term Visual Localization via Learned Features and View Synthesis. *IJCV*, 2020. 5