Learning of low-level feature keypoints for accurate and robust detection: Supplementary materials

Suwichaya Suwanwimolkul   Satoshi Komorita   Kazuyuki Tasaka
KDDI Research, Inc.
{su-suwanwimolkul, sa-komorita, ka-tasaka}@kdd-research.jp

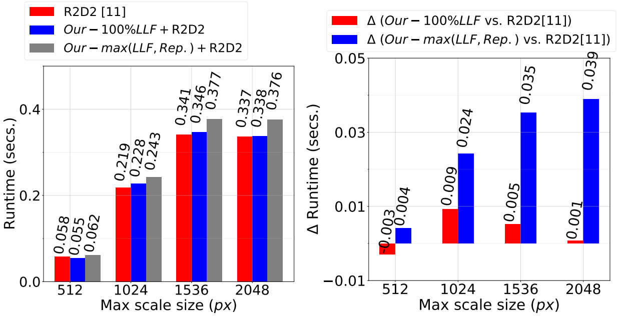
This document is the supplementary material for *Learning of low-level feature keypoints for accurate and robust detection*. In this supplementary material, we provide additional results and analysis to support the results in the main paper. This document covers the following contents:

1. The impact of feature extraction parameters

The performance of our method as well as R2D2 is affected by the following settings: (1) the scale factor; (2) the range of scale detection sizes; and (3) the number of keypoints. To observe the impact of the parameter settings, we fix the scale factor to $2^{0.25}$ and the minimum scale detection size to 256. Therefore, we evaluate the performance at different maximum sizes and number of keypoints. The following sections discuss the impact of maximum sizes and number of keypoints on runtime performance, mean matching accuracy (*MMA*), and mean matched error (*MME*). All the evaluations are performed on the HPatch datasets [1].

1.1. Runtime performance.

The runtime performance is mainly determined by the scale detection size, which is associated with the spatial dimension $H \times W$ of each layer in R2D2 as well as our *LLF* detector. We evaluate our 100%*LLF*+R2D2 and our max$(LLF, Rep.)$+R2D2, against R2D2 [11], at the maximum scale detection size (*max. scale size*) of 512*px*, 1024*px*, 1536*px*, and 2048*px*. The measurements are performed on NVIDIA GeForce RTX 2080 Ti GPU. We provide the average runtime in Figure 1a and calculate the difference between our runtime and R2D2's in Figure 1b. The number of keypoints is set to 5K. As *max. scale size* increases, the runtime of every method increases. Our 100%*LLF*+R2D2 has the runtime similar to R2D2 because we have replaced R2D2's repeatable detector with our *LLF* detector. Meanwhile, our max$(LLF, Rep.)$+R2D2 requires additional runtime (4 ms - 39 ms) because the *LLF* detector is used in addition to the existing detectors.

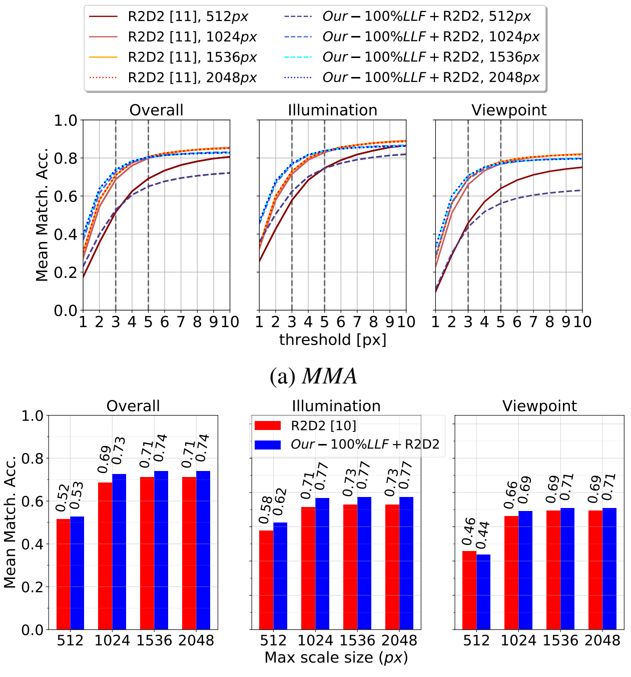


(a) Total runtime (b) Difference w.r.t. R2D2

Figure 1: Runtime performance of our method across different scale detection sizes: (a) total runtime; and (b) time difference with respect to R2D2.

1.2. The impact of scale detection size

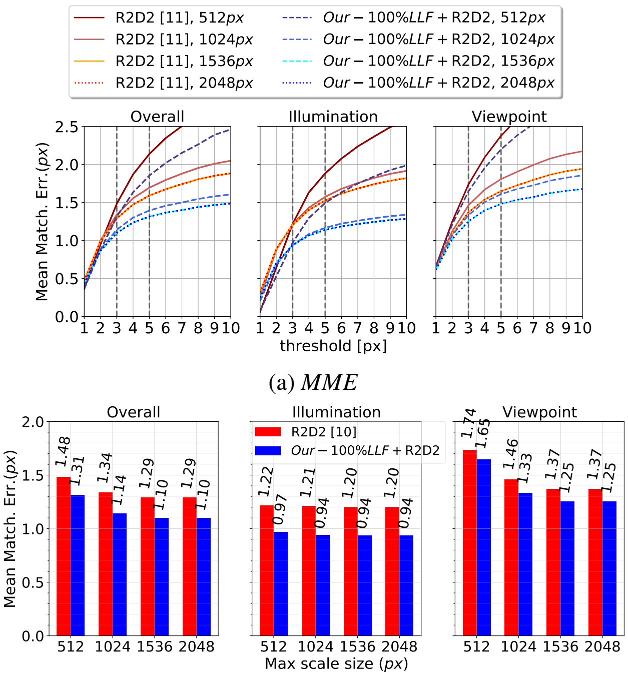
To study the impact of scale detection size, we evaluate the *MMA* of our 100%*LLF*+R2D2 and R2D2 at different maximum scale detection size (*max. scale sizes*): 512*px*, 1024*px*, 1536*px*, and 2048*px*. The results are demonstrated in Figure 2. Both the *MMA* of our 100%*LLF*+R2D2 and R2D2 increase with the *max. scale sizes*. In Figure 2a, our method provides higher *MMA* in most range of *max. scales size*, when the error threshold $< 4px$.



(a) *MMA*

(b) *MMA* at different max. scale sizes (px)

Figure 2: The impact of maximum scale detection size (*max. scales size*) on *MMA* [4]: (a) *MMA* across multiple error thresholds ($1-10px$), and (b) *MMA* across different *max. scales sizes* at the error threshold of $3px$. Our method provides higher *MMA* when the error threshold $< 4px$, and in every *max. scales size* at the error threshold of $3px$.



(a) *MME*

(b) *MME* at different max. scales sizes (px)

Figure 3: The impact of maximum scale detection size (*max. scales size*) on *MME*: (a) *MME* across multiple error thresholds $1-10px$; and (b) *MME* across different *max. scales sizes* at error threshold of $3px$. Our method provides lower *MME* than R2D2 across the *max. scales sizes*.

At the error threshold of $3px$, both methods saturate when *max. scale sizes* $> 1536px$, from Figure 2b. Our 100%*LLF*+R2D2 achieves higher *MMA* (0.74 vs 0.71).

The performance on *MME* is shown in Figure 3. Both the *MME* of our 100%*LLF*+R2D2 and R2D2 reduces as the *max. scale sizes* increase. For each *max. scale size*, our 100%*LLF*+R2D2 provides lower *MME* across different error threshold as shown in 3a, which indicates the improved sub-pixel accuracy. Figure 3b shows the *MME* at different *max. scale sizes* (px) at the error threshold of $3px$. Both methods saturate when *max. scale size* is above $1536px$. R2D2 yields the *MME* of 1.29. Meanwhile, our 100%*LLF*+R2D2 can achieve the lowest *MME* of 1.10, across different *max. scale sizes*.

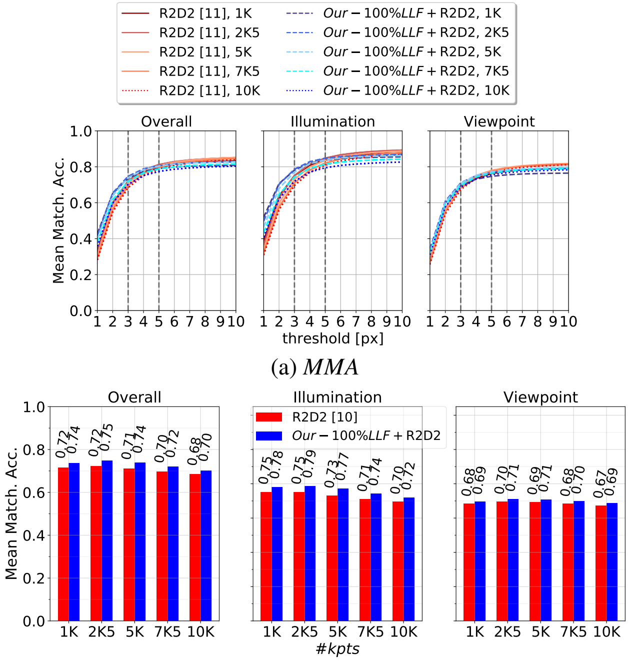
1.3. The impact of the number of keypoints

This section we study the impact of number of keypoints (#*kpts*). We use the *MMA* and *MME* to observe the impact on the performance of our 100%*LLF*+R2D2 and R2D2 [11]. We evaluate the performance at different #*kpts*: 1K, 5K, 7.5K, and 10K. Figure 4a shows the *MMA* for each #*kpts* setting across different error thresholds ($1-10px$). Our 100%*LLF*+R2D2 offers higher *MMA* than R2D2 when the error threshold is $< 5px$. Figure 4b provides the *MMA* across different #*kpts*: 1K, 5K, 7.5K, and 10K, where the error threshold is set to $3px$. The *MMA* of both our 100%*LLF*+R2D2 and R2D2 decreases as #*kpts* increases. Nevertheless, our method provides the higher *MMA* across different cases.

The impact on *MME* is provided in Figure 5. Our 100%*LLF*+R2D2 at each setting of #*kpts* offers lower *MME* than R2D2 across different error thresholds. The worst *MME* of our 100%*LLF*+R2D2 (at #*kpts* = 10K) is still better than the best *MME* of R2D2 (at #*kpts* = 1K) as shown in Figure 5a. This confirms the improved performance on the matched keypoint accuracy by our method. The performance on *MME* across different #*kpts* is in Figure 5b, where the error threshold is set to $3px$. Both our 100%*LLF*+R2D2 and R2D2 provide worse *MME* as #*kpts* increases; nevertheless, 100%*LLF*+R2D2 provides lower *MME* than R2D2 in all cases, across different #*kpts*.

2. Visual results for 3D reconstruction

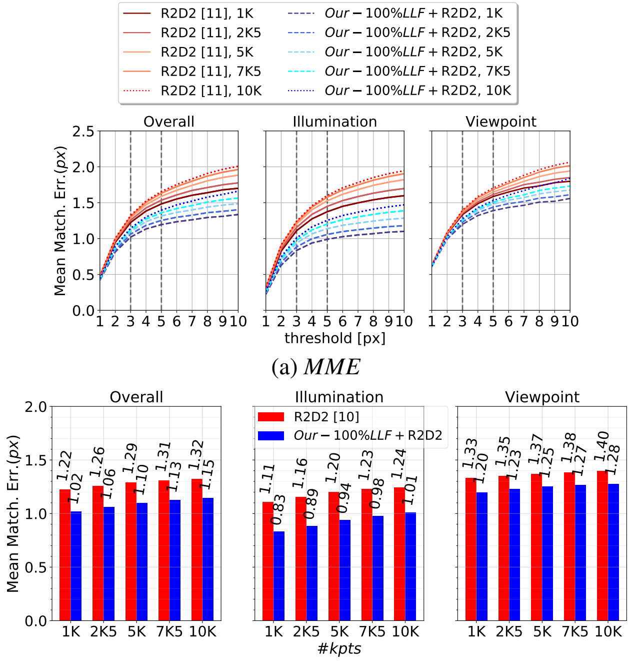
This section provides visual results of 3D reconstruction: more results for the impact of *LLF* keypoints in Section 2.1 and 3D reconstruction by the state-of-the-art in Section 2.2.



(a) *MMA*

(b) *MMA* at different #*kpts*

Figure 4: The impact of number of keypoints (#*kpts*) on *MMA*: (a) *MMA* across multiple error thresholds $1-10px$; and (b) *MMA* across different #*kpts* at the error threshold of $3px$. The *MMA* of our method and R2D2 decreases as #*kpts* increases. Our method has higher *MMA* when the error threshold $< 5px$, and in every #*kpts* at the error threshold of $3px$.



(a) *MME*

(b) *MME* at different #*kpt*

Figure 5: The impact of number of keypoints (#*kpts*) on *MME*: (a) *MME* across multiple error thresholds $1-10px$; and (b) *MME* across #*kpt* at error threshold of $3px$. The *MME* of both R2D2 and our method increases with #*kpts*. Our method gives lower *MME* than R2D2 in all cases.

2.1. Impact of *LLF* keypoints on 3D reconstruction

Here, we provide the additional visual results to demonstrate the impact of *LLF* keypoints on 3D reconstruction of Herzjesu. The proportion of *LLF* keypoints as the percent to the total keypoints from *LLF* and R2D2's repeatable detectors is varied from 0%, 25%, 50%, 100%. Here, we provide the visual results in Figure 6 corresponding the numerical results in the main paper (Table 2). To reflect the number of correct 3D points, we vary the point maximum error thresholds from $3px$ (top), $1px$, and $0.60px$ (bottom), where the lower threshold filters out more erroneous points. From Figure 6, the higher proportion of *LLF* keypoints results in the more complete and correct 3D shape at the low point maximum error thresholds. Our 100%*LLF*+R2D2 offers the most complete and correct 3D shape at the lowest threshold ($0.60px$).

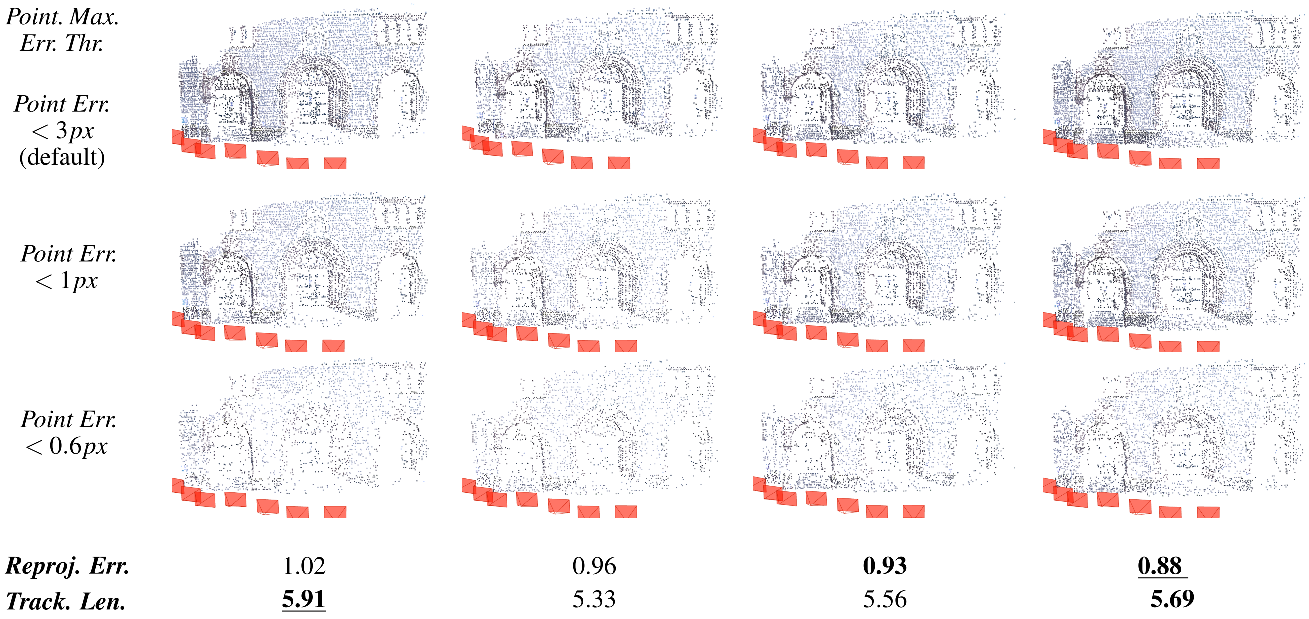
2.2. 3D reconstruction by state-of-the-art methods

The example 3D reconstruction by state-of-the-art local features, namely, (a) SIFT [7], (b) ASLFeat [8], (c) R2D2 [11], and (d) our 100%*LLF*+R2D2, on Herzjesu and Fountain are provided in Figure 8 and 9, respectively. For every method, we set the number of keypoints to 20K and the maximum scale detection size to $2048px$. At first, the noticeable differences between these methods is the distribution of 3D points. Our 100%*LLF*+R2D2 and R2D2 provide the sparse 3D points spreading over the building, yet our 100%*LLF*+R2D2 has slightly more points clustering on the edges and corner of Herzjesu building. Meanwhile, ASLFeat provides dense 3D points. The 3D points of SIFT and ASLFeat densely cluster around the edges and corner.

Then, we varied the point maximum error thresholds to $0.7px$ and the point minimum tracking length to 5 to reflect the amount of correct 3D points. Our 100%*LLF*+R2D2 offers the complete and correct 3D shape in most cases, where our *Reproj. Err.* is the second best after SIFT, and our *Track. Len.* slightly lower than R2D2. ASLFeat offers a good reconstruction in Herzjesu, but a noisy results on Fountain. The numerical results are provided in Table 1.

3. More keypoint detection and matching

Here, we provide additional results and discussion on keypoint detection and matching. In Section 3.1, we provided the comparison on the keypoint detection and matching where ASLFeat v.2 is included. The visual results on



(a) R2D2 (b) Our 25%*LLF*+R2D2 (c) Our 50%*LLF*+R2D2 (d) Our 100%*LLF*+R2D2

Figure 6: Full comparison: (a) R2D2, (b) our 25%*LLF*+R2D2, (c) our 50%*LLF*+R2D2, and (d) our 100%*LLF*+R2D2. The point maximum error thresholds are varied from $3px$ (top), $1px$, and $0.60px$ (bottom). The lower threshold filters out more erroneous points. Our *LLF* keypoints improve the accuracy of 3D points.

the keypoint detection and matching are in Section 3.2.

3.1. The comparison with ASLFeat v.2

The comparison with ASLFeat v.2 is provided in Table 2. We provide this comparison in the Supplementary because ASLFeat v.2 is improved from ASLFeat [8] by using more advanced training data: a large database with additional depth information for training, i.e., blended images and rendered depths, which is out of the scope of our work.

Nevertheless, similar to our report in the main paper (Table 3), our 100%*LLF*+R2D2, have the highest *MMA*, and both of our works, our 100%*LLF*+R2D2 and $\max(LLF, Rep.)$+R2D2, achieve the top three in *MME* and $\epsilon_{IoU}(SL)$. ASLFeat yields moderate results in many area. However, ASLFeat v.2 can achieve the top three performance in *MMA* and repeatability (L) and achieve the best ϵ_{IoU} (L). Nevertheless, similar to KeyNet and SuperPoint, ASLFeat v.2 still has higher error *MME* and ϵ_{IoU} (SL). This indicates that ASLFeat v.2 still has high error in matched keypoints and the lacks of robustness against the changed scales in viewpoint.

3.2. Visual results of keypoint detection & matching

We provide the visual results on keypoint matching in Figure 10. The green lines denote the correct matching, and the red lines denote the wrong matches under the error threshold of $3px$. The results are sorted by the range of geometric noise distributions in HPatches sequences [1], from the easy sequence (top row) to the very tough sequence (at bottom row). *MMA* and *MME* under $3px$ are also provided.

Datasets	**Methods**	***#Reg. Imges***	***#Sparse Points***	***Track. Len.***	***Reproj. Error***	***#Obs. Points***
Herzjesu	SIFT [7]	**8**	3.2K	4.01	**0.531**	13K
8 images	ASLfeat [8]	**8**	**15.3K**	5.14	0.881	**78K**
	R2D2 [11]	**8**	**13.6K**	**5.91**	1.020	**80K**
	Our 100%*LLF*+ R2D2	**8**	13.0K	**5.69**	**0.880**	74K
Fountain	SIFT [7]	**11**	5.7K	4.47	**0.431**	25K
11 images	ASLfeat [8]	**11**	**25.2K**	6.11	1.010	**154K**
	R2D2 [11]	**11**	**16.6K**	**7.53**	1.036	**125K**
	Our 100%*LLF*+ R2D2	**11**	16.3K	**7.31**	**0.883**	119K

Table 1: 3D reconstruction by state-of-the-art methods

From Figure 10, although our method provides similar or lower number of matches, our method offers the least amount of wrong matches, which is associated with the highest *MMA* and the best *MME* among the learning-based methods. Our *MME* is the second best after *SIFT*. However, SIFT has a much higher number of wrong matches, which results in the worse *MMA* in most cases. Meanwhile, ASLFeat and R2D2 has high number of keypoint matches. ASLFeat tends to have more wrong matches than R2D2 and ours. Nevertheless, our *LLF* detector is more selective than R2D2's detector, leading to the superior *MMA* and *MME*.

The visual results of keypoint detection corresponding to the previous matching results are in Figure 11-12. Among

	Overall								Illumination				Viewpoints			
	Feat. Matching			ϵ_{IoU}		Repeatability			Feat. Matching		ϵ_{IoU}		Feat. Matching		ϵ_{IoU}	
Methods	*MMA*	*MME*	*#Inlie.*	*SL*	*L*	*SL*	*L*	*#Corr.*	*MMA*	*MME*	*SL*	*L*	*MMA*	*MME*	*SL*	*L*
SIFT [7]	0.51	**1.014**	232	0.178	0.120	37.8	59.0	402	0.48	0.897	0.118	0.120	0.55	**1.127**	0.237	0.119
SURF [2]	0.47	1.211	213	0.173	0.120	44.2	62.1	451	0.47	1.040	0.109	0.113	0.48	1.378	0.235	0.127
Key.Net [5]	0.72	1.186	**408**	0.138	0.093	**60.3**	**68.2**	**591**	0.72	1.010	0.090	0.092	**0.71**	1.360	0.185	0.094
D2-Net [4]	0.30	1.725	141	0.219	0.183	37.2	54.7	210	0.39	1.607	0.179	0.168	0.22	1.843	0.257	0.197
ASLFeat [8]	0.69	1.178	**358**	0.142	**0.089**	49.8	61.3	**573**	0.72	1.024	**0.088**	0.088	0.66	1.330	0.195	**0.091**
ASLFeat v.2 [8]	**0.73**	1.175	**387**	0.140	**0.087**	50.9	**63.0**	**589**	**0.77**	1.036	**0.083**	**0.084**	0.69	**1.315**	0.195	**0.090**
DELF [9]	0.47	**1.016**	280	0.151	0.128	47.7	60.3	369	**0.89**	**0.043**	**0.005**	**0.011**	0.07	1.986	0.293	0.242
SuperPoint [3]	0.59	1.381	273	0.153	0.110	**57.7**	**79.1**	320	0.65	1.135	0.101	0.101	0.53	1.623	0.202	0.119
R2D2 [11]	0.71	1.265	311	**0.118**	0.096	**51.9**	59.2	559	0.73	1.100	0.099	0.097	0.69	1.428	**0.136**	0.096
Our $\max(LLF, Rep.)$+R2D2	**0.72**	1.083	262	**0.124**	**0.088**	46.6	55.8	570	0.75	**0.835**	0.099	**0.087**	**0.70**	1.327	**0.148**	**0.089**
Our 100%LLF+R2D2	**0.74**	**1.070**	269	**0.126**	0.092	47.3	57.1	562	**0.77**	**0.819**	0.102	0.092	**0.71**	**1.318**	**0.148**	0.092

Table 2: Comparison to state-of-the-art methods on the full HPatches dataset [1] with mean matching accuracy (*MMA*), mean matched keypoint error (*MME*), average intersection over union error (ϵ_{IoU}), and repeatability (%). The error threshold is set to $3px$. Our 100%LLF+R2D2 is the best in *MMA* and achieves the top three in *MME* and ϵ_{IoU} (*SL*) in overall.

all the detected keypoints, the pink color denotes the *inliers*, and the blue color denotes the *outliers* of the matched keypoints. The green color denotes the other detected keypoints. Figure 11 provides the keypoint detection of the easy and the hard sequences. Figure 12 provides the keypoint detection of the tough and the very tough sequences.

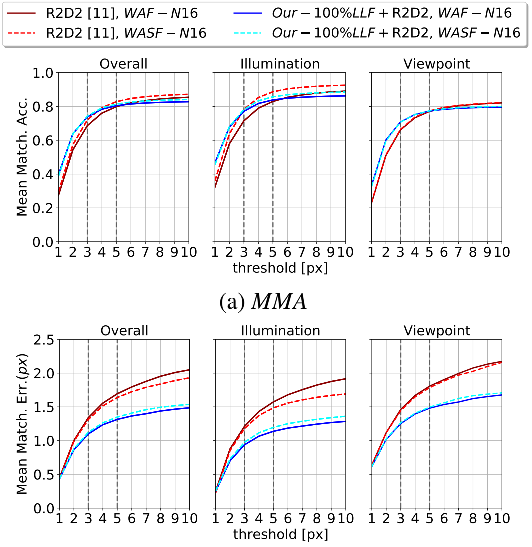
Similar to R2D2, our 100%LLF+R2D2 provides the sparse keypoints. Nevertheless, our 100%LLF+R2D2 has less *outliers* than R2D2, which explains the higher *MMA*. Our keypoints are not as structured as ASLFeat nor SIFT. The keypoints of ASLFeat and SIFT are very dense around the edge and corner in images, and both have more *outliers*.

4. Additional details for training data

From the main paper (Section 4.1, **Baseline and training data.**), we employed the same training data and settings of R2D2-WAF-$N16$ and R2D2-$WASF$-$N16$ released from the official site of [11]. In this section, we clarify the details of training data of WAF and $WASF$ for training our LLF detector and R2D2's backbone from scratches. According to [11], $WASF$ or $W-A-S-F$ is the tag names refer to combination of the following image pairs sets, *i.e.*:

- $W-$ denote *random web images*, i.e., the distractors from a retrieval dataset [10], and the synthetic image pairs are generated by applying random transformations (homography and color jittering) ;
- $A-$ denotes *Aachen database images* where the images are obtained from the Aachen dataset [13, 12], and the previous strategy is used to build the synthetic pairs;
- $S-$ denotes *Aachen style transfer pairs* where the style transfer [6] is used for building pairs from Aachen;
- $F-$ denotes *Aachen optical flow pairs* which are the pairs of nearby views from the Aachen dataset, and the pseudo ground-truth of the correspondence pixels between image pairs is obtained using optical flow [11].

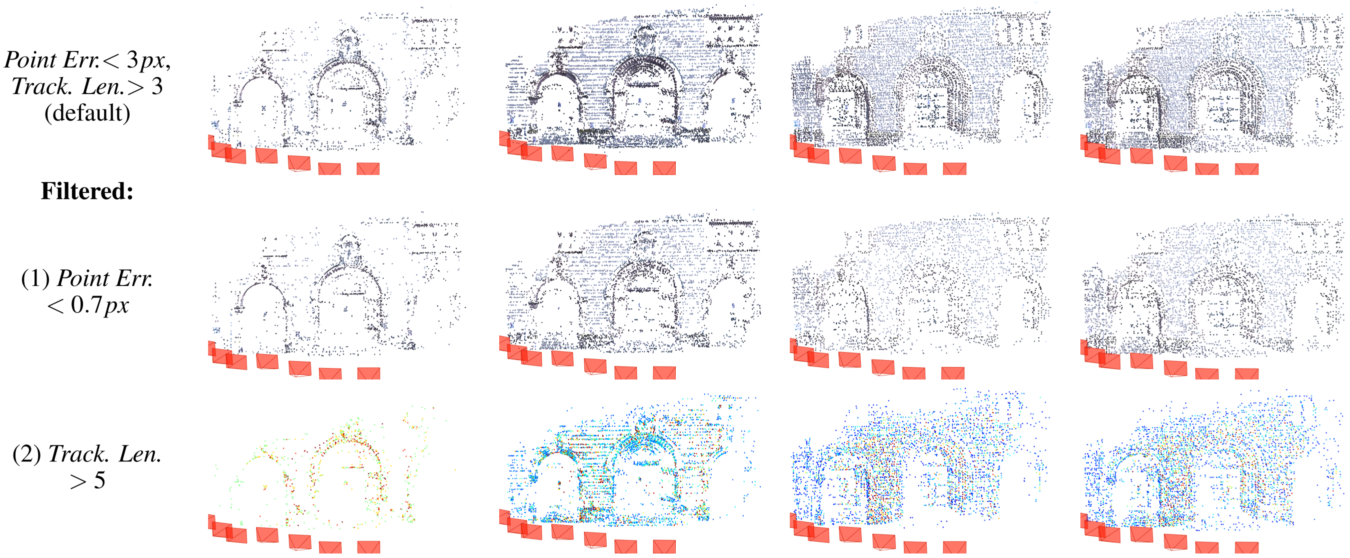
Therefore, $WASF$- refers to the settings where all the image pairs sets are used. Meanwhile, WAF- refers to using *random web images*, *Aachen database images* and *Aachen optical flow pairs* to form training datatset. To confirm consistent performance, we compare our work with R2D2 for both settings, WAF and $WASF$, in Figure 7. Our method yield better *MMA* when error threshold $< 4px$ and better *MME* in all cases for both settings.



(a) *MMA*

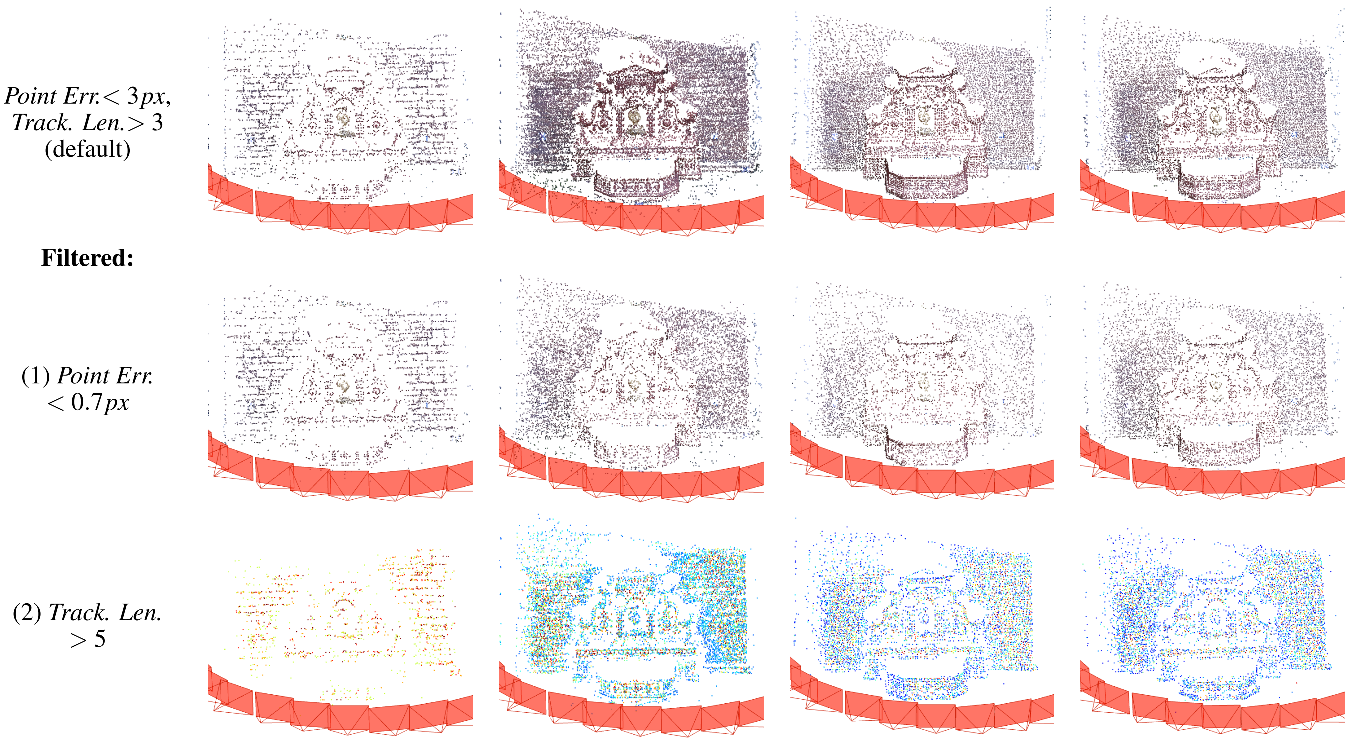
(b) *MME*

Figure 7: Comparison on WAF-and $WASF$-$N16$ settings.


Point Err. $< 3px$, *Track. Len.* > 3 (default)
Filtered:
(1) *Point Err.* $< 0.7px$
(2) *Track. Len.* > 5

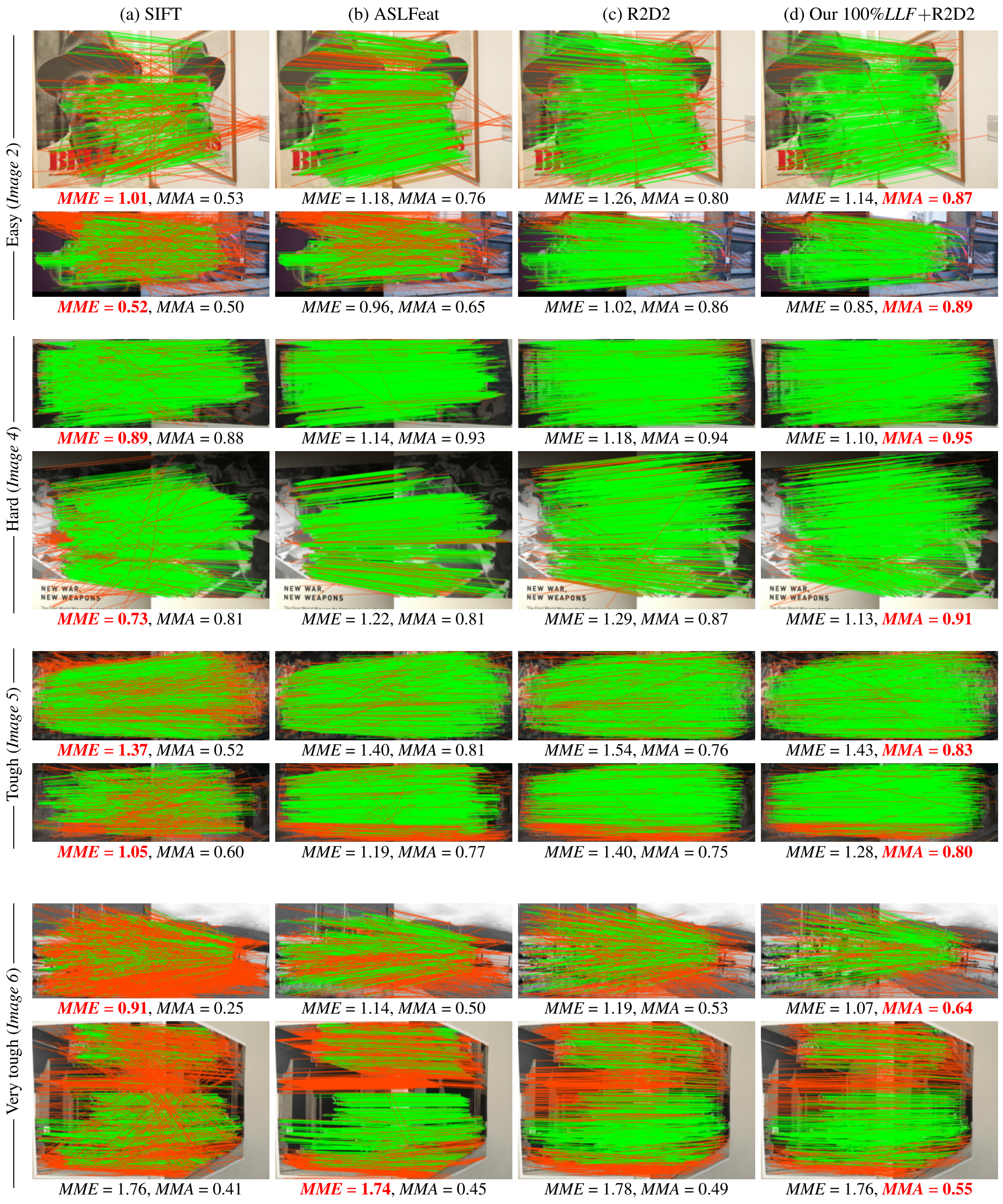
	(a) SIFT	(b) ASLFeat	(c) R2D2	(d) Our 100%*LLF*+R2D2
Reproj. Err.	**0.53**	0.88	1.02	**0.88**
Track. Len.	4.01	5.14	**5.91**	**5.69**

Figure 8: Example 3D reconstruction of Herzjesu: (a) SIFT (b) ASLFeat, (c) R2D2, and (d) our 100%*LLF*+R2D2.


Point Err. $< 3px$, *Track. Len.* > 3 (default)
Filtered:
(1) *Point Err.* $< 0.7px$
(2) *Track. Len.* > 5

	(a) SIFT	(b) ASLFeat	(c) R2D2	(d) Our 100%*LLF*+R2D2
Reproj. Err.	**0.43**	1.01	1.04	**0.88**
Track. Len.	4.47	6.11	**7.53**	**7.31**

Figure 9: Example 3D reconstruction of Fountain: (a) SIFT (b) ASLFeat, (c) R2D2, and (d) our 100%*LLF*+R2D2.



(a) SIFT (b) ASLFeat (c) R2D2 (d) Our 100%*LLF*+R2D2

Easy (*Image 2*)

***MME* = 1.01**, *MMA* = 0.53 *MME* = 1.18, *MMA* = 0.76 *MME* = 1.26, *MMA* = 0.80 *MME* = 1.14, ***MMA* = 0.87**

***MME* = 0.52**, *MMA* = 0.50 *MME* = 0.96, *MMA* = 0.65 *MME* = 1.02, *MMA* = 0.86 *MME* = 0.85, ***MMA* = 0.89**

Hard (*Image 4*)

***MME* = 0.89**, *MMA* = 0.88 *MME* = 1.14, *MMA* = 0.93 *MME* = 1.18, *MMA* = 0.94 *MME* = 1.10, ***MMA* = 0.95**

***MME* = 0.73**, *MMA* = 0.81 *MME* = 1.22, *MMA* = 0.81 *MME* = 1.29, *MMA* = 0.87 *MME* = 1.13, ***MMA* = 0.91**

Tough (*Image 5*)

***MME* = 1.37**, *MMA* = 0.52 *MME* = 1.40, *MMA* = 0.81 *MME* = 1.54, *MMA* = 0.76 *MME* = 1.43, ***MMA* = 0.83**

***MME* = 1.05**, *MMA* = 0.60 *MME* = 1.19, *MMA* = 0.77 *MME* = 1.40, *MMA* = 0.75 *MME* = 1.28, ***MMA* = 0.80**

Very tough (*Image 6*)

***MME* = 0.91**, *MMA* = 0.25 *MME* = 1.14, *MMA* = 0.50 *MME* = 1.19, *MMA* = 0.53 *MME* = 1.07, ***MMA* = 0.64**

MME = 1.76, *MMA* = 0.41 ***MME* = 1.74**, *MMA* = 0.45 *MME* = 1.78, *MMA* = 0.49 *MME* = 1.76, ***MMA* = 0.55**

Figure 10: Qualitative results on HPatch by (a) SIFT, (b) ASLFeat, (3) R2D2, and (4) our 100%*LLF* keypoints+R2D2. The green lines show the correct matching, and the red lines show the wrong matches under the error threshold of 3*px*. The results of keypoint matching are sorted by the range of geometric noise distributions in HPatches sequences [1], from the easy sequence (top row) to the very tough sequence (at bottom row). Our method offers high correct matches with small number of wrong matches which explain the highest *MMA*. Our *MME* is the second best after *SIFT*.

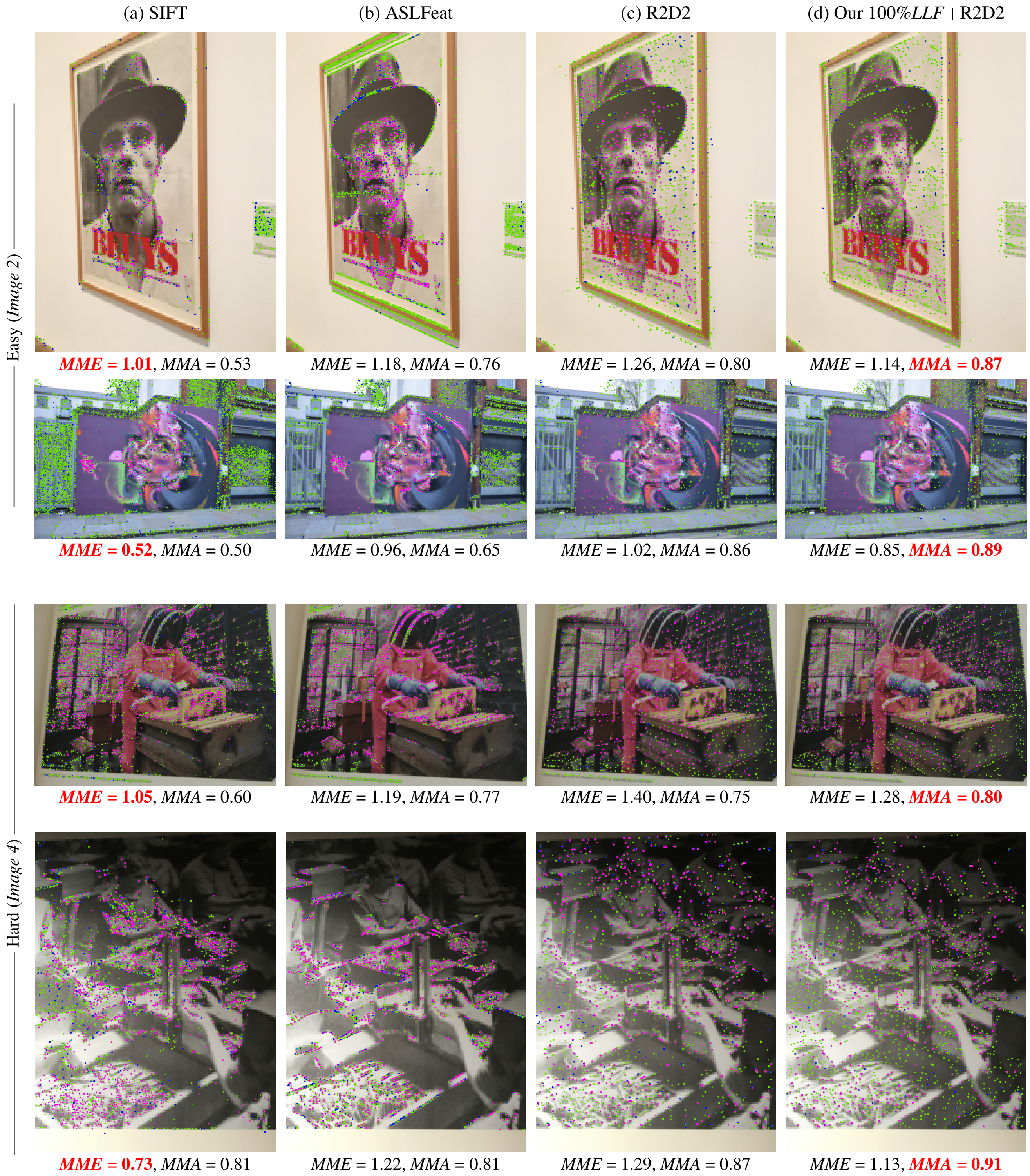


Figure 11: Visual results of keypoints detection by (a) SIFT, (b) ASLFeat (c) R2D2 and (d) our 100%*LLF*+R2D2, corresponding to the previous keypoint matching. The pink color denotes the *inliers*. The blue color denotes the *outliers* of the matched keypoints. The green color denotes the other detected keypoints. From the easy to the hard sequences, our 100%*LLF*+R2D2 less *outliers* than the other methods, which explain the high *MMA* of our keypoints. Similar to R2D2, our keypoints are sparse and not as dense nor structured as ASLFeat and SIFT.

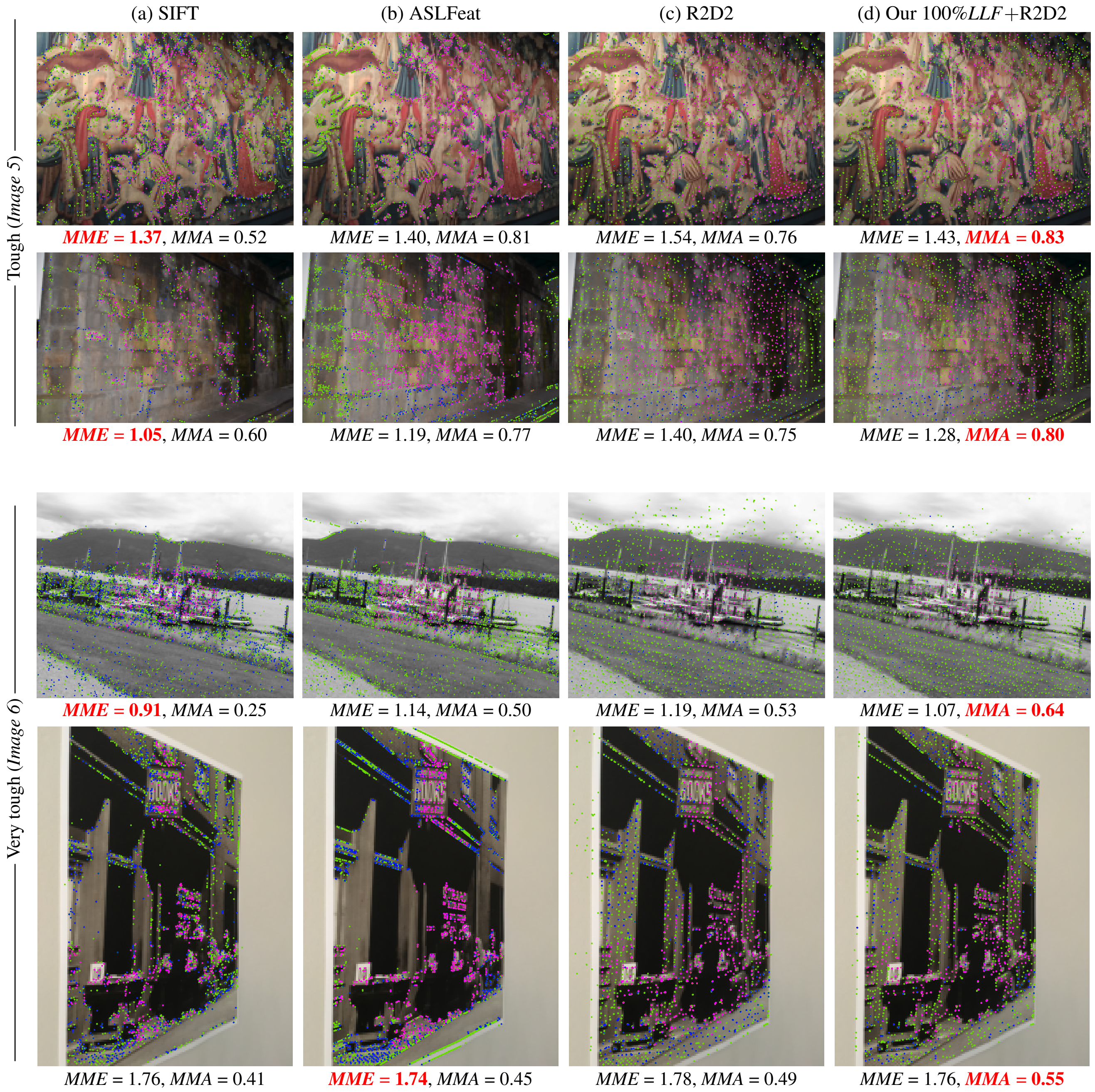


Figure 12: (cont') Visual results of keypoints detection by (a) SIFT, (b) ASLFeat (c) R2D2 and (d) our 100%*LLF*+R2D2. The results of the tough sequence to the very tough sequences are sorted from the top to the bottom rows. The pink and blue color denotes *inliers* and *outliers* among all the keypoints which are denoted by green color. In these tough samples, our 100%*LLF*+R2D2 has notably less *outliers* than the others. ASLFeat and SIFT keypoints are more dense at the edge and corners and much more number of *outliers*.

References

[1] V. Balntas, K. Lenc, A. Vedaldi, and K. Mikolajczyk. HPatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[2] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (SURF). *Computer Vision Image Understanding*, 110(3):346359, June 2008.

[3] D. DeTone, T. Malisiewicz, and A. Rabinovich. SuperPoint: Self-supervised interest point detection and description. In *2018 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 337–33712, 2018.

[4] M. Dusmanu, I. Rocco, T. Pajdla, M. Pollefeys, J. Sivic, A. Torii, and T. Sattler. D2-Net: A trainable CNN for joint description and detection of local features. In *2019 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8084–8093, 2019.

[5] A. B. Laguna, E. Riba, D. Ponsa, and K. Mikolajczyk. Key.net: Keypoint detection by handcrafted and learned cnn filters. In *2019 IEEE International Conference on Computer Vision (ICCV)*, pages 5835–5843, 2019.

[6] Y. Li, M. Liu, X. Li, M. Yang, and J. Kautz. A closed-form solution to photorealistic Image stylization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 468–483, 2018.

[7] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–, 11 2004.

[8] Z. Luo, L. Zhou, X. Bai, H. Chen, J. Zhang, Y. Yao, S. Li, T. Fang, and L. Quan. ASLFeat: Learning local features of accurate shape and localization. *2020 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[9] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han. Large-Scale image retrieval With attentive deep local features. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[10] F. Radenović, A. Iscen, G. Tolias, Y. Avrithis, and O. Chum. Revisiting oxford and paris: Large-scale image retrieval benchmarking. In *2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[11] J. Revaud, P. Weinzaepfel, C. Roberto de Souza, and M. Humenberger. R2D2: Repeatable and reliable detector and descriptor. In *Advances in Neural Information Processing Systems*, 2019.

[12] T. Sattler, W. Maddern, C. Toft, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. Sivic, F. Kahl, and T. Pajdla. Benchmarking 6DOF outdoor visual localization in changing conditions. In *2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[13] T. Sattler, T. Weyand, B. Leibe, and L. Kobbelt. Image Retrieval for image-Based localization revisited. In *Proceedings of the British Machine Vision Conference*, pages 76.1–76.12. BMVA Press, 2012.