Supplementary Material: FeatureFlow: Robust Video Interpolation via Structure-to-texture Generation

Shurui Gui*[1,2], Chaoyue Wang*[1], Qihua Chen[1,3], and Dacheng Tao[1]

[1]UBTECH Sydney AI Centre, School of Computer Science, Faculty of Engineering, The University of Sydney, Darlington, NSW 2008, Australia
[2]School of Computer Science, University of Science and Technology of China, China
[3]School of Information and Technology, University of Science and Technology of China, China
agnesgsr@mail.ustc.edu.cn, chaoyue.wang@sydney.edu.au, cqh@mail.ustc.edu.cn, dacheng.tao@sydney.edu.au

In this supplementary material, we first introduce the network architecture details for reproduction and better understanding. Then we would like to discuss the videos contained in the package, for demonstrating the results on real videos. Finally, we will show more visual comparisons with our method's variants (*i.e.* ablation studies) and state-of-the-art algorithms. The source code and more video examples are available on https://github.com/CM-BF/FeatureFlow.

1. Network Architectures

1.1. Feature extractor

We adopt modified Resnet50 [3] as our *feature extractor* where the network structure is shown as Table 1.

	Input	Output	Kernel_size	Ic	Oc	Stride	padding	Norm_layer	activation	Output_size
in	-	rgbs	-	-	4	-	-	-	-	$H \times W$
	rgbs	mid	7×7	4	64	2	3	BatchNorm	ReLu	$H/2 \times W/2$
	mid	ft0	3×3	-	-	2	1	-	-	$H/4 \times W/4$
resnet50.layer1 [3]	ft0	ft1	-	64	256	-	-	-	-	$H/4 \times W/4$
resnet50.layer2 [3]	ft1	ft2	-	256	512	-	-	-	-	$H/8 \times W/8$

Table 1: Architecture of *feature extractor*. Ic: input channels, Oc: output channels. Please refer [3] for resnet50.layer1 and resnet50.layer2.

1.2. Multi-Flow Predictor (MFP)

The input of MFP is the concatenating results of two frames features, so the number of input channels is 1024. The network can be described as Table 2 where RB is Residual Block.

Then we can obtain the result flows:

$$Flow_{0\to t}, Flow_{1\to t} = \mathcal{C}_1(out), \tag{1}$$

where $\mathcal{C}_1$ is the inverse operation of concatenation along dimension 1 (depth dimension).

* indicates equal contribution

	Input	Output	Kernel_size	Ic	Oc	Stride	padding	Norm_layer	activation	Output_size
	in	mid	3×3	1024	512	1	1	BatchNorm	LeakyReLu	$H/8 \times W/8$
5 RBs with BN	mid	mid2	-	512	512	-	-	BatchNorm	LeakyReLu	$H/8 \times W/8$
3 RBs w/o BN	mid2	mid3	-	512	512	-	-	-	LeakyReLu	$H/8 \times W/8$
Conv2d	mid3	out	3×3	512	576	1	1	-	-	$H/8 \times W/8$

Table 2: Architecture of MFP. Note that the number of output channels of Conv2d can be calculated as $2 \times 2 \times 16$ (semantic groups) $\times 3^2$ (deformable kernel size) $= 576$.

1.3. Multi-Attention Predictor (MAP)

We can calculate the attention maps as following equation:

$$A_0, A_1 = \mathcal{C}_1(\mathcal{S}(\textbf{mix\_map}(\text{Cat}_1(\textbf{Emb}(in_0), \textbf{Emb}(in_1))))), \tag{2}$$

where $\mathcal{S}$ is *sigmoid* function, and Cat_1 is concatenation along depth dimension. **Emb** is a convolution network: kernel size $= 3 \times 3$, the number of input/output channels $= 512$; padding $= 1$. **mix_map** is shown in Table 3

	Input	Output	Kernel_size	Ic	Oc	Stride	padding	Norm_layer	activation	Output_size
Conv2d1	in	mid	3×3	1024	512	1	1	-	LeakyReLu	$H/8 \times W/8$
3 RBs w/o BN	mid	mid2	-	512	512	-	-	-	LeakyReLu	$H/8 \times W/8$
Conv2d2	mid2	out	3×3	512	32	1	1	-	-	$H/8 \times W/8$

Table 3: Architecture of **mix_map**. Note that the number of output channels of Conv2d2 can be calculated as 2×16 (semantic groups) $= 32$.

1.4. Global Generator

We adopt 9 RBs, 3 transposed convolutions and one more convolution as the Global Generator which is shown in Table 4.

	Input	Output	Kernel_size	Ic	Oc	Stride	padding	Norm_layer	activation	Output_size
9 RBs w/o BN	in	mid	-	512	512	-	-	-	ReLu	$H/8 \times W/8$
ConvTranspose2d1	mid	mid2	3×3	512	256	2	1	-	ReLu	$H/4 \times W/4$
ConvTranspose2d2	mid2	mid3	3×3	256	128	2	1	-	ReLu	$H/2 \times W/2$
ConvTranspose2d3	mid3	mid4	3×3	128	64	2	1	-	ReLu	$H \times W$
Conv2d1	mid4	out	7×7	64	4	1	3	-	Tanh	$H \times W$

Table 4: Architecture of Global Generator.

1.5. Frame Texture Compensator (FTC)

Please refer to EDVR architecture [5] for the basic structure. In FTC, we set different hyperparameters: 5 RBs in Shallow Feature Extractor, 8 deformable groups for PCD module with Pre-Deblur module, 3 frames for TSA module, and 20 RBs for Texture Generator. Texture Generator is modified from the Reconstruction module where after RBs, we delete the upsampling and pixel shuffle operations, instead, we plus the residual to the reference frames directly after 2 convolutions with 1 ReLu activation function.

2. Real Videos Exhibition

We provide videos on https://github.com/CM-BF/FeatureFlow.

3. Visual Comparisons

3.1. Structure-to-Texture

We show two groups of $\tilde{I}_t^{s_1}$ and $\tilde{I}_t^{s_2}$ in Figure 1 to explain the results of the two stages.

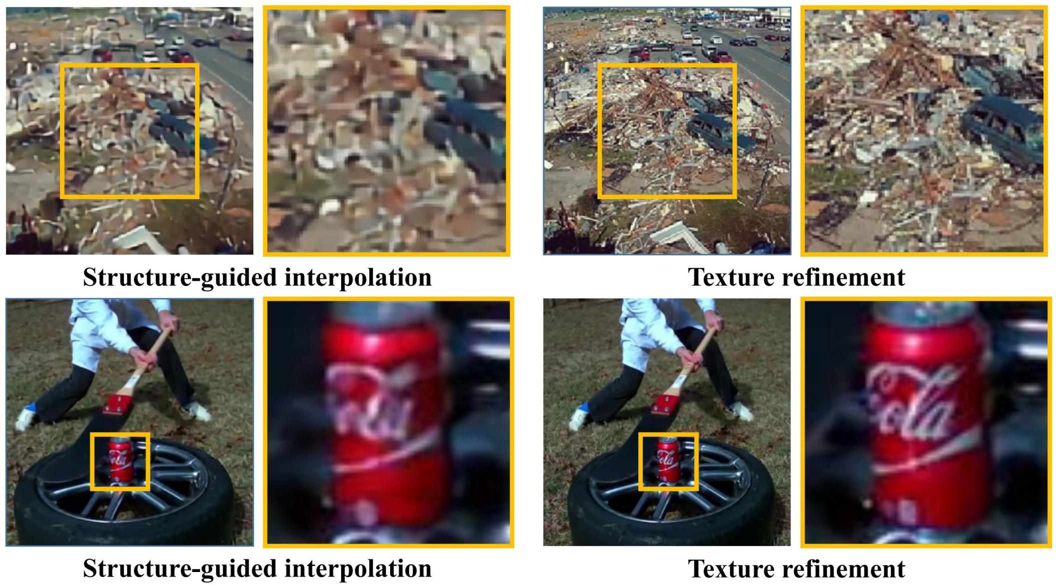
Structure-guided interpolation
Texture refinement
Structure-guided interpolation
Texture refinement

Figure 1: **Visualization of two stage results.**

3.2. Edge Generation

In Figure 2, the edge images generated by Global Generator protect the edge information from lost and give the stage-I result cutting edges for further matching.

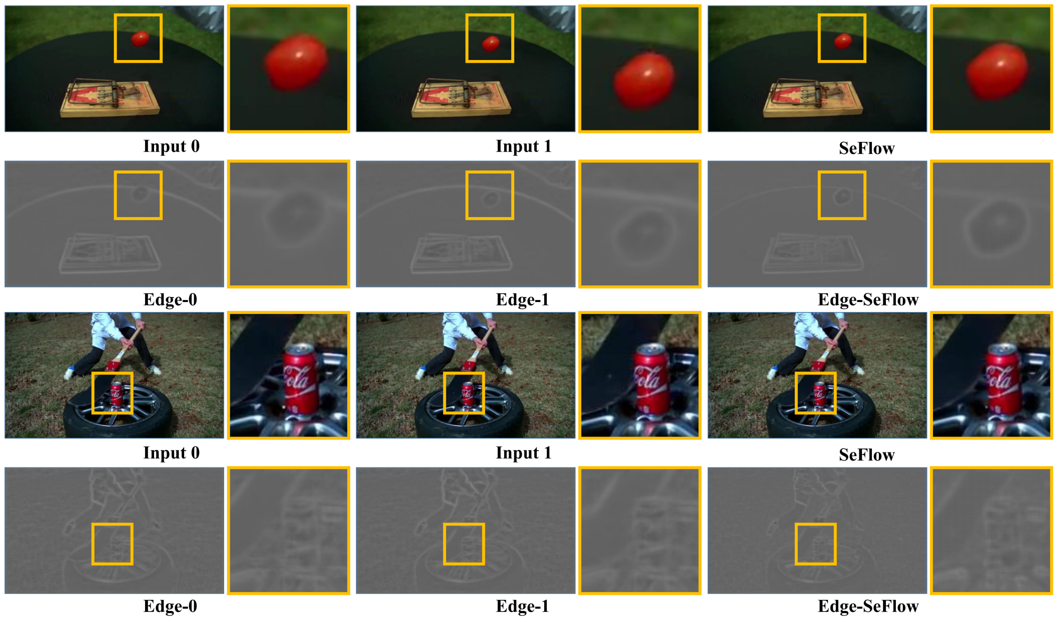

Figure 2: **Visualization of edge images generation**.

3.3. Ablation Study

In Figure 3, we show a visual comparison between SeFlow-None and SeFlow-Full where the lack of edge information causes blur edge in SeFlow-None-ref (the stage-I result) which is hard to compensate in the stage-II.

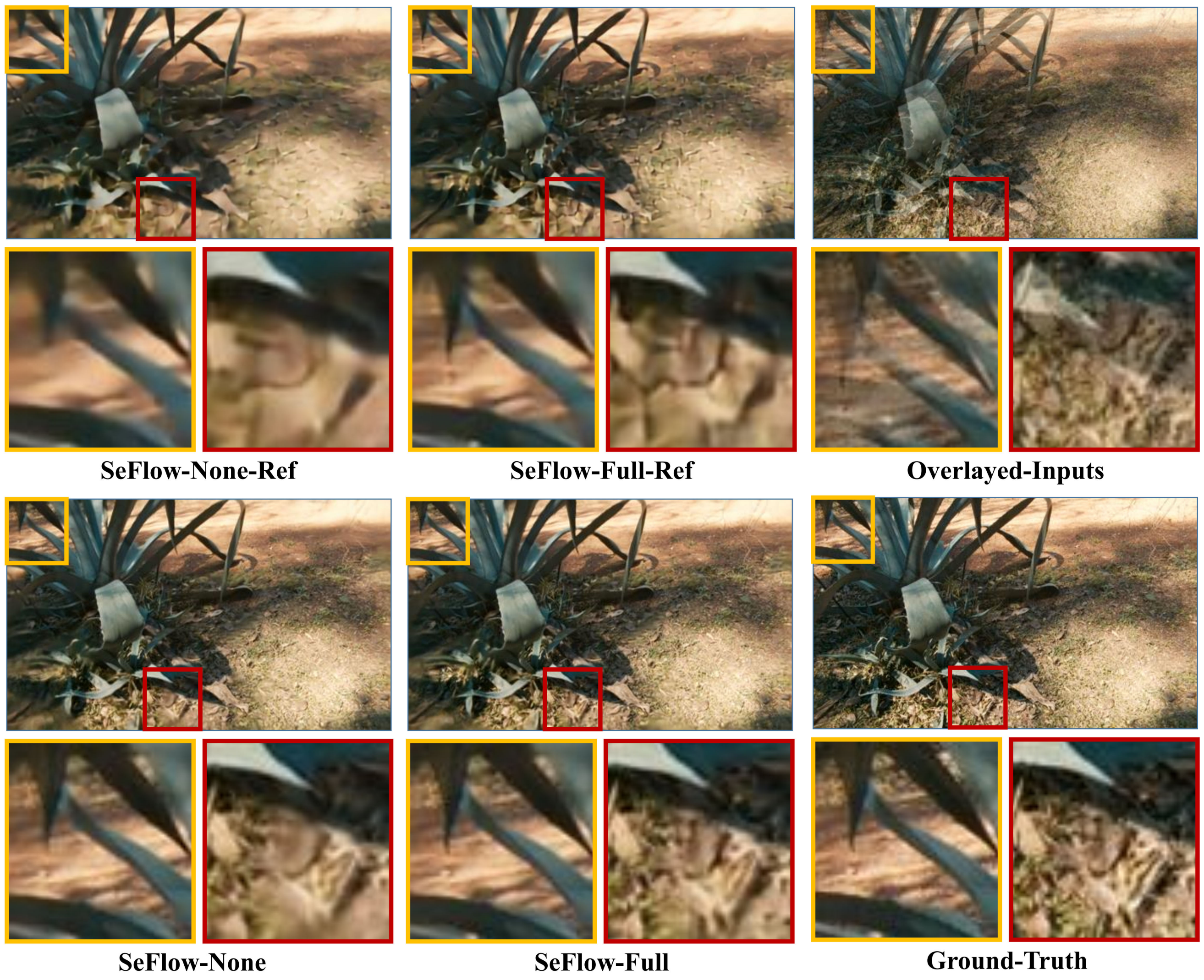

Figure 3: **The comparison between SeFlow-None and SeFlow-Full (denoted as SeFlow).** This figure includes their two-stage results (*-Ref is the stage-I result).

3.4. Comparisons with State-of-the-arts

We further display more visual compared examples on Vimeo90K test set to show our high-quality performance against all the compared algorithms [4, 2, 1].

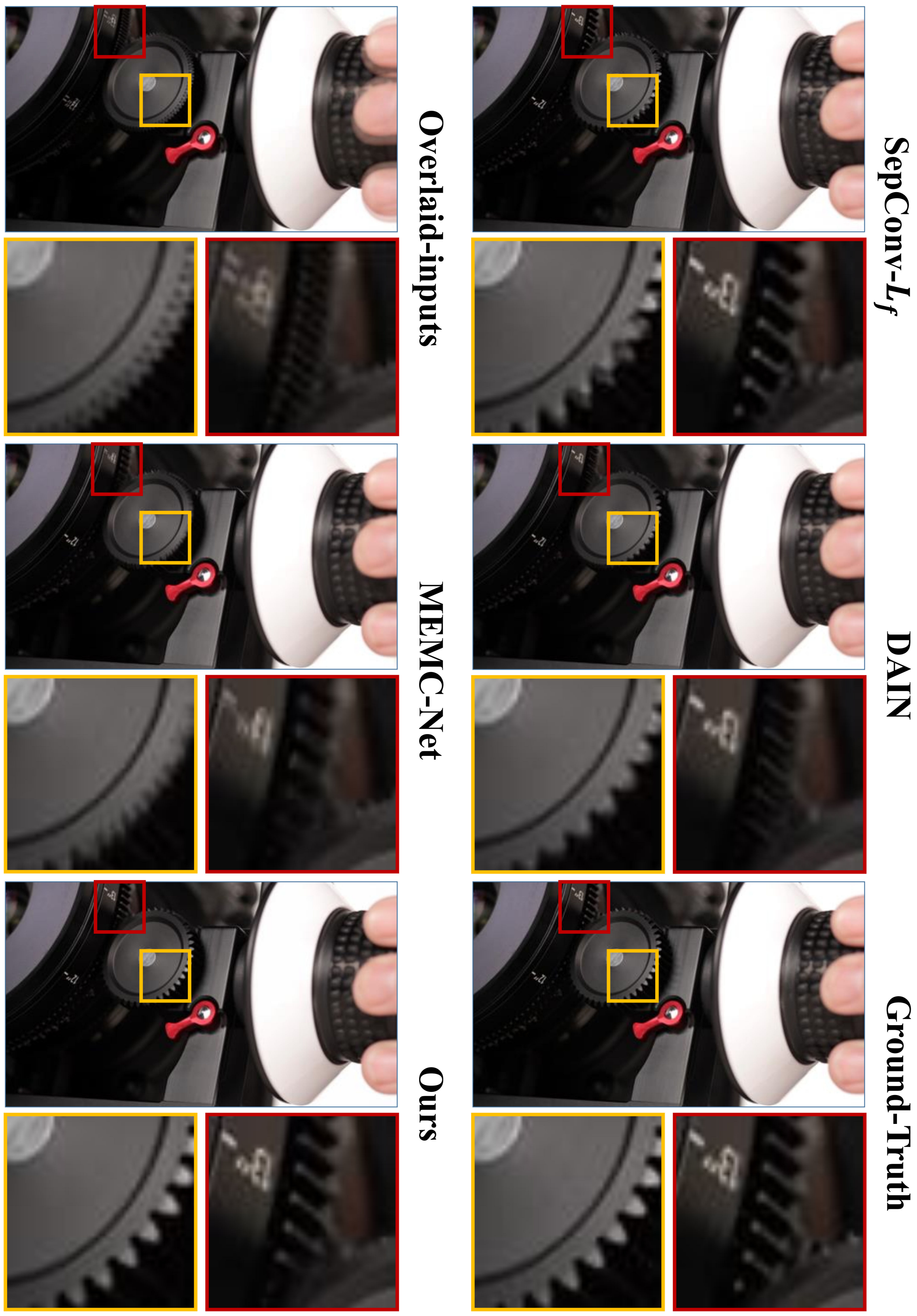

Figure 4: Our method generate high quality results in rotated motion cases.

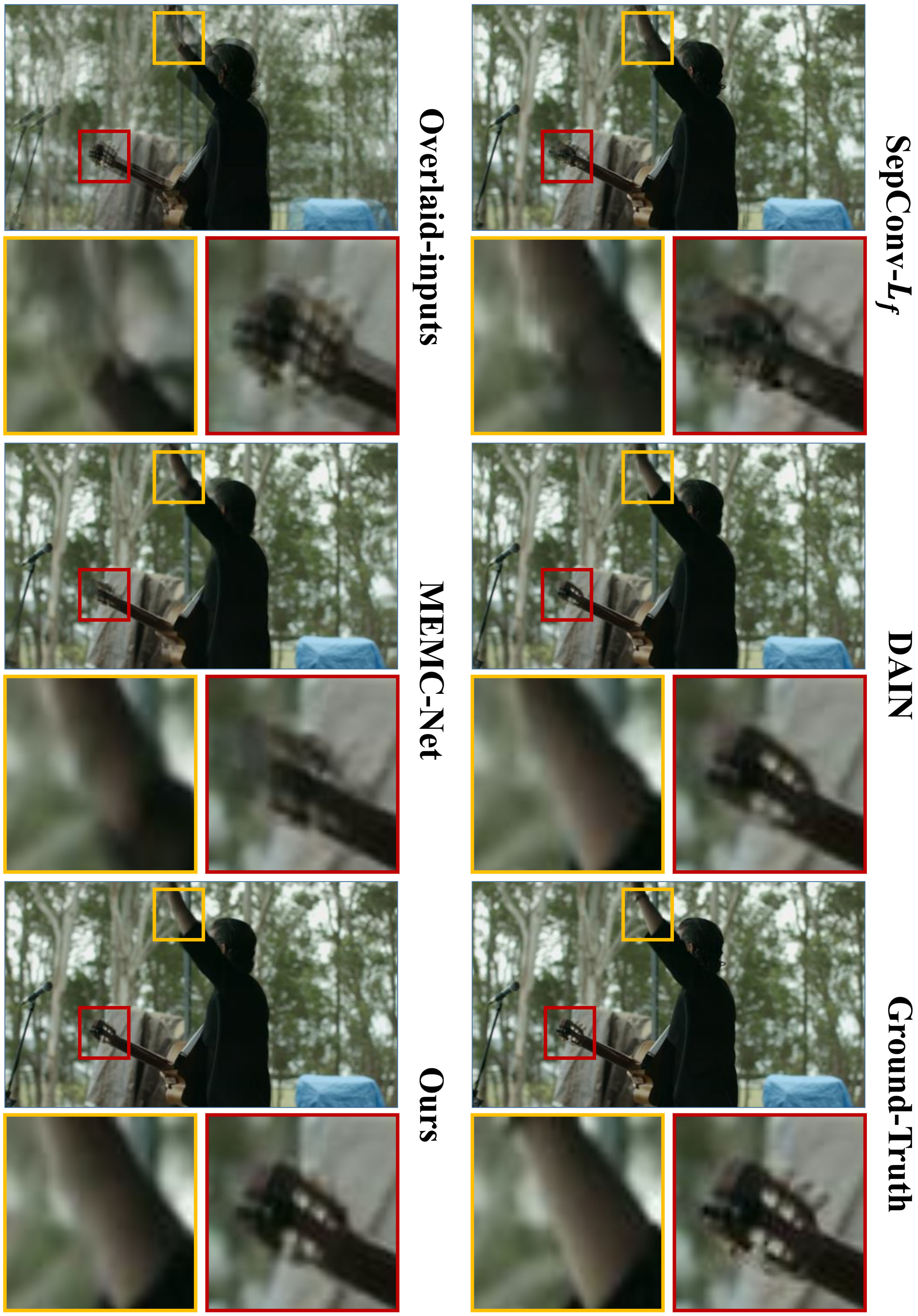

Figure 5: SeFlow is mastered at producing accurate edges and high-quality textures.

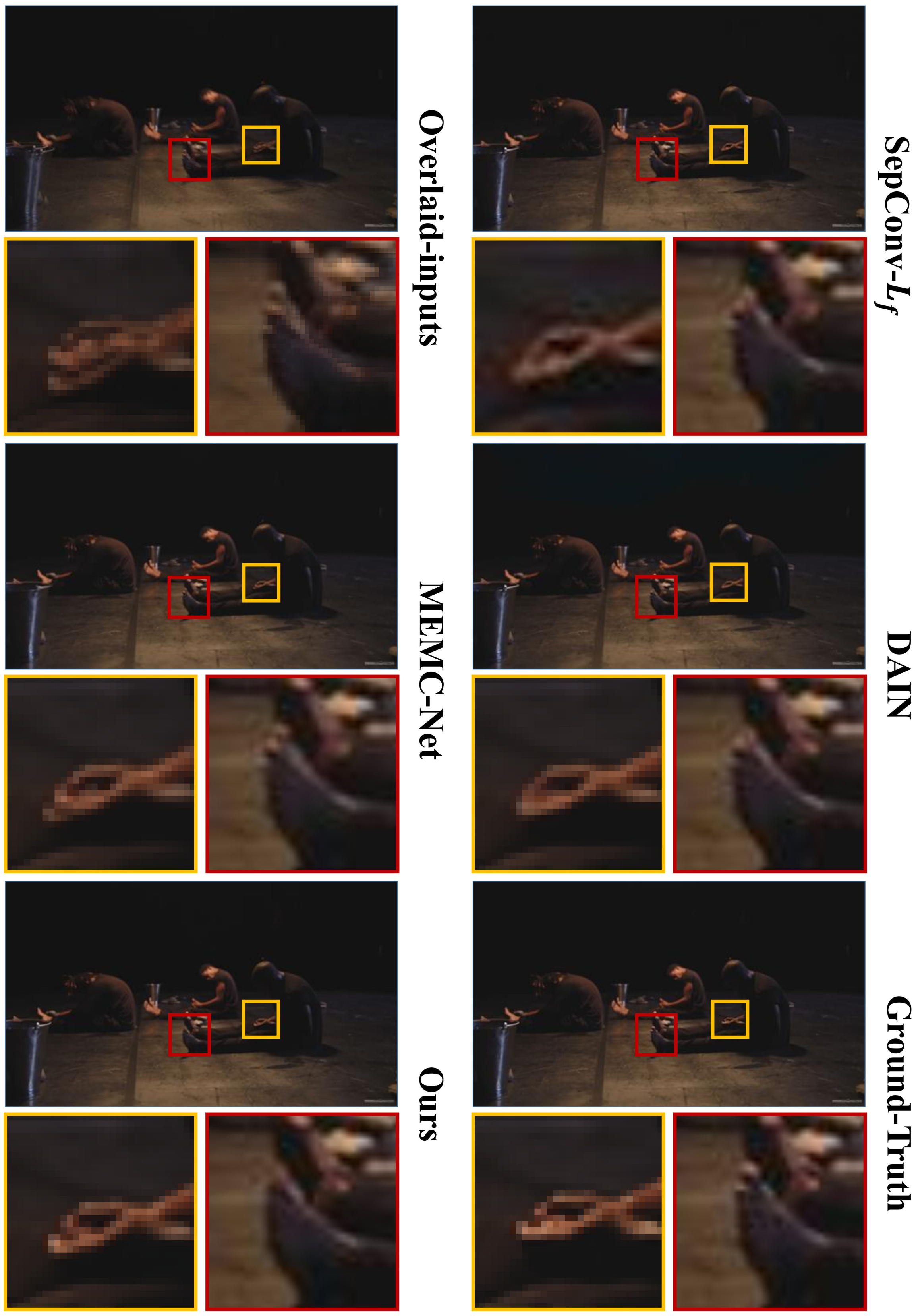

Figure 6: Note that all of the other methods generate one more toe.

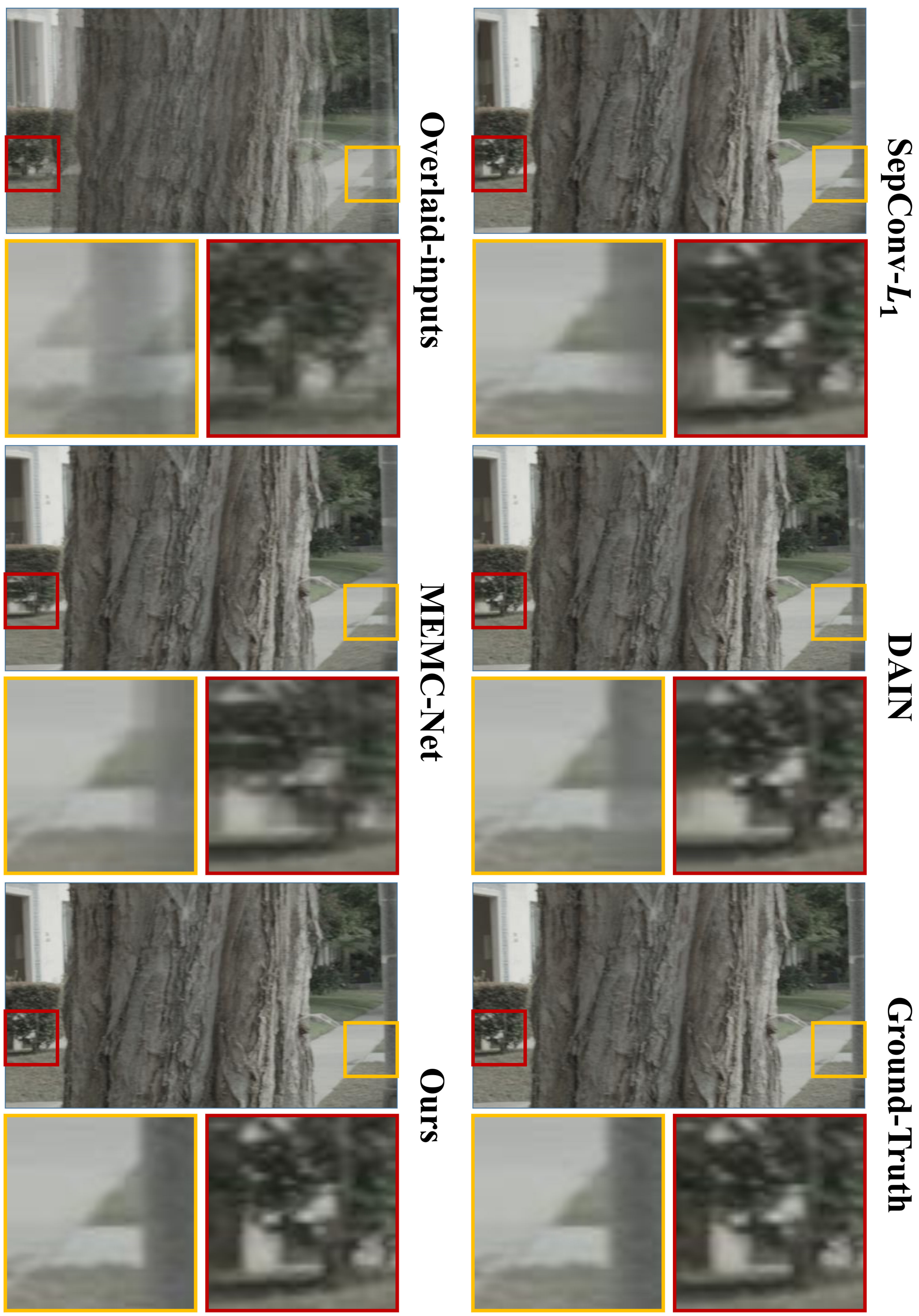

Figure 7: Our algorithm shows its effectiveness in tackling boundary occlusion.

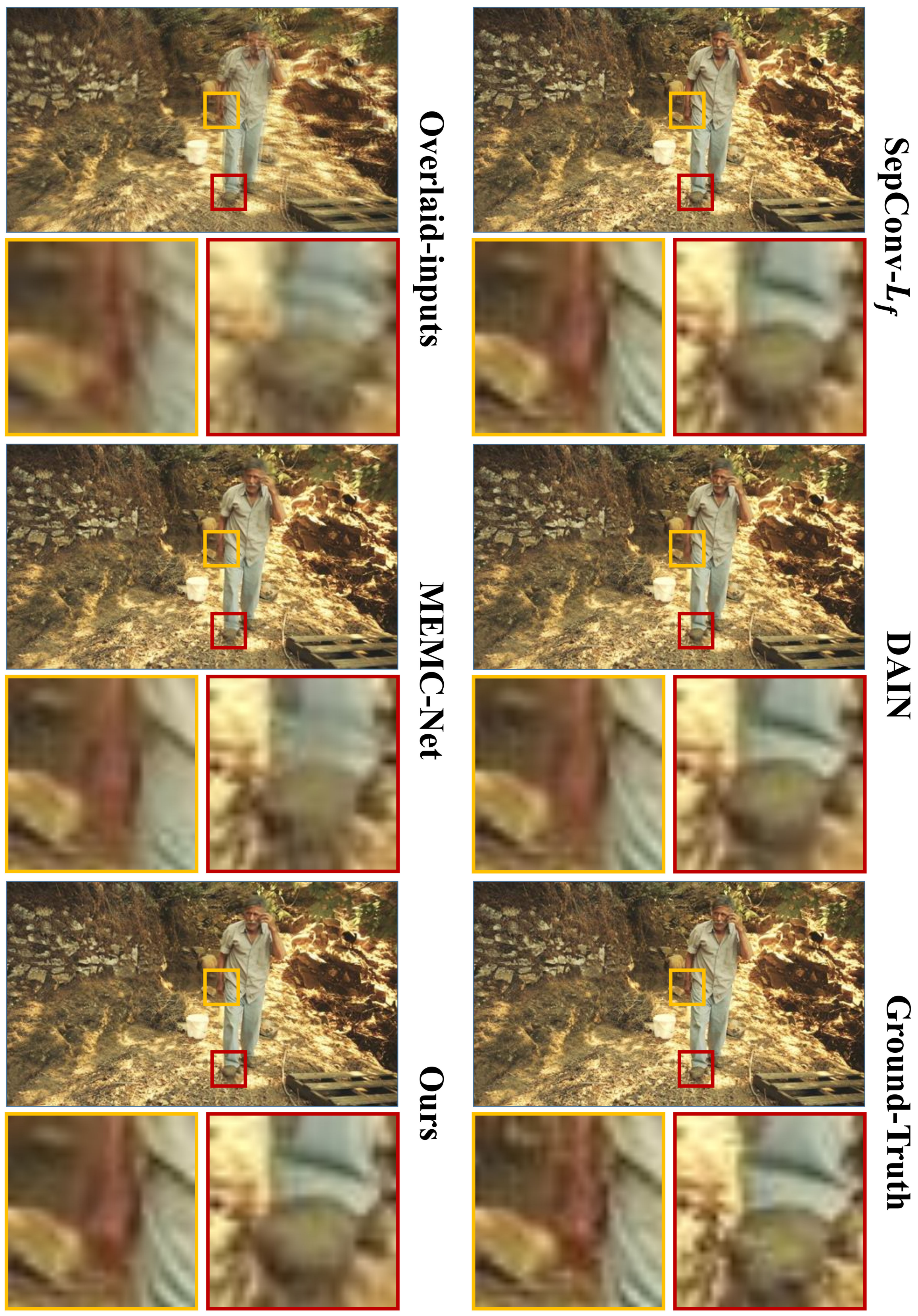

Figure 8: Our results contain less blurriness than other algorithms.

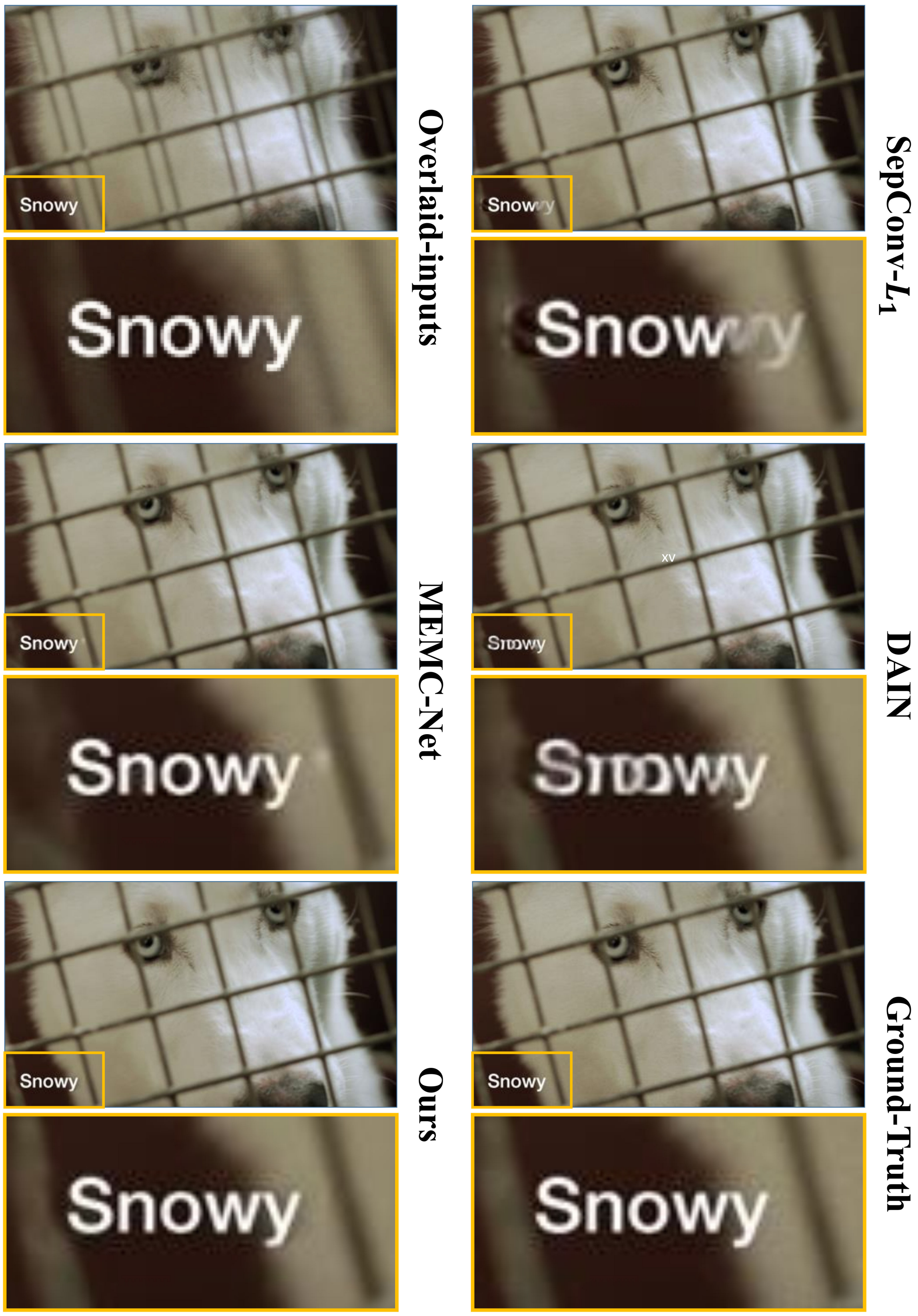

Figure 9: Our method's "Snowy" does not get distorted by background motion.

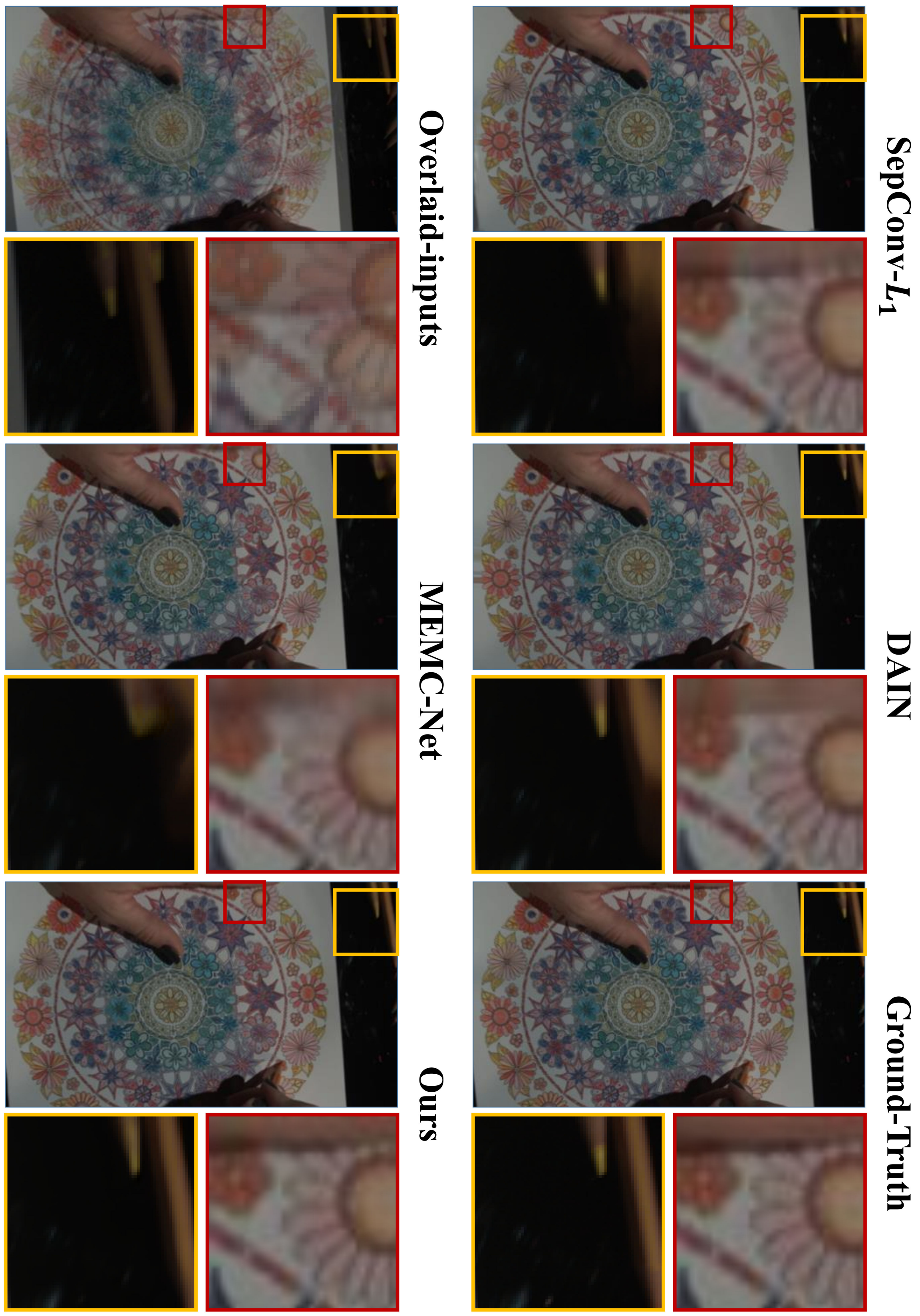

Figure 10: Our method shows its advantage on boundary prediction.

References

[1] Wenbo Bao, Wei-Sheng Lai, Chao Ma, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. Depth-aware video frame interpolation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[2] Wenbo Bao, Wei-Sheng Lai, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. Memc-net: Motion estimation and motion compensation driven neural network for video interpolation and enhancement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[4] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive separable convolution. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[5] Xintao Wang, Kelvin C.K. Chan, Ke Yu, Chao Dong, and Chen Change Loy. Edvr: Video restoration with enhanced deformable convolutional networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.