Supplementary Materials for Free-form Video Inpainting with 3D Gated Convolution and Temporal PatchGAN

Ya-Liang Chang* Zhe Yu Liu* Kuan-Ying Lee Winston Hsu

National Taiwan University, Taipei, Taiwan

{yaliangchang, zhe2325138}@cmlab.csie.ntu.edu.tw, {r03922165, whsu}@ntu.edu.tw

1. Implementation Details

1.1. Network

We use PyTorch1.0 to implement the network. We use kernel size $(3 \times 5 \times 5)$ for the first convolutional layer, kernel size $(3 \times 4 \times 4)$ with stride 2 for the down-sampling layers, kernel size $(3 \times 3 \times 3)$ with dilation of 2, 4, 8, 16 for dilated layers and kernel size $(3 \times 3 \times 3)$ for other convolutional layers in the generator. For non-linearity layers, we use LeakyReLU. We use nearest up-sampling and convolutions for up-sampling layers.

1.2. Training Procedure

At the first stage in training, we found applying only l_1 loss, perceptual loss and style loss enhances the training speed and stability. After the first stage converges, we add the Temporal PatchGAN to fine-tune. We set $\lambda_{l_1} = 1$, $\lambda_{l_{1mask}} = 6$, $\lambda_{perc} = 1$, $\lambda_{style} = 10$ and $\lambda_G = 1$. We use Adam optimizer [3] with learning rate 0.0001.

The first stage of the training process takes about 1 or 3 days on the FaceForensics/FVI dataset, while fine-tuning takes about 3 or 10 days, respectively with a single Tesla V100 GPU. We use PyTorch data and model parallel to train our model on multiple GPUs to speed up the process.

1.3. Data Pre-processing

See the Dataset section in the main paper.

2. Execution Time

During testing, our model runs at 17 frame per second (FPS), Edge-Connect 55 FPS, CombCN 120 FPS on a single Tesla V100 GPU, and TCCDS 0.05 FPS on a Intel(R) Xeon(R) CPU E5-2650.

3. More Comparison with Other Methods

3.1. Quantitative results

The state-of-the-art patch-based video inpainting method TCCDS by Huang *et al.* [1], image inpainting method Edge-Connect(EC) by Nazeri *et al.* [4] and learning based video inpainting method CombCN by Wang *et al.* [8] are set for comparison.

For results of different ratios on FVI testing set in Table 2 of the main paper, see Table 1a for curve-like mask without border constraint, Table 1b for curve-like masks with border constraint, Table 1c for object-like masks without border constraint, and Table 1d for object-like masks with border constraint.

3.2. Qualitative results

See Fig. 3, 4, 5, and 6. TCCDS [1] is prone to finding wrong patches at complex regions. Edge-Connect [4] produces plausible results for each frame but suffers from temporal inconsistency (see changes in the circled area). CombCN [8] generally generates blurry images in both object-like and curve-like masks.

4. Ablation Study Visualization

See Fig. 7, 8 and 9. Without 3D convolutions (using 2D convolutions), the results are temporally inconsistent (see changes in the red circles); without gated convolutions (using vanilla convolutions), the model tends to mix textures from nearby pixels and generates more artifacts in complex areas; without T-PatchGAN, all masked areas tend to have fish scale artifacts.

5. Extension to Video Super-resolution

5.1. Mask Explanation

See Fig. 1 for the visualization of special masks for video super-resolution and video interpolation.

	Mask Ratio	TCCDS	EC#	CombCN	Ours
MSE↓	0-10%	0.0019	0.0012	**0.0003**	0.0003
	10-20%	0.0055	0.0020	**0.0008**	0.0009
	20-30%	0.0114*	0.0030	**0.0014**	0.0016
	30-40%	0.0212*	0.0040	**0.0020**	0.0023
	40-50%	0.0366*	0.0056	**0.0027**	0.0032
	50-60%	0.0554*	0.0075	**0.0035**	0.0040
	60-70%	nan	0.0104	**0.0043**	0.0050
LPIPS↓	0-10%	0.0508	0.0337	0.0103	**0.0074**
	10-20%	0.1344	0.0514	0.0291	**0.0210**
	20-30%	0.2280*	0.0729	0.0505	**0.0350**
	30-40%	0.3332*	0.0966	0.0717	**0.0483**
	40-50%	0.4538*	0.1389	0.0984	**0.0654**
	50-60%	0.5026*	0.1897	0.1301	**0.0836**
	60-70%	nan	0.2598	0.1663	**0.1046**
FID↓	0-10%	0.6250	0.4700	0.2400	**0.1503**
	10-20%	1.2050	0.6770	0.4649	**0.3420**
	20-30%	1.5920*	0.8580	0.6848	**0.5188**
	30-40%	2.1630*	1.0250	0.8046	**0.6160**
	40-50%	3.0270*	1.2290	0.9228	**0.7588**
	50-60%	4.0190*	1.3980	1.0763	**0.8771**
	60-70%	nan	1.5770	1.1750	**1.0039**

(a) Results on curve-like masks without border constraint.

	Mask Ratio	TCCDS	EC#	CombCN	Ours
MSE↓	0-10%	0.0019	0.0011	**0.0003**	0.0003
	10-20%	0.0019	0.0021	**0.0009**	0.0010
	20-30%	0.0104*	0.0031	**0.0015**	0.0017
	30-40%	0.0267*	0.0044	**0.0021**	0.0025
	40-50%	0.0288*	0.0058	**0.0028**	0.0032
	50-60%	0.0091*	0.0080	**0.0037**	0.0043
	60-70%	nan	0.0106	**0.0046**	0.0054
LPIPS↓	0-10%	0.0507	0.0338	0.0100	**0.0072**
	10-20%	0.0507	0.0549	0.0325	**0.0231**
	20-30%	0.2393*	0.0802	0.0563	**0.0390**
	30-40%	0.3807*	0.1145	0.0840	**0.0560**
	40-50%	0.4390*	0.1592	0.1136	**0.0742**
	50-60%	0.4114*	0.2166	0.1555	**0.0988**
	60-70%	nan	0.2789	0.1980	**0.1236**
FID↓	0-10%	0.6250	0.4460	0.2308	**0.1422**
	10-20%	0.6250	0.6840	0.5046	**0.3589**
	20-30%	1.7090*	0.9240	0.7142	**0.5141**
	30-40%	2.4970*	1.1150	0.8544	**0.6412**
	40-50%	3.2270*	1.2800	0.9866	**0.7640**
	50-60%	2.1660*	1.4440	1.1019	**0.9103**
	60-70%	nan	1.5720	1.2285	**1.0443**

(b) Results on curve-like masks with border constraint.

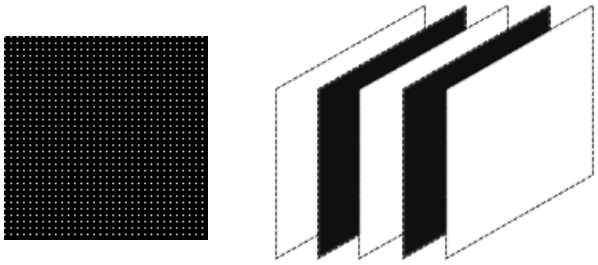
	Mask Ratio	TCCDS	EC#	CombCN	Ours
MSE↓	0-10%	0.0017	0.0016	**0.0006**	0.0007
	10-20%	0.0034	0.0033	**0.0018**	0.0021
	20-30%	0.0060	0.0050	**0.0031**	0.0036
	30-40%	0.0090	0.0071	**0.0046**	0.0053
	40-50%	0.0125*	0.0094	**0.0061**	0.0072
	50-60%	0.0189*	0.0118	**0.0079**	0.0092
	60-70%	0.0257*	0.0145	**0.0098**	0.0114
LPIPS↓	0-10%	0.459	0.0411	0.0352	**0.0152**
	10-20%	0.0791	0.0717	0.0905	**0.0428**
	20-30%	0.1343	0.1024	0.1500	**0.0716**
	30-40%	0.1923	0.1360	0.2108	**0.1036**
	40-50%	0.2472*	0.1706	0.2613	**0.1369**
	50-60%	0.3221*	0.2075	0.3161	**0.1725**
	60-70%	0.3824*	0.2493	0.3737	**0.2121**
FID↓	0-10%	0.6020	0.5150	0.3869	**0.2483**
	10-20%	0.8320	0.7680	0.7528	**0.5509**
	20-30%	1.1250	0.9710	0.9979	**0.7863**
	30-40%	1.3520	1.1380	1.2075	**0.9890**
	40-50%	1.5280*	1.2930	1.3161	**1.1250**
	50-60%	1.7030*	1.3610	1.4362	**1.2622**
	60-70%	1.9430*	1.4820	1.5380	**1.3733**

(c) Results on object-like masks without border constraint.

	Mask Ratio	TCCDS	EC#	CombCN	Ours
MSE↓	0-10%	0.0015	0.0016	**0.0006**	0.0008
	10-20%	0.0033	0.0031	**0.0017**	0.0019
	20-30%	0.0058	0.0048	**0.0029**	0.0034
	30-40%	0.0089	0.0066	**0.0042**	0.0050
	40-50%	0.0137*	0.0090	**0.0059**	0.0070
	50-60%	0.0215*	0.0115	**0.0077**	0.0091
	60-70%	0.0249*	0.0141	**0.0097**	0.0117
LPIPS↓	0-10%	0.0412	0.0421	0.0365	**0.0165**
	10-20%	0.0847	0.0716	0.0910	**0.0433**
	20-30%	0.1441	0.1052	0.1539	**0.0753**
	30-40%	0.1995	0.1378	0.2102	**0.1081**
	40-50%	0.2733*	0.1756	0.2704	**0.1441**
	50-60%	0.3491*	0.2181	0.3301	**0.1841**
	60-70%	0.4176*	0.2589	0.3834	**0.2263**
FID↓	0-10%	0.5030	0.4980	0.3819	**0.2493**
	10-20%	0.8590	0.7900	0.7564	**0.5433**
	20-30%	1.1420	1.0030	1.0142	**0.7733**
	30-40%	1.3210	1.0960	1.1646	**0.9257**
	40-50%	1.5630*	1.2560	1.3237	**1.1331**
	50-60%	1.8180*	1.3950	1.4317	**1.2592**
	60-70%	2.2120*	1.4540	1.5375	**1.3686**

(d) Results on masks with border constraint.

Table 1: Quantitative comparison on our FVI testing set. *TCCDS fails on some masks; the results are average of the successful cases. #Edge-Connect is pretrained on the Places2 [11] dataset.



(a) Spatial super resolution (b) Frame interpolation

Figure 1: Visualization for extensions of masks for video super-resolution and video interpolation.

	None	Patch GAN	VGAN*	T-PatchGAN (Ours)
LPIPS	0.1716	0.1403	0.1462	**0.1209**
FID	1.201	1.138	1.144	**1.034**
Param. #	**0**	∼6M	∼50M	∼50M

Table 2: Different GAN discriminator comparison on the FVI testing set with object-like masks. *We modified the VGAN discriminator to fit our the FVI input size.

5.2. Qualitative Comparison

See Fig. 2. Our model could recover some details (e.g. the eye of the bird).

6. Extension to Object Removal Task

See Fig. 10. Our model is more capable of recovering structured patterns and filling reasonable background.

7. Free-form Video Masks Generation Algorithm

See procedure **FreeFormVideoMask** in Algorithm 1. Function **RandomStroke**, derived from [10], determines a stroke for the first frame. Different from [10], we apply the concepts of velocity and acceleration to the head point h to ensure its smooth movement like a human drawn stroke. Function **RandomMove** realizes the deformation of a stroke by randomly moving each control point with probability p^p_{move} and the stroke movement by using stroke velocity v^s and θ^s.

8. GAN Discriminator Comparison

There are different GANs proposed for video generation, such as VGAN [7], TGAN [5], MoCoGAN [6] and TempoGAN [9]. Most of the them use naive 0/1 3D discriminators with designed generators for a specific task, which could not be applied to free-form video inpainting. Therefore, we only compare our T-PatchGAN discriminator with the naive 0/1 discriminator in VGAN and the original PatchGAN discriminator for image inpainting in [2].

From Table 2 we could see that our T-PatchGAN discriminator outperforms the naive VGAN discriminator with the same number of parameters. Although compared with the image discriminator, more parameters are used, our T-PatchGAN increases the video perceptual quality and temporal consistency by utilizing information from neighboring frames. In addition, these additional parameters are not used for inference, so the inference speed is the same.

References

[1] J.-B. Huang, S. B. Kang, N. Ahuja, and J. Kopf. Temporally coherent completion of dynamic video. *ACM Transactions on Graphics (TOG)*, 35(6):196, 2016. 1, 6, 7, 8, 9

[2] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017. 3

[3] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 1

[4] K. Nazeri, E. Ng, T. Joseph, F. Qureshi, and M. Ebrahimi. Edgeconnect: Generative image inpainting with adversarial edge learning. 2019. 1, 6, 7, 8, 9

[5] M. Saito, E. Matsumoto, and S. Saito. Temporal generative adversarial nets with singular value clipping. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2830–2839, 2017. 3

[6] S. Tulyakov, M.-Y. Liu, X. Yang, and J. Kautz. Mocogan: Decomposing motion and content for video generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1526–1535, 2018. 3

[7] C. Vondrick, H. Pirsiavash, and A. Torralba. Generating videos with scene dynamics. In *Advances In Neural Information Processing Systems*, pages 613–621, 2016. 3

[8] C. Wang, H. Huang, X. Han, and J. Wang. Video inpainting by jointly learning temporal structure and spatial details. In *Proceedings of the 33th AAAI Conference on Artificial Intelligence*, 2019. 1, 6, 7, 8, 9

[9] Y. Xie, E. Franz, M. Chu, and N. Thuerey. tempogan: A temporally coherent, volumetric gan for super-resolution fluid flow. *ACM Transactions on Graphics (TOG)*, 37(4):95, 2018. 3

[10] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang. Free-form image inpainting with gated convolution. *arXiv preprint arXiv:1806.03589*, 2018. 3

[11] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 2

Algorithm 1 Algorithm to generate free-form video masks. N^s_{max} control the maximum number of strokes. b_{min}, b_{max}, N^p_{min}, and N^p_{max} are parameters for stroke brush width and number of a stroke's control points. v^h_{max}, θ^h_{max}, Δv^h_{max}, $\Delta\theta^h_{max}$ are parameters to control stroke shape. p^p_{move}, Δx^p_{max}, Δy^p_{max}, and are parameters to control stroke deformation. v^s_{max}, θ^s_{max}, Δv^s_{max}, and $\Delta\theta^s_{max}$ are parameters to control stroke movement.

function RANDOMSTROKE()
  $x_0, y_0 \leftarrow$ uniform$(0, W)$, unifrom$(0, H)$
  $v^h, \theta^h \leftarrow$ uniform(0, v^h_{max}), uniform(0, θ^h_{max})
  $N \leftarrow$ uniform(N^p_{min}, N^p_{max})
  for i **from** 1 **to** $N-1$ **do**
    $v^h \leftarrow v^h$ + uniform$(-\Delta v^h_{max}, \Delta v^h_{max})$
    $\theta^h \leftarrow v^h$ + uniform$(-\Delta\theta^h_{max}, \Delta\theta^h_{max})$
    $x_i, y_i \leftarrow x_{i-1} + v^h \cos\theta^h, y_{i-1} + v^h \sin\theta^h$
  $x, y \leftarrow [x_0, \ldots, x_{N-1}], [y_0, \ldots, y_{N-1}]$
  $v^s, \theta^s \leftarrow |$normal$(0, v^s_{max})|$, uniform$(0, 2\pi)$
  $b \leftarrow$ uniform(b_{min}, b_{max})
  $S \leftarrow (x, y, v^s, \theta^s, b)$
  return S
function RANDOMMOVE(S)
  $x, y, v^s, \theta^s, b \leftarrow S$
  $N \leftarrow |x|$
  for i **from** 0 **to** $N-1$ **do**
    $x_i, y_i \leftarrow x_i + v^s \cos\theta^s, y_i + v^s \sin\theta^s$
    $p \leftarrow$ uniform(0, 1)
    if $p < p^p_{move}$ **then**
      $x_i \leftarrow x_i$+ uniform$(-\Delta x^p_{max}, \Delta x^p_{max})$
      $y_i \leftarrow y_i$+ uniform$(-\Delta y^p_{max}, \Delta y^p_{max})$
  $v^s, \theta^s \leftarrow v^s$+normal$(0, \Delta v^s_{max}), \theta^s$+normal$(0, \Delta\theta^s_{max})$
  $S \leftarrow (x, y, v^s, \theta^s, b)$
  return S
function DRAWSTROKE(S, M)
  $x, y, v^s, \theta^s, b \leftarrow S$
  Draw a line between (x_i, y_i) and $(x_{i+1}, y_{i+1}), \forall i \in [0, |x|-2]$ with brush width b on M
  Draw a circle on $(x_i, y_i), \forall i \in [0, |x|-1]$ with radius $\frac{b}{2}$ on M to ensure smoothness of the stroke
  return M
procedure FREEFORMVIDEOMASK
  $N^s \leftarrow$ uniform$(0, N^s_{max})$
  for i **from** 0 **to** N^s-1 **do**
    $S_i \leftarrow$ RANDOMSTROKE()
  for t **from** 0 **to** videoLength -1 **do**
    $M_t \leftarrow$ empty mask
    for i **from** 0 **to** N^s-1 **do**
      $M_t \leftarrow$ DRAWSTROKE(S_i, M_t)
      $S_i \leftarrow$ RANDOMMOVE(S_i)
  return $\{M_t | t \in [0, \text{videoLength}-1]\}$

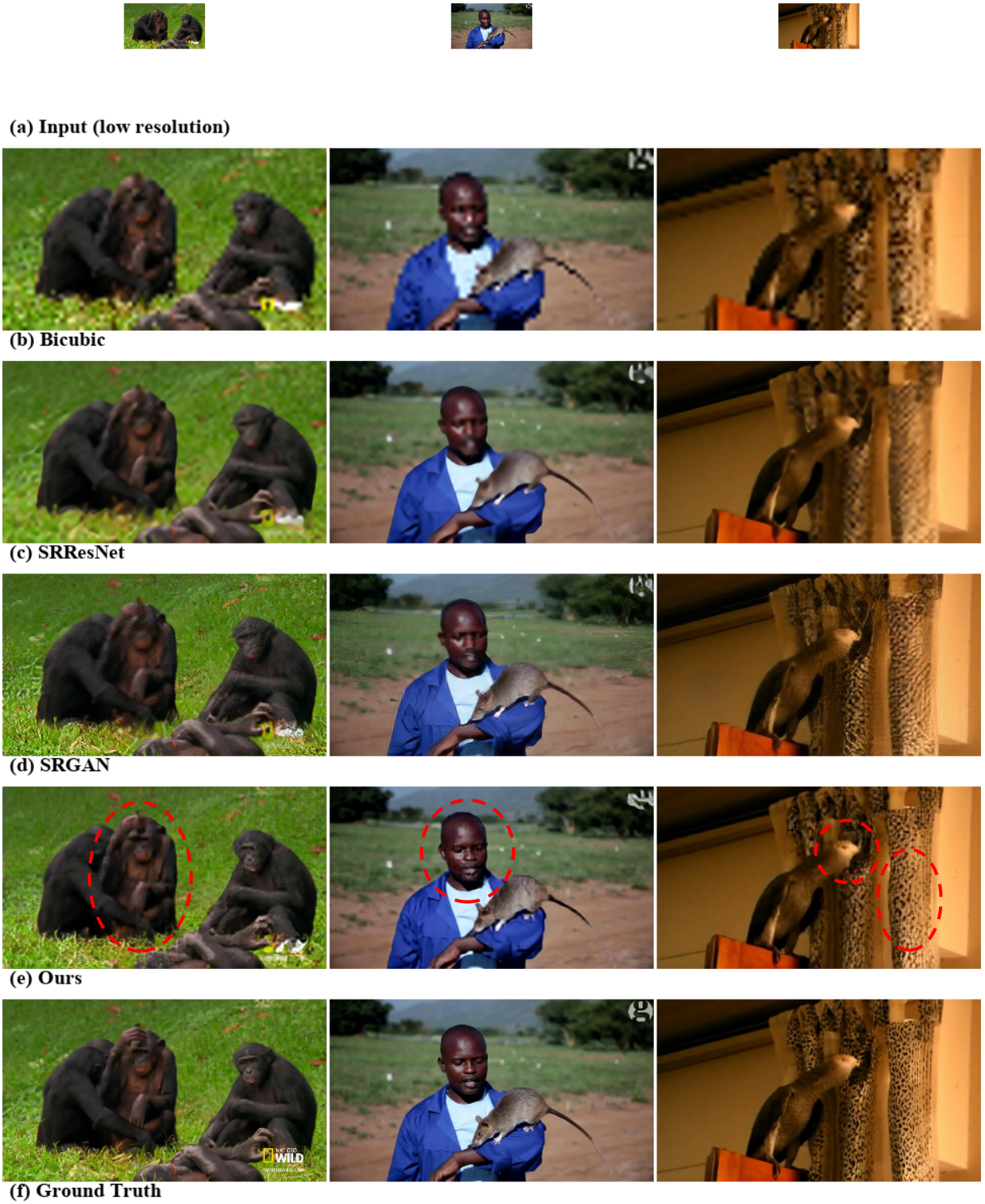


Figure 2: Qualitative comparison for 4x up-sampling video super-resolution.

Figure 3: Qualitative comparison with other methods. TCCDS [1] is prone to finding wrong patches at complex regions. Edge-Connect [4] produces plausible results for each frame but suffers from temporal inconsistency (see changes in the circled area). CombCN [8] generally generates blurry images in both object-like and curve-like masks.

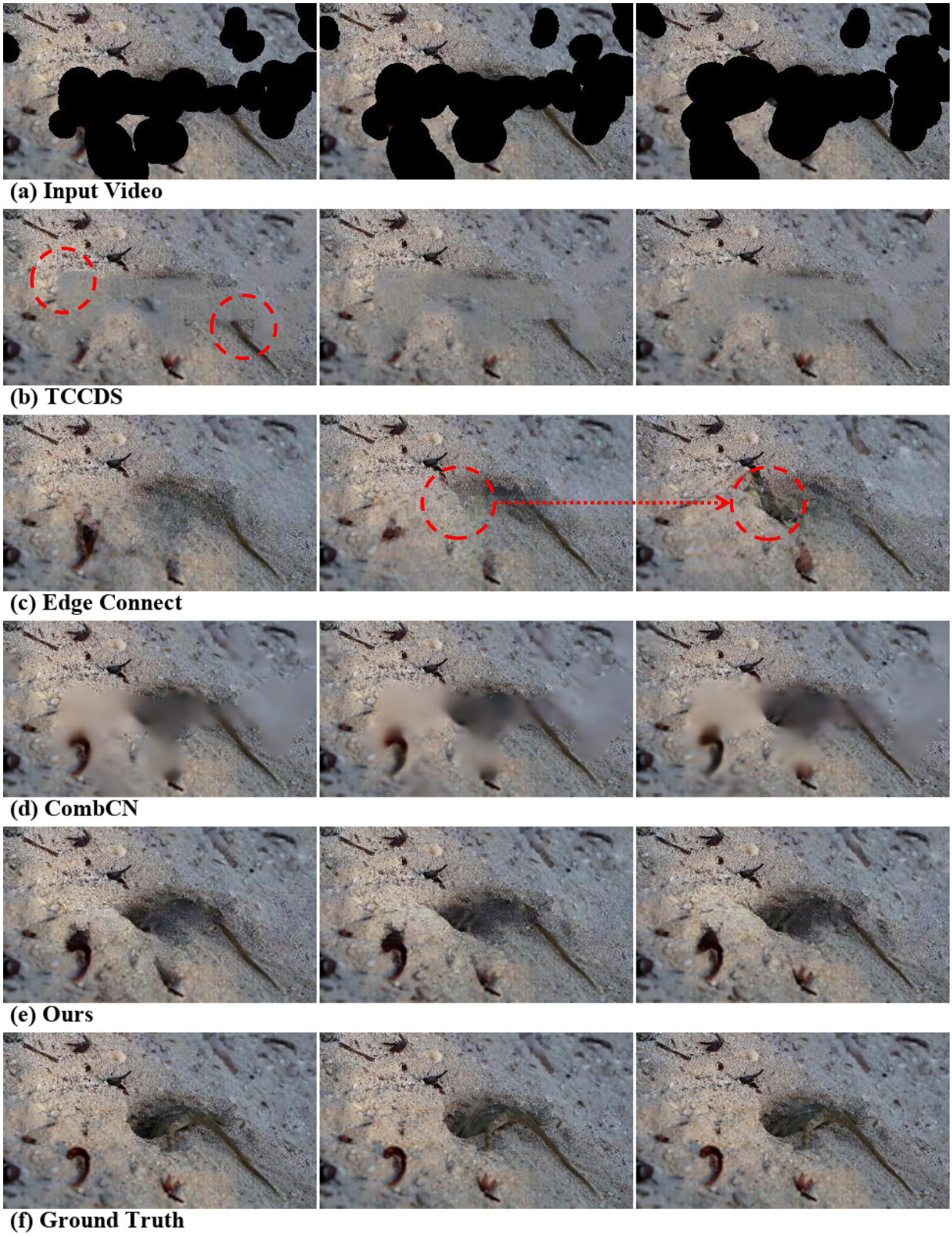


Figure 4: Qualitative comparison with other methods. TCCDS [1] is prone to finding wrong patches at complex regions. Edge-Connect [4] produces plausible results for each frame but suffers from temporal inconsistency (see changes in the circled area). CombCN [8] generally generates blurry images in both object-like and curve-like masks.

Figure 5: Qualitative comparison with other methods. TCCDS [1] is prone to finding wrong patches at complex regions. Edge-Connect [4] produces plausible results for each frame but suffers from temporal inconsistency (see changes in the circled area). CombCN [8] generally generates blurry images in both object-like and curve-like masks.

Figure 6: Qualitative comparison with other methods. TCCDS [1] is prone to finding wrong patches at complex regions. Edge-Connect [4] produces plausible results for each frame but suffers from temporal inconsistency (see changes in the circled area). CombCN [8] generally generates blurry images in both object-like and curve-like masks.

Figure 7: Qualitative comparison for ablation study. Without 3D convolutions (using 2D convolutions), the results are temporally inconsistent (see changes in the red circles); without gated convolutions (using vanilla convolutions), there are more artifacts in some complex areas (see the red circles); without T-PatchGAN, all masked areas tend to have fish scale artifacts.

Figure 8: Qualitative comparison for ablation study. Without 3D convolutions (using 2D convolutions), the results are temporally inconsistent (see changes in the red circles); without gated convolutions (using vanilla convolutions), there are more artifacts in some complex areas (see the red circles); without T-PatchGAN, all masked areas tend to have fish scale artifacts.

Figure 9: Qualitative comparison for ablation study. Without 3D convolutions (using 2D convolutions), the results are temporally inconsistent (see changes in the red circles); without gated convolutions (using vanilla convolutions), there are more artifacts in some complex areas (see the red circles); without T-PatchGAN, all masked areas tend to have fish scale artifacts.

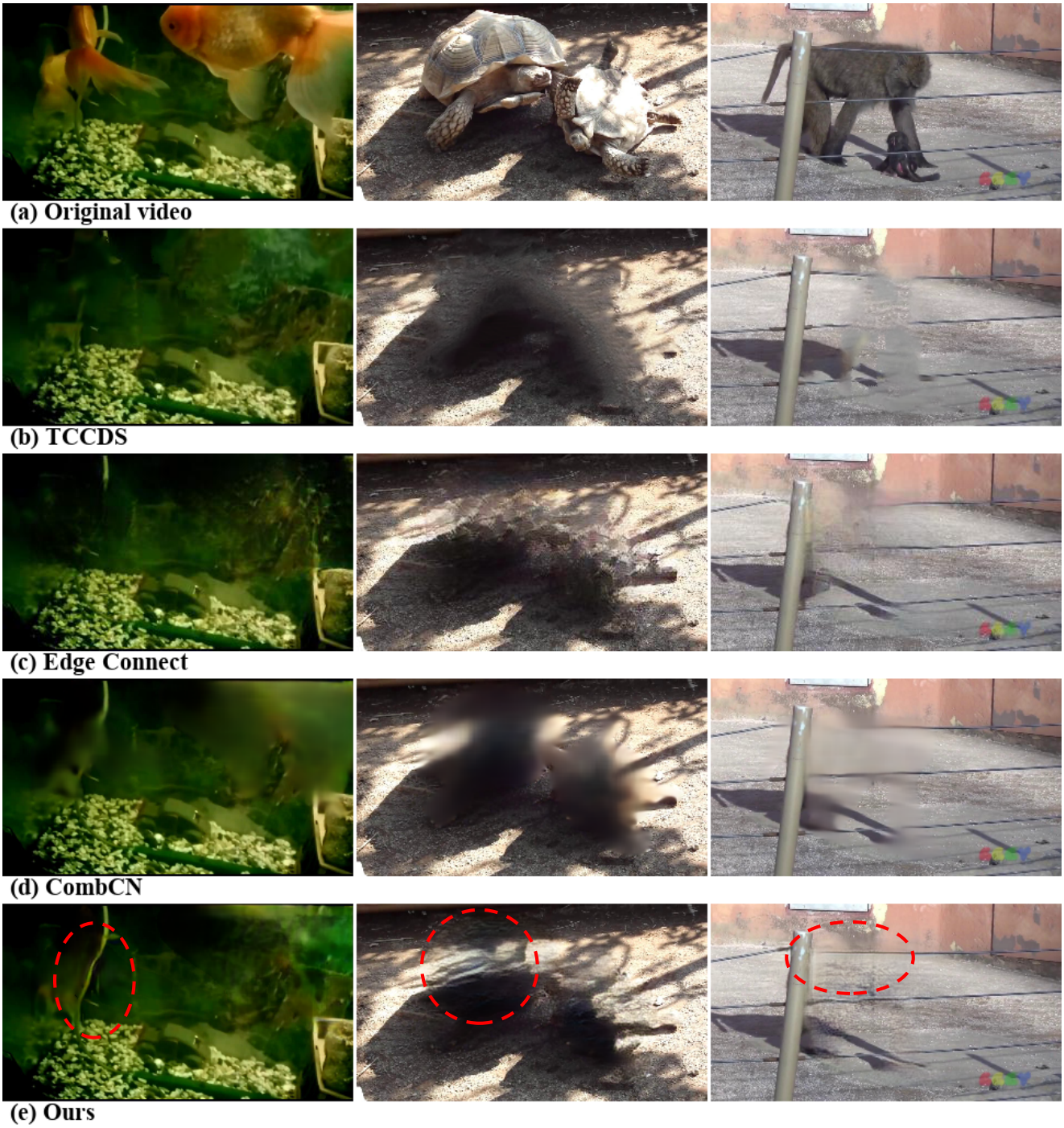


Figure 10: Object removal examples. Our model successfully recovers the aquatic and the string in the first and third example. The background of the second example is also filled with appropriate shadow.