Attention-based Multi-Reference Learning for Image Super-Resolution - Supplementary Material

Marco Pesavento Marco Volino Adrian Hilton
Centre for Vision, Speech and Signal Processing
University of Surrey, UK
{m.pesavento,m.volino,a.hilton}@surrey.ac.uk

This document presents additional details on the architecture of the Super-Resolution (SR) module and on the training implementation of AMRSR (Section 1). A pseudocode of AMRSR is then proposed with a legend summarizing the notation used in the main paper (Section 2) . Ablation study results are added by training AMRSR with all the losses (visual-oriented) for different number of references and for different configurations of the attention mapping (Section 3). Another ablation study is conducted by changing the number of parts that the LR input and the reference images are divided into, with a focus also on the inference time (Section 3). More visual comparisons between AMRSR and the other approaches cited in the experimental results of the original paper are illustrated (Section 4). Further results of the comparison between AMRSR and CIMR [1] are finally presented (Section 5).

1. Super-resolution module and training implementation

AMRSR aims to super-resolve the low-resolution (LR) input by concatenating its feature vector with the feature vectors created by the hierarchical attention-based similarity mapping. To this purpose, a generative adversarial network (GAN) is applied with a discriminator that is the same as the one used in [16]. After an initial feature extraction of the input image with a residual block, a generator, whose layers are listed in Table 1, retrieves the output image.
The multi-scale approach that creates the feature vectors in each level of the hierarchy of AMRSR adopts three layers of the VGG-19 network [13]: relu1_1, relu2_1, and relu3_1. This last layer is used to perform the convolutional operation that estimates the similarities between the feature vectors. The computed correspondences are then projected to the other two layers to create the related vectors.
The network is pre-trained with the reconstruction loss for 5 epochs and then trained for 100 epochs with the other losses. The l_2 versions are trained for 100 epochs with the reconstruction loss. Adam optimizer is adopted with a learning rate of 1e-4. The datasets have been augmented by flipping and rotating the data. The weights for L_1, L_{per}, L_{adv}, and L_{tex} are 1, 1e-4, 1e-6, and 1e-4, respectively.

Id	Network Layers
0	ImgFeature=Residual Blocks(Input)
1	Concat: ImgFeature, O_3
2	Conv, LeakyReLU
3-18	Residual Blocks (Conv,ReLu,Conv)
19	ImgFeature + output18
20	Conv, PixelShuffle(2x), LeakyReLU
21	Concat:output20,O_2
22	Conv, LeakyReLU
23-30	Residual Blocks (Conv,ReLu,Conv)
31	output20+output30
32	Conv, PixelShuffle(2x), LeakyReLU
33	Concat:output32,O_1
34	Conv, LeakyReLU
35-38	Residual Blocks (Conv,ReLu,Conv)
39	output32+output38
40	Conv, PixelShuffle(2x), LeakyReLU

Table 1: Architecture of AMRSR generator.

Symbol	Meaning
I_{LR}	Low resolution input image
I_{SR}	Super resolved image
I_{ref}	Reference image
$m = 1..N_M$	Number of reference images
$r = 1..N_R$	Number of parts a reference is divided into
$i = 1..N_I$	Number of parts the input is divided into
$l = 1..N_L$	Number of levels of the hierarchy
$c = 1..N_C$	Number of parts the input feature vector is divided into
$\phi^c(I_{LR})$	Set of N_C subvectors of the input feature
$\phi^r(I^m_{ref})$	Set of features of the N_R parts of the N_M references
s^l_k	Similarity map for layer l. $k = c, r, m$
O^l_{ref}	Output map of level l containing reference features
P	Patches of patch-match approach
W	Set of output weights

Table 2: Legend of AMRSR notation.

2. AMRSR legend and algorithm

Table 2 presents a legend with the notations used in Section 3 of the main paper.

Algorithm 1 represents the pseudocode of AMRSR algorithm in the case that $N_M > 1$ and $N_R > 1$. The cases when $N_M = 1$ or $N_R = 1$ can be represented with a simple modification of the pseudocode. If $N_M = 1$ and $N_R = 1$, only the "INPUTATT" function is executed. The function INPUTATT is defined by Equation 1 and 2 from the main paper, while REFATT is defined by Equation 1 and 3.

Algorithm 1 AMRSR ALGORITHM

Divide N_M reference images in N_R parts
Extract feature vectors of reference image parts $\phi^r(I^m_{ref})$
Extract feature vectors of LR input $\phi(I_{LR})$
Divide $\phi(I_{LR})$ into N_C subvectors $\rightarrow \{\phi^c(I_{LR})\}_{c=1}^{N_C}$
for m = 1:N_M (for all the references) **do**
    for r=1:N_R (for all parts of a reference) **do**
        $O^{1,m,r}_{ref}$=INPUTATT$(\phi^c(I_{LR}), \phi^r(I^m_{ref}), c)$
    end for
    $O^{2,m}_{ref}$=REFATT$(\phi^c(I_{LR}), O^{1,m,r}_{ref}, N_C, r)$
    $W_m = max(\phi^c(I_{LR}) \cdot O^{2,m}_{ref})$
end for
$O = O^3_{ref}$ =REFATT$(\phi^c(I_{LR}), O^{2,m}_{ref}, N_C, m)$
$W(x, y) = W_{m^*}(x, y)$
Image Super-resolution by leveraging O and W

function INPUTATT(X, Y, k)
    $s_k = X_k * \frac{P_k(Y)}{||P_k(Y)||}$
    $M(x, y) = P_{k^*}(Y)(x, y)$
    where $k^* = \text{argmax}_k\, s_k(x, y)$
    return M
end function

function REFATT(X, Y, N_C, k)
    for $j = 1 : N_C$ **do**
        $s_k = X_j * \frac{P(Y_k[j])}{||P(Y_k[j])||}$
        $M[j](x, y) = Y_{k^*}[j](x, y)$
        where $k^* = \text{argmax}_k\, s_k(x, y)$
    end for
    return M
end function

3. Ablation studies: further evaluations

	Nr. references	CU4REF	Sun80	GEMAP	HUMAP
Visual-Oriented	1 Reference	26.77/.7882	30.15/.8162	35.64/.9106	45.38/.9760
	2 References	27.30/.8087	30.27/.8205	35.65/.9108	45.45/.9763
	4 References	**27.49/.8145**	**30.41/.8257**	35.80/.9122	**45.56/.9771**
	8 References	–	–	**35.87/.9145**	45.50/.9767
	2^{nd} best	26.42/.7738	29.72/.7984	34.78/.8963	45.03/.9743

Table 3: Visual-oriented quantitative results of AMRSR obtained by changing the number of references.

Number of reference images and attention mapping: Table 5 and Table 6 of the main paper present the PSNR and SSIM values computed by respectively changing the number of references and modifying the attention mapping mechanism of AMRSR trained with only the reconstruction loss (PSNR-oriented). The visual-oriented results (trained with all the losses) are shown in Table 3 for the first ablation study and in Table 4 for the second, confirming that the performance improves with more references and when the attention mapping is executed on the LR input vector.

	Config.	CU4REF	Sun80	GEMAP	HUMAP
Visual-Oriented	No attention	27.05/.8026	29.72/.7997	35.02/.9003	45.25/.9746
	Ref. attention	26.78/.7909	29.84/.8022	35.11/.9003	45.28/.9753
	Both attention	26.52/.7795	29.73/.7982	35.16/.9015	45.35/.9754
	AMRSR	**27.49/.8145**	**30.41/.8257**	**35.80/.9122**	**45.56/.9771**

Table 4: Visual-oriented quantitative results obtained by dividing into subvectors the feature vectors of references (ref), of both references and input (both) or none (no).

Algorithms	CU4REF	Sun80	GEMAP	HUMAP
AMRSR 1ref	20.03	113.36	84.28	44.05
AMRSR 2ref	41.78	204.38	104.85	84.24
AMRSR 8ref	–	–	418.82	316.04
$N_R = 1$	46.73	80.62	126.91	147.26
$N_R = 4$	137.41	154.03	265.73	161.27
$N_R = 16$	270.76	391.63	265.14	159.63
cut 1ref	–	–	21.95	19.08
cut 4ref	–	–	111.65	98.95
SRNTT [16]	20.87	26.93	(19.37)	(15.37)
TTSR [12]	17.03	17.60	(17.13)	(10.31)
MASA [5]	**5.36**	**5.41**	**(9.62)**	**(7.66)**

Table 5: Processing time (in seconds) during inference for the algorithms of Table 7 of the main paper.

Part-based mechanism and GPU memory usage: in the main paper, we proved that our approach required less GPU memory during inference compared to other RefSR approaches without deteriorating its performances. We analysed the results when references are divided into parts. Table 6 shows the PSNR and SSIM values obtained when also the LR input is divided into parts ($N_I = 1, 4, 16$). Table 7 shows the GPU memory consumption and the processing time for the same configurations. The GPU memory usage depends on the size of the LR input and of the references. The size of the LR inputs for the different datasets are: (125x125) for CU4REF, (256x232) for Sun80, (64x64) for HUMAP and (64x64) for GEMAP. If the LR input is divided into parts, the part-based mechanism significantly reduces the consumption of GPU when it is applied to inputs of higher size. However, it slightly worsens the performances and increases the processing time. Table 5 shows the inference time (in seconds) for the algorithms presented in Table 7 of the main paper. AMRSR is generally slower than the other approaches: the runtime has a sub-linear increase with increasing number and size of reference images. The time of AMRSR is similar to the other RefSR methods if a single reference is used. This is the main limitation of our work and it will be addressed in future work. We did not evaluate Cross-Net [17] and SSEN [8] because the reference images are resized to be equal to the LR input in their implementation.

Configurations	*Visual-Oriented*				*PSNR-Oriented*			
	CU4REF	Sun80	GEMAP	HUMAP	CU4REF	Sun80	GEMAP	HUMAP
$N_I = 1, N_R = 1$	**27.49/.8145**	30.14/.8170	35.58/.9077	45.55/.9771	**28.32/.8394**	30.84/.8393	36.60/.9221	46.81/.9811
$N_I = 1, N_R = 4$	27.17/.8073	30.30/.8217	35.63/.9106	45.45/.9762	28.00/.8358	30.87/.8415	36.65/.9232	46.85/.9814
$N_I = 1, N_R = 16$	27.08/.8040	**30.42/.8263**	**35.80/.9122**	**45.56/.9771**	27.97/.8341	**30.95/.8438**	**36.82/.9248**	**46.86/.9814**
$N_I = 4, N_R = 1$	26.44/.7793	29.86/.8082	35.49/.9059	45.39/.9761	27.28/.8108	30.62/.8329	36.57/.9219	46.81/.9812
$N_I = 4, N_R = 4$	26.43/.7776	30.04/.8131	35.50/.9071	45.34/.9754	27.22/.8096	30.61/.8332	36.59/.9225	46.82/.9812
$N_I = 4, N_R = 16$	26.38/.7749	30.17/.8167	35.65/.9111	45.38/.9762	27.16/.8068	30.69/.8360	36.69/.9236	46.80/.9811
$N_I = 16, N_R = 1$	25.58/.7405	29.76/.8024	35.42/.9055	45.40/.9760	26.45/.7809	30.51/.8292	36.53/.9215	46.81/.9810
$N_I = 16, N_R = 4$	25.63/.7441	29.89/.8075	35.46/.9061	45.24/.9753	26.40/.7830	30.48/.8284	36.54/.9217	46.74/.9808
$N_I = 16, N_R = 16$	25.62/.7415	29.90/.8082	35.57/.9101	45.33/.9760	26.36/.7789	30.47/.8299	36.60/.9229	46.74/.9808

Table 6: PSNR/SSIM values of different configurations of AMRSR. N_I is the number of parts which the LR input is divided into while N_R is the number of parts which the reference images are divided into. $N_M = 4$ reference images used.

Configurations	CU4REF		Sun80		GEMAP		HUMAP	
	GPU	Time	GPU	Time	GPU	Time	GPU	Time
$N_I = 1, N_R = 1$	1.36	**46.73**	3.96	**80.62**	40.15	**126.91**	29.60	**147.26**
$N_I = 1, N_R = 4$	1.21	137.41	3.23	154.03	28.49	265.73	20.98	161.27
$N_I = 1, N_R = 16$	1.22	270.76	3.24	391.63	15.69	265.14	11.59	159.63
$N_I = 4, N_R = 1$	1.00	65.74	3.33	112.81	39.54	240.00	26.93	275.27
$N_I = 4, N_R = 4$	0.80	77.46	2.06	155.39	28.18	545.41	20.73	310.19
$N_I = 4, N_R = 16$	0.80	168.98	2.08	372.73	15.53	483.30	11.44	430.83
$N_I = 16, N_R = 1$	0.90	56.19	3.17	129.62	39.29	252.81	26.77	380.07
$N_I = 16, N_R = 4$	**0.72**	76.85	**1.79**	204.95	28.06	1072.46	20.63	635.21
$N_I = 16, N_R = 16$	0.73	168.70	1.80	470.91	**15.47**	968.72	**11.39**	716.55

Table 7: GPU memory usage (GB) and processing time (in seconds) of different configurations of AMRSR. N_I is the number of parts which the LR input is divided into while N_R is the number of parts which the reference images are divided into. $N_M = 4$ reference images used.

4. Visual comparisons with other approaches

We conduct a qualitative evaluation of the SR outputs of the methods cited in Section 5 of the main paper, namely: the PSNR-oriented networks EDSR [4], MDSR [4], RRDB-Net [11], SRResNet [2], RCAN [15], NHR [3], NLR [3], CSNLN [6], MAFFSRN [7]; the visual-oriented GANs SRGAN [2], ESRGAN [11], RSRGAN [14] and the RefSR approaches CrossNet [17], SSEN [12], SRNTT [16], TTSR [12] and MASA [5]. More specifically, Figures 1, 2, 3, 4 shows examples from CU4REF dataset, Figures 5, 6, 7 from Sun80, Figures 8, 9 illustrate SR texture map from GEMAP while Figures 10, 11 from HUMAP. The presented examples prove the superiority of AMRSR among other approaches. Its SR outputs are less blurry than the outputs of PSNR-oriented methods and they have richer and sharper details than the ones of the visual-oriented and RefSR approaches.

5. Further comparisons with CIMR [1]

In the main paper, we evaluate our approach with the same training and test settings adopted by CIMR [1], the other multi-reference super-resolution network, in the case that multiple references are exploited. CIMR was further tested on content-similar references using a single reference image with 4 different levels of similarity to the LR input. We evaluate AMRSR with this setting by using a single reference taken from four similarity level groups (from L1 to L4) of CUFED5 [16] dataset. The quantitative results presented in Table 8 show that AMRSR outperforms CIMR also when only a single reference is used, for all the considered levels of similarity.

Table 9 compared different configurations of the part-based mechanism when AMRSR is trained and tested with the same datasets as CIMR [1] in the case of multiple references. Specifically, N_M references, taken from the Outdoor Scene (OST) dataset [10], are randomly associated to each LR input image. The maximum size of the reference images is (984x736). The highest figures of PSNR and SSIM are obtained when the reference images are divided into 16 parts, confirming that the part-based mechanism improves the performances if high resolution references are exploited. Figure 12 illustrates visual examples of SRNTT [16], CIMR and AMRSR applied to Sun80 [9] dataset in the case of multiple references. SRNTT and CIMR images are taken from [1]. The higher quality of AMRSR outputs proves its qualitative superiority.

Algorithms	L1	L2	L3	L4
CIMR-l_2	27.32/.805	27.05/.799	26.92/.796	26.86/.794
AMRSR-l_2	**28.38/.844**	**27.87/.825**	**27.76/.821**	**27.59/.817**
CIMR [1]	26.50/.786	26.47/.784	26.45/.784	26.44/.784
AMRSR	27.32/.808	27.01/.795	26.95/.793	26.85/.791

Table 8: Quantitative comparison between AMRSR and CIMR exploiting single reference images with different levels of similarity to the LR inputs. The results of CIMR are taken from [1].

Config.	N_M	*Visual Oriented*		*PSNR Oriented*	
		CUFED5	Sun80	CUFED5	Sun80
$N_I = 1$ $N_R = 1$	4	26.86/.7936	30.49/.8255	27.48/.7974	30.98/.8420
	8	26.90/.7950	30.56/.8273	27.55/.7992	31.06/.8443
	16	26.95/.7950	30.60/.8288	27.62/.7994	31.11/.8456
	32	26.97/.7959	30.66/.8303	27.68/.8208	31.17/.8470
	64	27.05/.7979	30.72/.8321	27.71/.8221	31.22/.8483
$N_I = 1$ $N_R = 4$	4	26.84/.7941	30.52/.8281	27.54/.8192	31.04/.8450
	8	26.92/.7956	30.63/.8311	27.60/.8200	31.18/.8489
	16	26.95/.7971	30.67/.8326	27.65/.8222	31.24/.8514
	32	26.99/.7982	30.74/.8341	27.68/.8237	31.29/.8513
	64	27.07/.7999	30.80/.8356	27.70/.8230	31.30/.8508
$N_I = 1$ $N_R = 16$	4	26.87/.7957	30.53/.8293	27.57/.8207	31.10/.8477
	8	26.92/.7963	30.64/.8326	27.63/.8216	31.24/.8516
	16	26.98/.7976	30.69/.8340	27.69/.8238	31.29/.8527
	32	27.01/.7988	30.75/.8354	27.75/.8248	31.35/.8540
	64	**27.08/.8004**	**30.80/.8369**	**27.81/.8261**	**31.41/.8548**

Table 9: Part-based mechanism results when AMRSR is trained and evaluated with the same datasets as CIMR. N_M is the number of reference images, N_I is the number of parts which the LR input is divided into while N_R is the number of parts which the reference images are divided into.

References

[1] Shuguang Cui. Towards content-independent multi-reference super-resolution: Adaptive pattern matching and feature aggregation. 2020. 1, 3, 16

[2] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017. 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

[3] Yawei Li, Vagia Tsiminaki, Radu Timofte, Marc Pollefeys, and Luc Van Gool. 3d appearance super-resolution with deep learning. In *In Proceedings of the IEEE International Conference on Computer Vision*, 2019. 3, 12, 13, 14, 15

[4] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017. 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

[5] Liying Lu, Wenbo Li, Xin Tao, Jiangbo Lu, and Jiaya Jia. Masa-sr: Matching acceleration and spatial adaptation for reference-based image super-resolution. *arXiv preprint arXiv:2106.02299*, 2021. 2, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

[6] Yiqun Mei, Yuchen Fan, Yuqian Zhou, Lichao Huang, Thomas S Huang, and Honghui Shi. Image super-resolution with cross-scale non-local attention and exhaustive self-exemplars mining. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5690–5699, 2020. 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

[7] Abdul Muqeet, Jiwon Hwang, Subin Yang, Jung Heum Kang, Yongwoo Kim, and Sung-Ho Bae. Ultra lightweight image super-resolution with multi-attention layers. *arXiv preprint arXiv:2008.12912*, 2020. 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

[8] Gyumin Shim, Jinsun Park, and In So Kweon. Robust reference-based super-resolution with similarity-aware deformable convolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8425–8434, 2020. 2, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

[9] Libin Sun and James Hays. Super-resolution from internet-scale scene matching. In *2012 IEEE International Conference on Computational Photography (ICCP)*, pages 1–12. IEEE, 2012. 3

[10] Xintao Wang, Ke Yu, Chao Dong, and Chen Change Loy. Recovering realistic texture in image super-resolution by deep spatial feature transform. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 606–615, 2018. 3

[11] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018. 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

[12] Fuzhi Yang, Huan Yang, Jianlong Fu, Hongtao Lu, and Baining Guo. Learning texture transformer network for image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5791–5800, 2020. 2, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

[13] Huanjing Yue, Xiaoyan Sun, Jingyu Yang, and Feng Wu. Landmark image super-resolution by retrieving web images. *IEEE Transactions on Image Processing*, 22(12):4865–4878, 2013. 1

[14] Wenlong Zhang, Yihao Liu, Chao Dong, and Yu Qiao. Ranksrgan: Generative adversarial networks with ranker for image super-resolution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3096–3105, 2019. 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

[15] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 286–301, 2018. 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

[16] Zhifei Zhang, Zhaowen Wang, Zhe Lin, and Hairong Qi. Image super-resolution by neural texture transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7982–7991, 2019. 1, 2, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16

[17] Haitian Zheng, Mengqi Ji, Haoqian Wang, Yebin Liu, and Lu Fang. Crossnet: An end-to-end reference-based super resolution network using cross-scale warping. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 88–104, 2018. 2, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

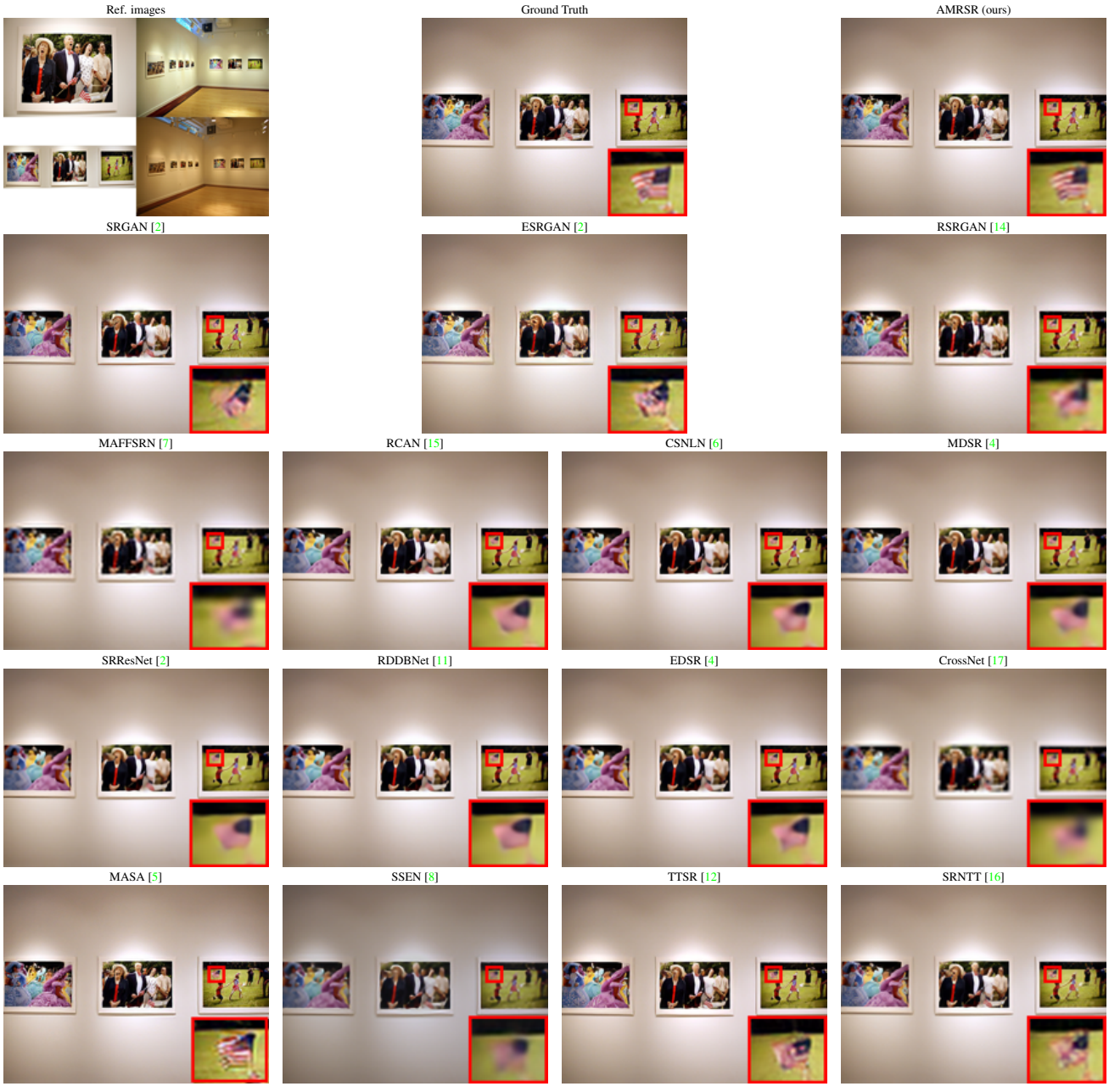

Figure 1: Examples of SR outputs of different approaches from CU4REF dataset.

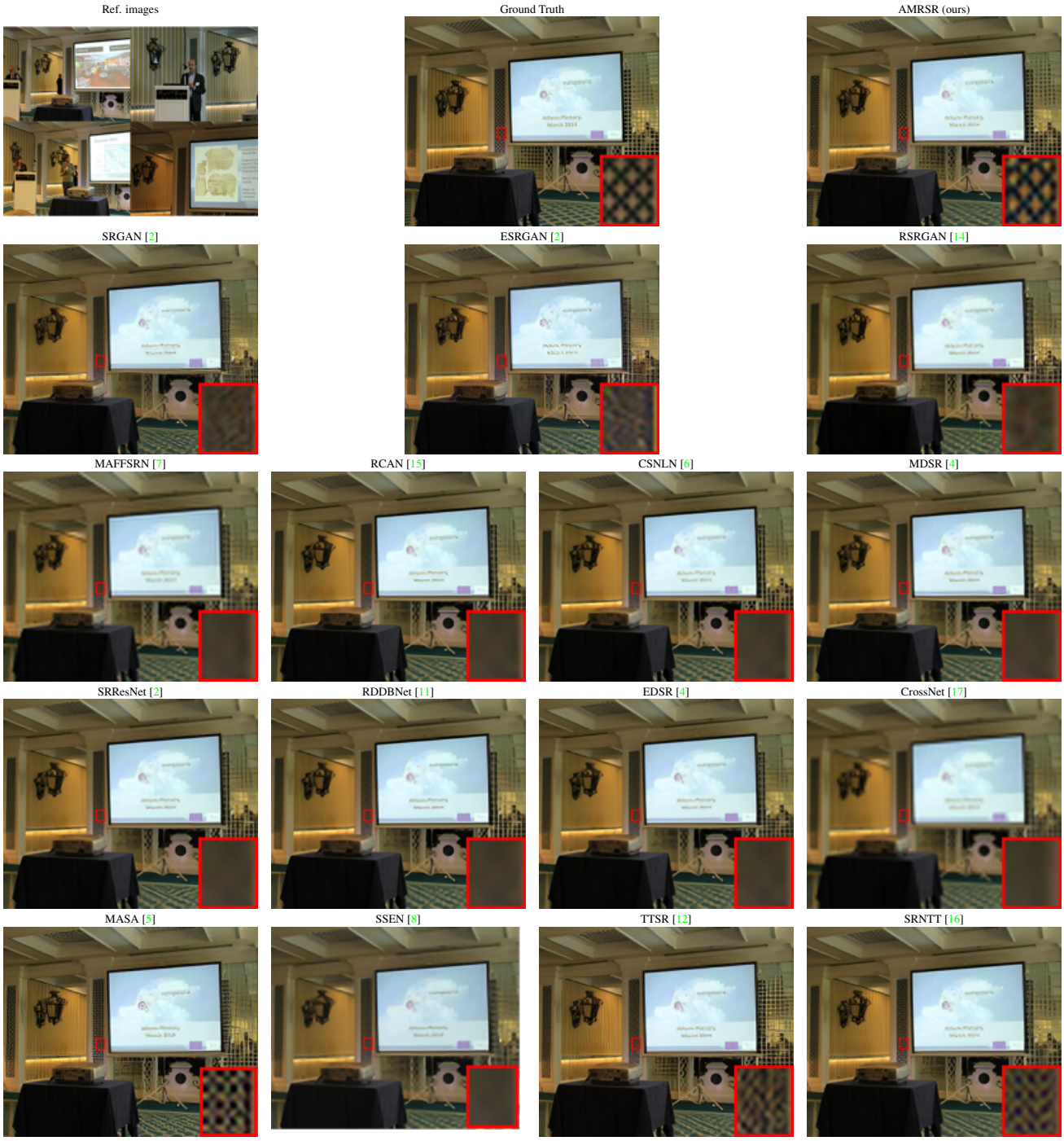

Figure 2: Examples of SR outputs of different approaches from CU4REF dataset.

Figure 3: Examples of SR outputs of different approaches from CU4REF dataset.

Figure 4: Examples of SR outputs of different approaches from CU4REF dataset.

Figure 5: Examples of SR outputs of different approaches from Sun80 dataset.

Figure 6: Examples of SR outputs of different approaches from Sun80 dataset.

Figure 7: Examples of SR outputs of different approaches from Sun80 dataset.

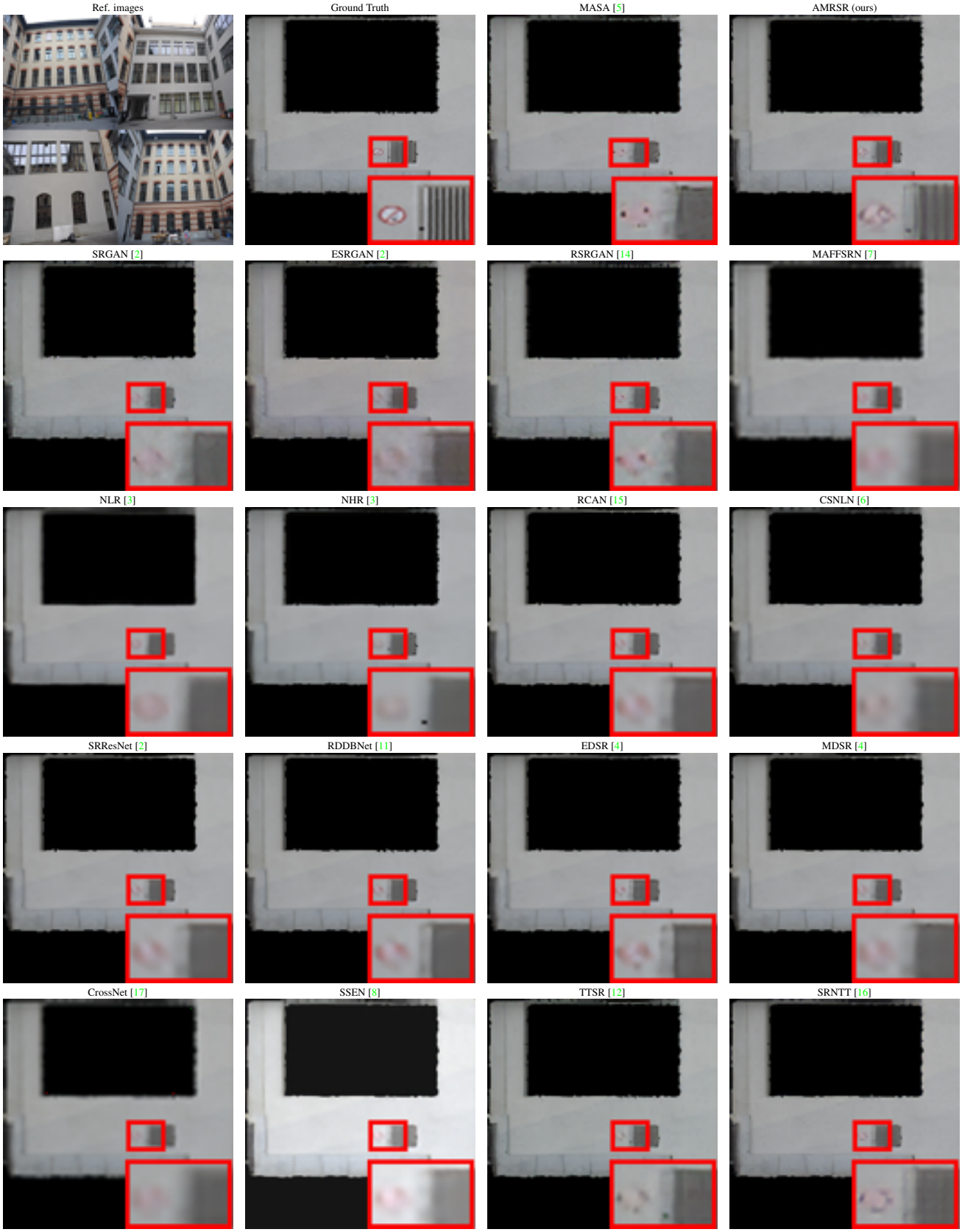

Figure 8: Examples of SR outputs of different approaches from GEMAP dataset.

Figure 9: Examples of SR outputs of different approaches from GEMAP dataset.

Figure 10: Examples of SR outputs of different approaches from HUMAP dataset.

Figure 11: Examples of SR outputs of different approaches from HUMAP dataset.

Figure 12: Visual comparison between state-of-the-art single reference method (SRNTT) and the other multiple reference approach (CIMR). The figures of these two are taken from [1].