Learning to See Through Obstructions
Supplementary Material

Yu-Lun Liu[1,4] Wei-Sheng Lai[2] Ming-Hsuan Yang[2,3] Yung-Yu Chuang[1] Jia-Bin Huang[5]
[1]National Taiwan University [2]Google [3]UC Merced [4]MediaTek Inc. [5]Virginia Tech
https://www.cmlab.csie.ntu.edu.tw/~yulunliu/ObstructionRemoval

1. Overview

In this supplementary material, we present additional results to complement the main manuscript. First, we describe the detailed training steps in Section 2. Second, we show the detailed procedure for our synthetic reflection sequences generation process in Section 4. Third, we illustrate the network architecture of the initial flow decomposition network in Section 3. Finally, we analyze the effect of initial flow decomposition, background/reflection layer reconstruction, TV loss, and visualize temporal consistency of our video reflection removal results in Section 5. We also provide comprehensive visual results in our project website.

2. Training Algorithm

We describe the training steps of our two-stage training strategy and unsupervised online optimization in Algorithm 1 and Algorithm 2. We implement our model with TensorFlow. We use the Adam optimizer to update the network parameters and set the learning rate to 0.0001 and batch size to 2. Each training sample contains five consecutive frames. The hyper-parameter settings are the same for both the training on synthetic data and the online optimization on real data. We provide the detailed training algorithms and network architectures in the supplementary material.

Algorithm 1 Two-stage training strategy

Input: Quadruplets frames $\{Q_t\}$
Output: Parameters of the initial flow decomposition network Θ_F, the background reconstruction network Θ_B, and the reflection reconstruction network Θ_R

while iterations $iter < 400k$ **do**
  Randomly Sample a minibatch of $\{\hat{B}_t\}$ and $\{\hat{R}_t\}$ from $\{Q_t\}$ to synthesize $\{I_t\}$.
  Feed $\{\hat{B}_t\}$ and $\{\hat{R}_t\}$ to PWC-Net to generate ground-truth dense flow fields for background and reflection layer.
  if $iter < 200k$ **then**
    Update Θ_F with loss function $\mathcal{L}_{dec}$ in Equation 6.
  else
    Fix the weights of Θ_F.
    Update Θ_B and Θ_R from level 0 to level 4 in an end-to-end fashion with loss function $\mathcal{L}$ in Equation 9.

3. Network Architecture of Initial Flow Decomposition Network

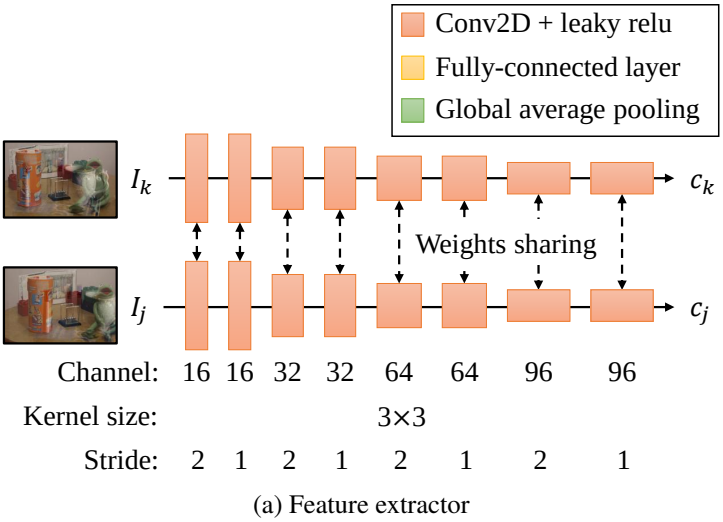
The overall architecture of the initial flow decomposition network is shown in Figure 1. Our initial flow decomposition network consists of two sub-modules: 1) a feature extractor, and 2) a layer flow estimator. Finally, we tile the global motion vectors into two *uniform* flow fields $V^0_{B,k\to j}$ and $V^0_{R,k\to j}$, for the background and reflection layers, respectively.

Algorithm 2 Online optimization

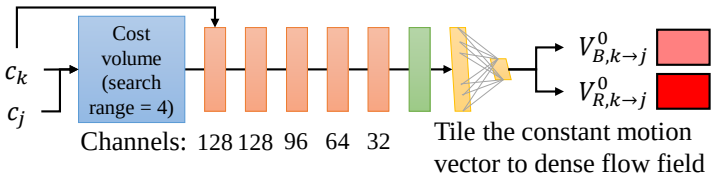
Input: Parameters of the pre-trained initial flow decomposition network Θ_F, the background reconstruction network Θ_B, and the reflection reconstruction network Θ_R

Output: Parameters of the online fine-tuned background reconstruction network Θ_B, and the reflection reconstruction network Θ_R

1: Fix the weights of Θ_F.
2: **while** iterations $iter < 1k$ **do**
3: Randomly crop the testing data to form a training minibatch.
4: Update Θ_B and Θ_R with unsupervised loss function $\mathcal{L}_{online}$ in Equation 12.



(a) Feature extractor



(b) Layer flow estimator

Figure 1: **Architecture of initial flow decomposition network.** Given a keyframe I_k and a reference frame I_j, the feature extractor first generates two features c_k and c_j. Then, we construct a cost volume with the two features and use six convolutional layers, a global average pooling layer, and a fully connected layer to generate two motion vectors. We then tile these two vectors into constant flow fields $V^0_{B,k\to j}$ and $V^0_{R,k\to j}$ for the background and reflection layers, respectively.

4. Dataset Generation

We illustrate our synthetic reflection/obstruction sequences generation process and data augmentation in Figure 2 and 3. We also provide examples of the training pairs generated from our pipeline in Figure 4 and 5.

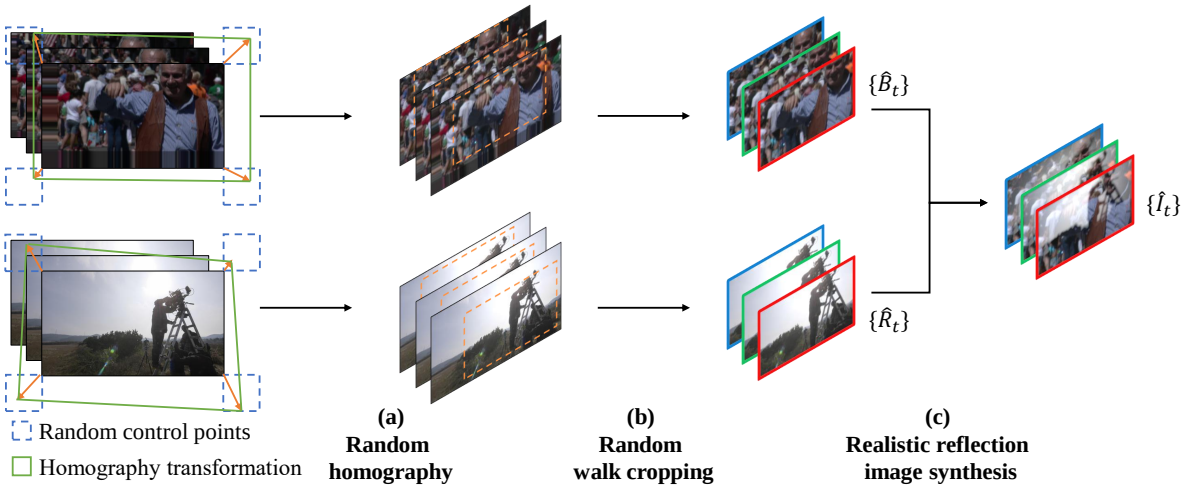


Figure 2: **Reflection sequence generation.** Given two randomly picked sequences, we first apply random homography transformations independently on every frame. Then, we apply random walk cropping to simulate camera movements. Afterward, we use the realistic reflection image synthesis model in [1, 5] to generate a sequence with reflections.

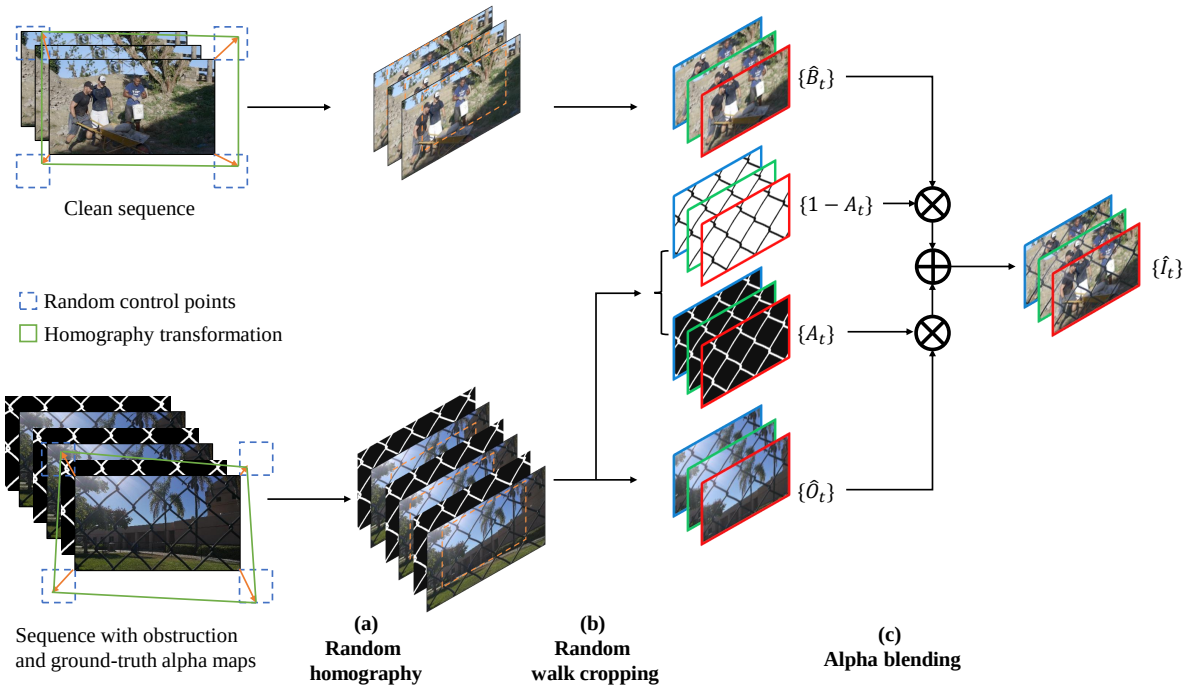


Figure 3: **Obstruction sequence generation.** We fist randomly pick a clean sequence and a sequence with fence or obstruction. Similar to the reflection sequence generation, we apply random homography and random cropping to two sequences as well as the ground-truth alpha maps of the fences or obstruction. Then, we use an alpha blending to generate a new sequence with fences or obstruction.

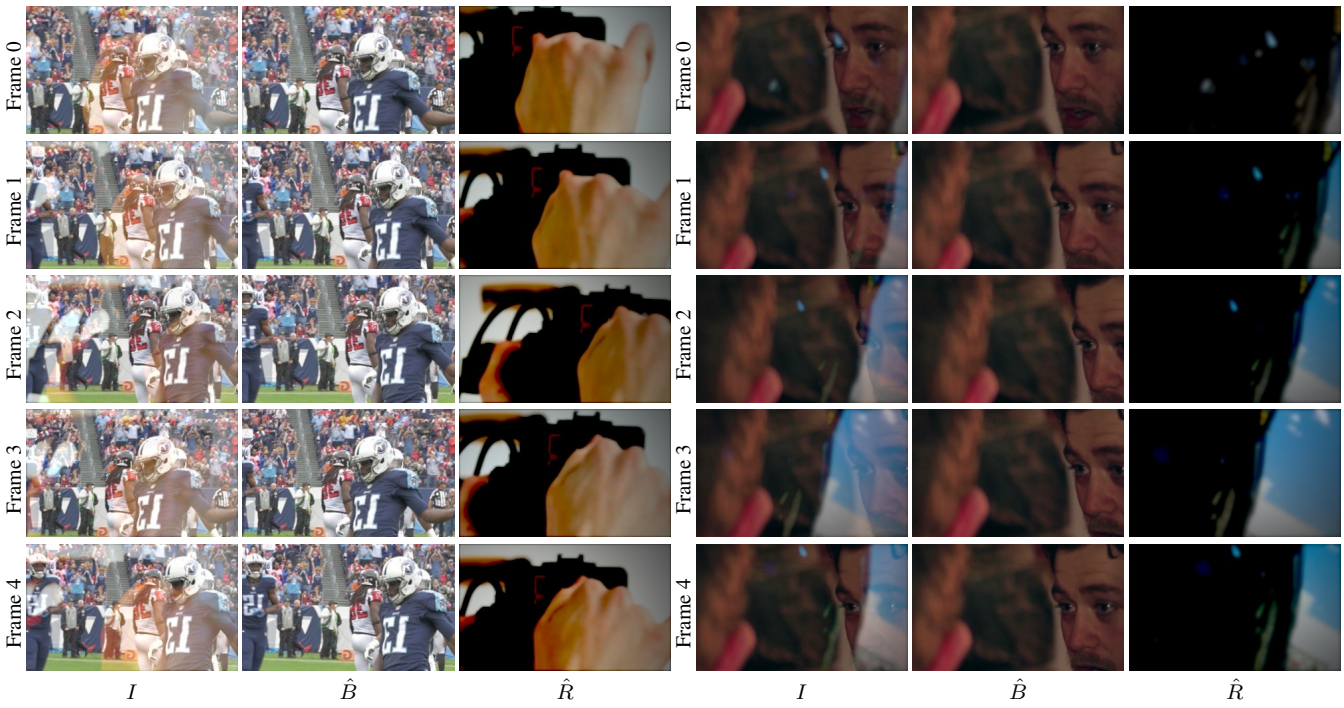


Figure 4: **Training pairs generated by our synthetic reflection data generation pipeline.**

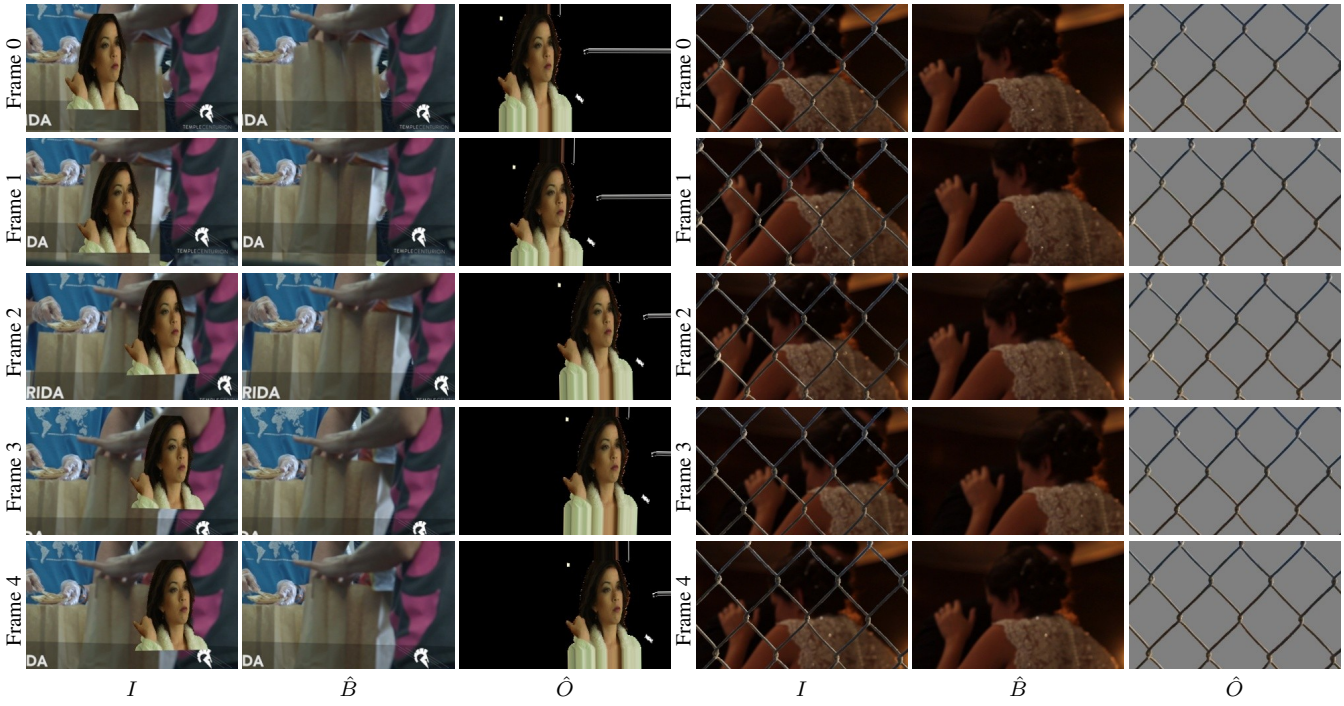
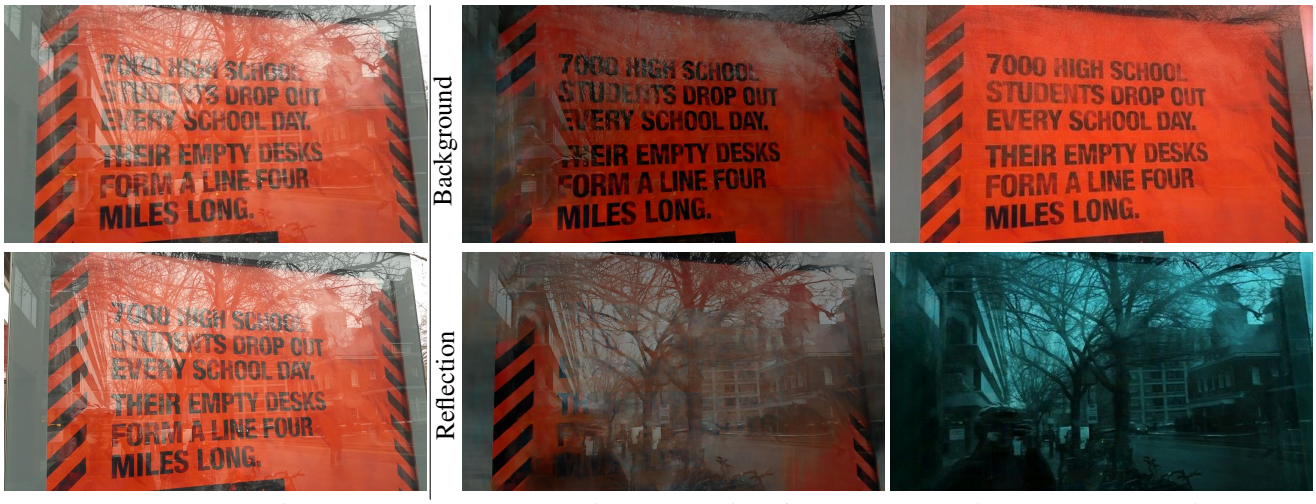


Figure 5: **Training pairs generated by our synthetic obstruction data generation pipeline.**

5. Additional Analysis

5.1. Initial Flow Decomposition

Figure 6 shows that estimating dense flow fields at the coarsest level may result in noisy predictions and lead to inconsistent layer separation. In contrast, our uniform flow prediction serves as a good initial prediction to facilitate the following background reconstruction and flow refinement steps.

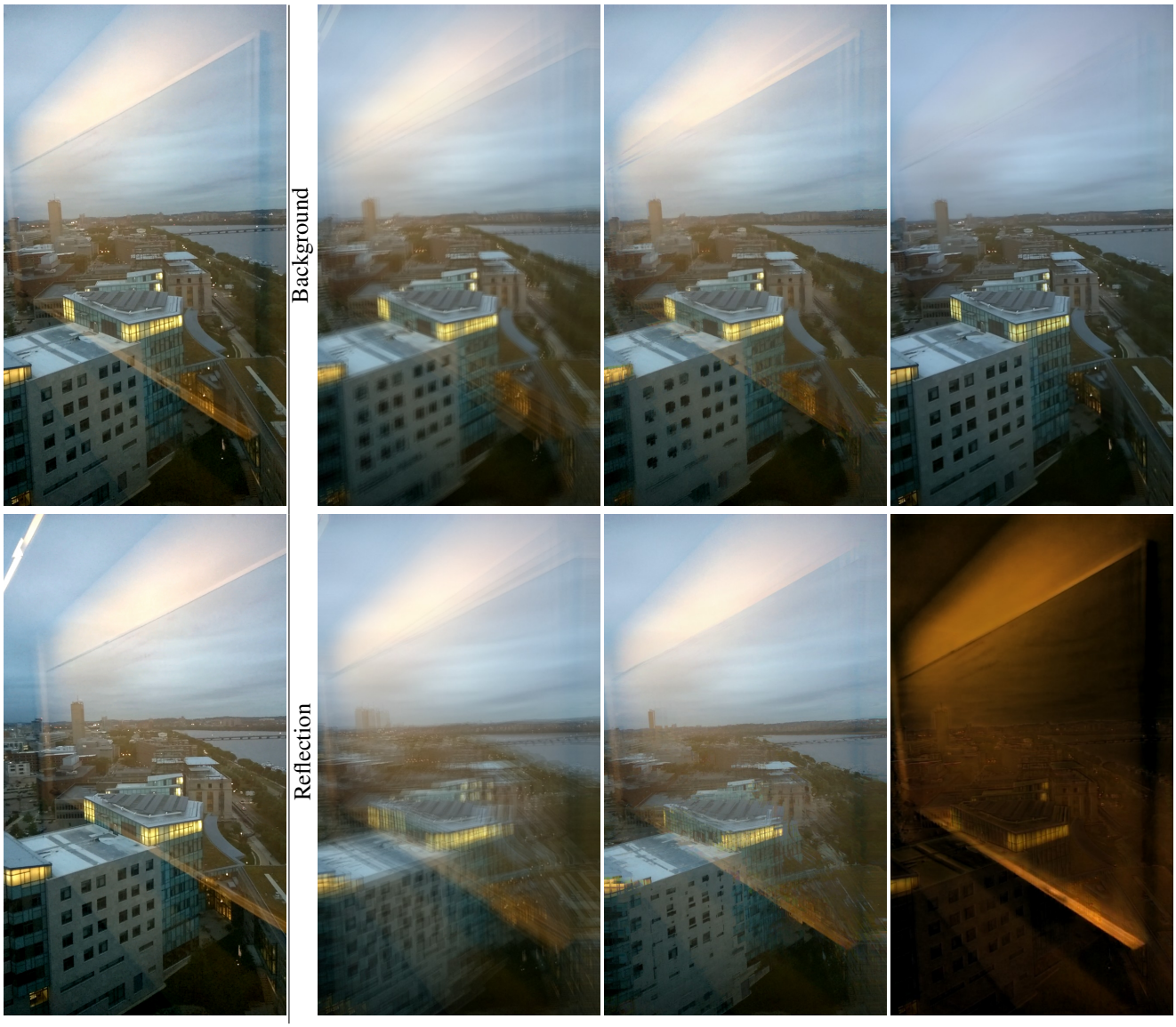


(a) Input (two rep. frames) (b) Estimate dense flow field (c) Estimate global translation vector

Figure 6: **Initialization with global translation vectors is better than dense flow field.** Predicting dense flow field with spatial-variant CNN may results in separated background and foreground in the same image. Replacing with global translation vectors leads to consistent layer predictions.

5.2. Background/Reflection Layer Reconstruction

In Figure 7, we show that the model using temporal mean or median filter for image reconstruction does not perform well and often generates ghosting artifacts. The proposed image reconstruction network, on the other hand, is capable of compensating alignment errors caused by the flow estimation in the previous level and fuses the flow-warped images into artifact-free images.



(a) Input (two rep. frames) (b) Mean (c) Median (d) Ours

Figure 7: **Image reconstruction network can compensate warping errors.**

5.3. TV loss

Figure 8 shows that online optimization without TV loss results in noisy predictions. In contrast, TV loss helps the network generating smooth predictions by regularizing sparse image gradient priors.

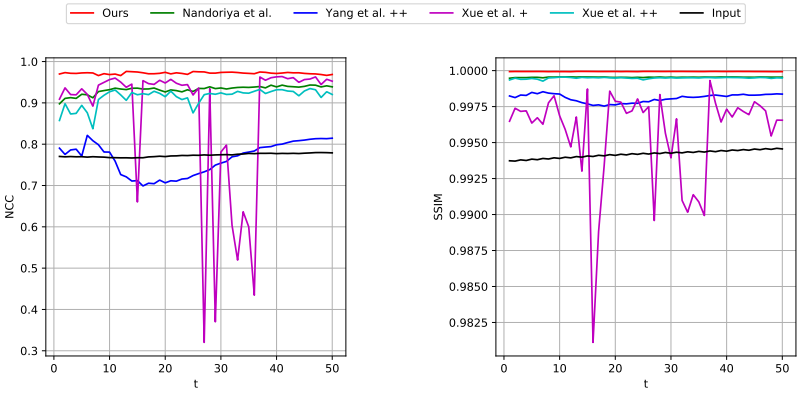
(a) Input (two rep. frames) (b) Without TV loss (c) With TV loss

Figure 8: **Online optimization with TV loss is better.**

5.4. Temporal Coherency

The proposed method takes 5 neighboring frames as input and generates the separation results for the reference frame. Although predicting each reference frame *independently*, our method still generates temporally coherent results on the entire video. Here, we compare our method with four video reflection removal approaches [3, 4, 2]. Both the methods of Xue et al. [3] and Yang et al. [4] take multiple frames as input and generates the middle frame, similar to our model. Xue et al.+ [3] is an extension of [3] which uses the moving window strategy in [4] to improve the temporal consistency. Both Xue et al.++ [3] and Yang et al.++ [4] adopt a temporal average filtering to reduce the temporal flickering. Nandoriya et al. [2] use a spatio-temporal optimization to process the entire video sequence jointly.

We evaluate the temporal consistency of each method on a controlled synthetic video sequence provided by [2], which blends two videos through an alpha blending. The two layers have different global movements. In addition, there is a third layer on the background which contains a flying bird to simulate local moving objects. Figure 9 shows that our method generates video results with better spatial separation and temporal coherency in terms of NCC and SSIM. In Figure 10, we show separation results on real input sequences, where the proposed method not only separates the background and reflection layers well but also preserves temporal coherency.



(a) NCC of the background layer

(b) SSIM of the background layer

(c) NCC of the reflection layer

(d) SSIM of the reflection layer

Figure 9: **Evaluation different reflection removal methods on a controlled synthetic sequence provided by [2].** Our method generates the best temporal coherency and layer separation.

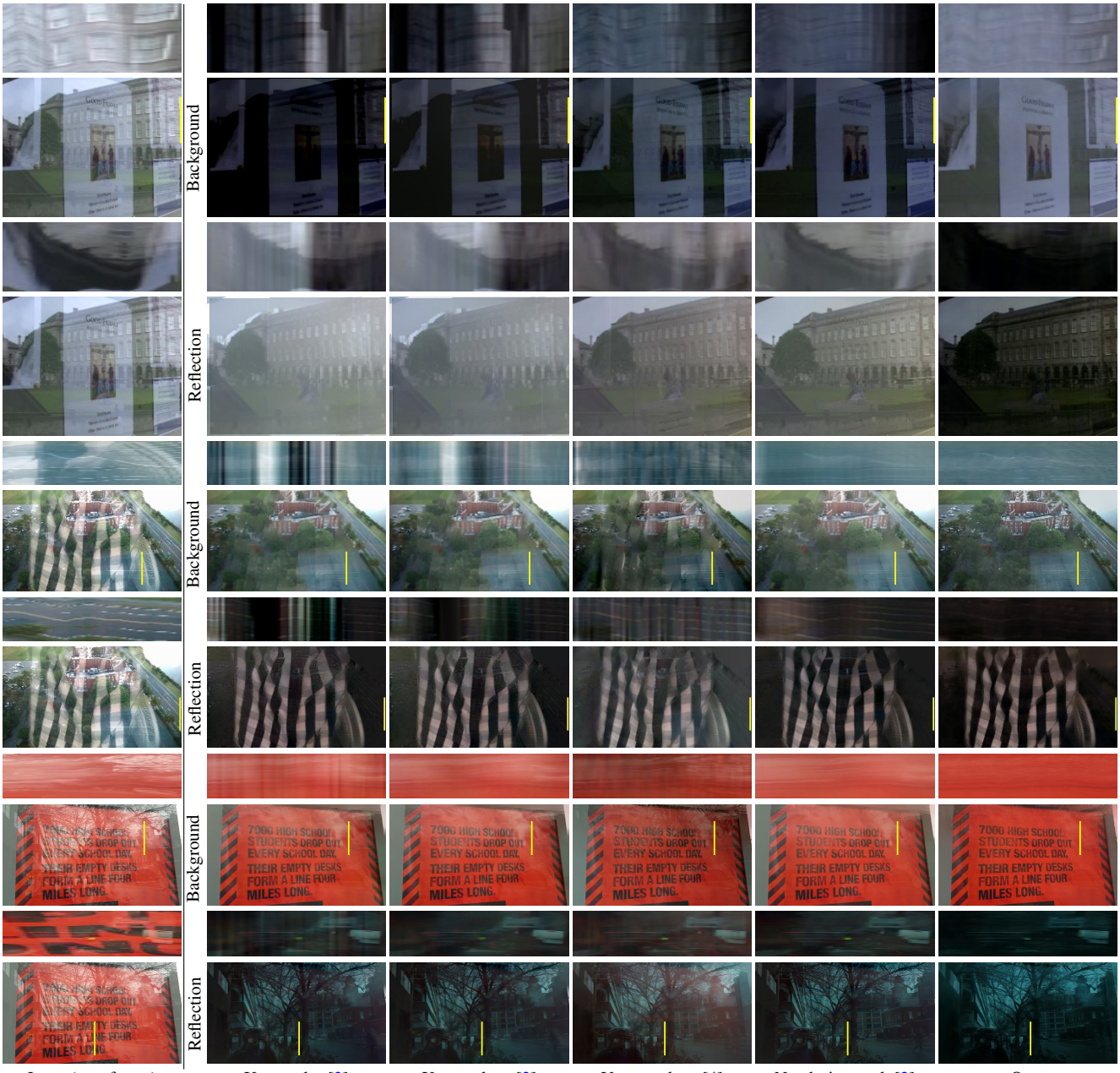


Figure 10: **Our method generate better layer separation with temporal coherency (yellow slice).** '+': applies the original method using moving window strategy as mentioned in [4]. '++': uses a moving temporal average filtering to reduce flickering based on '+'.

References

[1] Qingnan Fan, Jiaolong Yang, Gang Hua, Baoquan Chen, and David Wipf. A generic deep architecture for single image reflection removal and image smoothing. In *ICCV*, 2017. 3

[2] Ajay Nandoriya, Mohamed Elgharib, Changil Kim, Mohamed Hefeeda, and Wojciech Matusik. Video reflection removal through spatio-temporal optimization. In *ICCV*, 2017. 8, 9

[3] Tianfan Xue, Michael Rubinstein, Ce Liu, and William T Freeman. A computational approach for obstruction-free photography. *ACM TOG*, 34(4):79, 2015. 8, 9

[4] Jiaolong Yang, Hongdong Li, Yuchao Dai, and Robby T Tan. Robust optical flow estimation of double-layer images under transparency or reflection. In *CVPR*, 2016. 8, 9

[5] Xuaner Zhang, Ren Ng, and Qifeng Chen. Single image reflection separation with perceptual losses. In *CVPR*, 2018. 3