Learning to See Through Obstructions with Layered Decomposition Supplementary Material

Yu-Lun Liu, Wei-Sheng Lai, Ming-Hsuan Yang, Yung-Yu Chuang and Jia-Bin Huang

1 Overview

In this supplementary material, we present additional results to complement the main manuscript. First, we illustrate the network architecture of the initial flow decomposition network (Section 2). Second, we show the detailed procedure for our synthetic reflection sequences generation process (Section 3). Third, we analyze the effect of initial flow decomposition, background/reflection layer reconstruction, TV loss, and realistic training data generation (Section 4). Finally, we analyze and visualize the temporal consistency of our video reflection and fence removal results in Section 5. We also provide comprehensive visual results on our project website https://alex04072000.github.io/SOLD/.

2 Initial Flow Decomposition Network

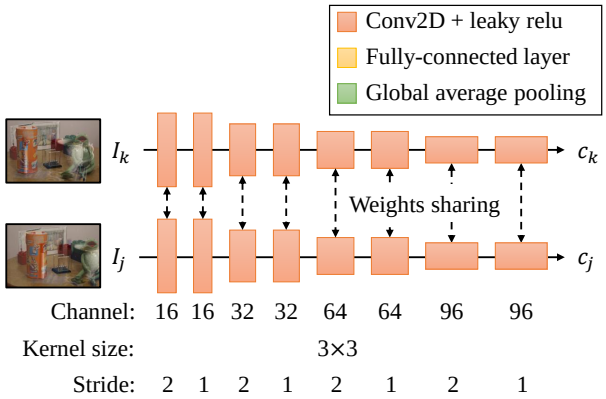
We show the overall architecture of the initial flow decomposition network in Figure 1. Our initial flow decomposition network consists of two sub-modules: 1) a feature extractor, and 2) a layer flow estimator. The feature extractor first generates deep features from the two input frames, and then the layer flow estimator applies a correlation layer to construct a cost volume from the two input features and predicts a global motion vector through a global average pooling layer. 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.

3 Dataset Generation

During the training stage, we apply on-the-fly random color augmentation, including hue, saturation, brightness, and contrast, on both background and reflection layers, as shown in Figure 2. We improve the data synthesis pipeline in [1] to generate more diverse training data. The detailed differences between the reflection image blending method of [1] and ours are summarized in Table 1. We present examples of the training pairs generated from our pipeline in Figure 3.

4 Additional Analysis

In this section, we provide additional ablation studies and analysis on our initial flow decomposition, image reconstruction network, TV loss, and realistic training data generation.



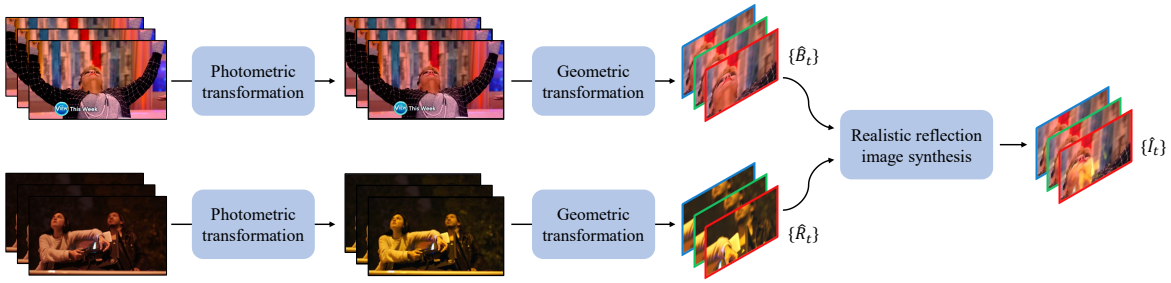
(a) Feature extractor

(b) Layer flow estimator

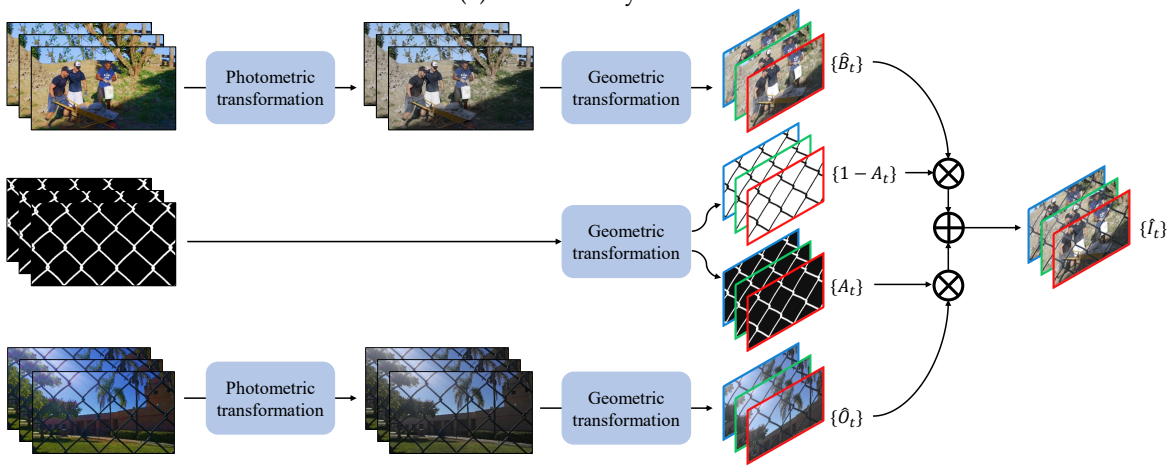
Fig. 1: **Architecture of initial flow decomposition network.** Given a keyframe I_k and a reference frame I_j, the feature extractor first generates two features c_k and c_j. Then, we construct a cost volume with the two features and use six convolutional layers, a global average pooling layer, and 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.

TABLE 1: **Detailed differences between the reflection image blending method of [1] and ours.**

Augmentations	[1]	Ours
Kernel size of Gaussian blur of reflection	×	[3, 17]
Vignette with random Gaussian kernel size	×	[300, 1000]
Random color augmentation	×	✓
Standard deviations of Gaussian noise	×	[0, 0.02]
Quality of random JPEG compression	×	[50, 100]
Motion range between frames	[-20, 20]	[-40, 40]
Number of input frames	5	[2, 7]



(a) Reflection synthesis



(b) Fence generation

Fig. 2: **Synthetic sequence generation.** Given two randomly selected sequences, we first apply random color augmentation independently on both the background and foreground layers. Then, we apply random homography transformations independently on every frame. Afterward, we apply random walk cropping to simulate camera movements. We use the realistic reflection image synthesis model in [2], [3] to generate a sequence with reflections. Finally, we augment random Gaussian noise and random JPEG compression artifacts on every fused frame.

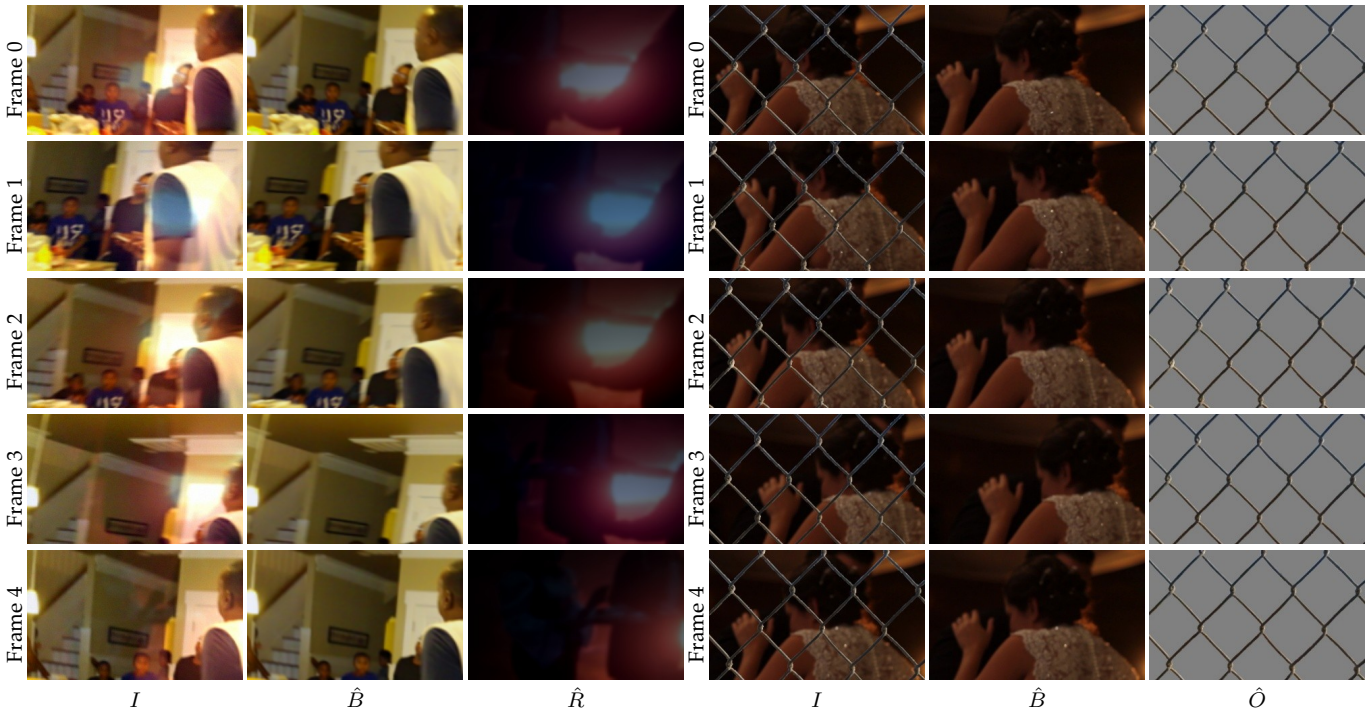
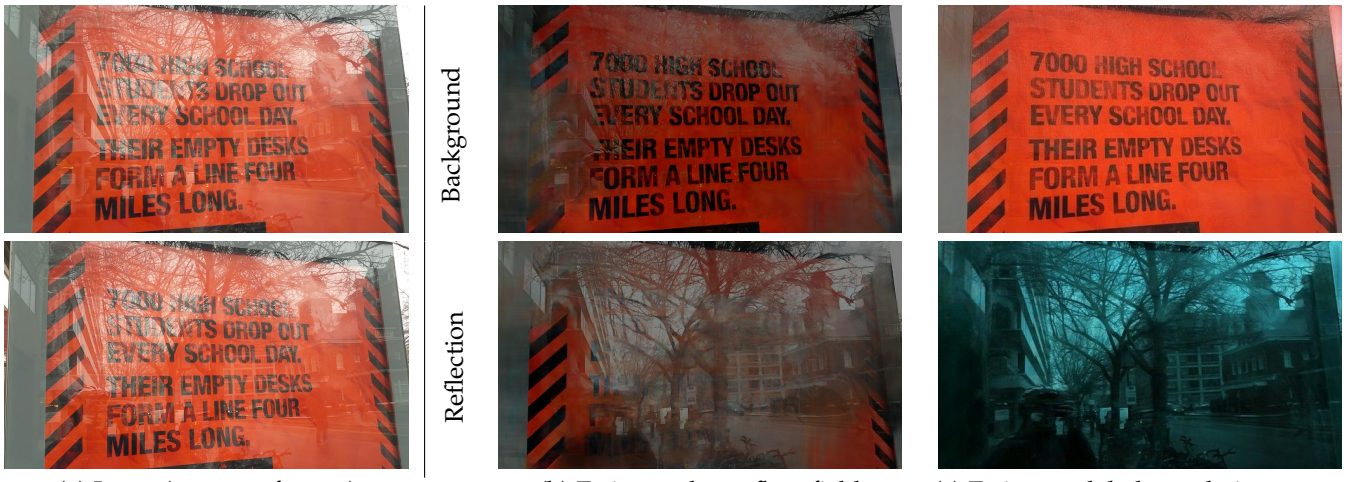
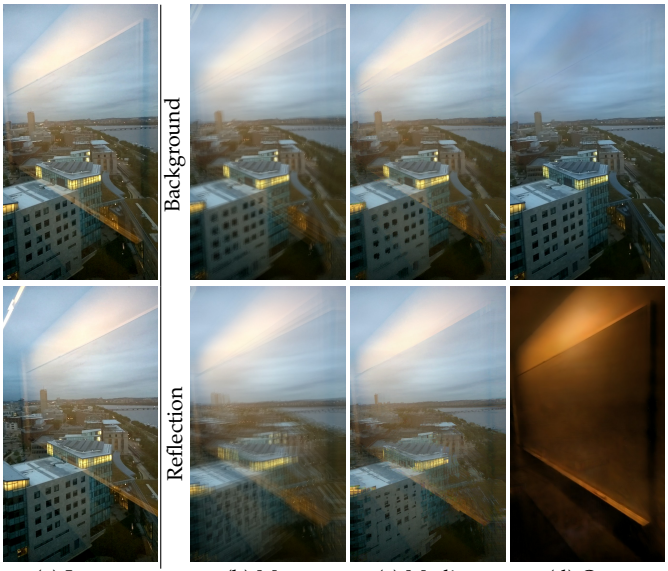


Fig. 3: **Training pairs generated by our synthetic reflection data generation pipeline.**



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

Fig. 4: **Analysis on initialization flow decomposition.** Predicting dense flow fields at the coarsest level often lead to noisy prediction, resulting in the same reconstructed background and foreground layers. Instead, our model predicts global translation vectors as the initial flows, which provide more consistent layer separation at the coarsest level.



(a) Inputs (b) Mean (c) Median (d) Ours

Fig. 5: **Effect of image reconstruction network.** Applying a simple mean or median temporal filter to the aligned frames cannot separate the background and reflection layers effectively. In contrast, our image reconstruction network learns to compensate warping errors and provide better separation results.

4.1 Initial Flow Decomposition

Figure 4 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.

4.2 Background/Reflection Layer Reconstruction

In Figure 5, we show that the model using temporal mean or median filter for image reconstruction does not perform well and often generates ghosting artifacts. In contrast, our image reconstruction network learns to reduce warping and alignment errors and generates clean foreground and background images.

(a) Input (b) w/o TV loss (c) w/ TV loss

Fig. 6: **Effect of TV loss in online optimization.** TV loss regularizes the model to predict sparse image gradients, leading to better separation of reflection and background.

4.3 TV Loss

Figure 6 shows that our 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.

4.4 Realistic Training Data Generation

Figure 7 shows that our realistic training data generation leads to better separation of background and reflection layers both qualitatively and quantitatively.

4.5 Predicted optical flow results

We show the predicted optical flows for real-world sequences in Fig. 10.

Fig. 7: **Effect of realistic training data generation.**

Fig. 8: **Our method generate better layer separation with temporal coherency (yellow slice).** '+': applies the original method using moving window strategy as mentioned in [5]. '++': uses a moving temporal average filtering to reduce flickering based on '+'.

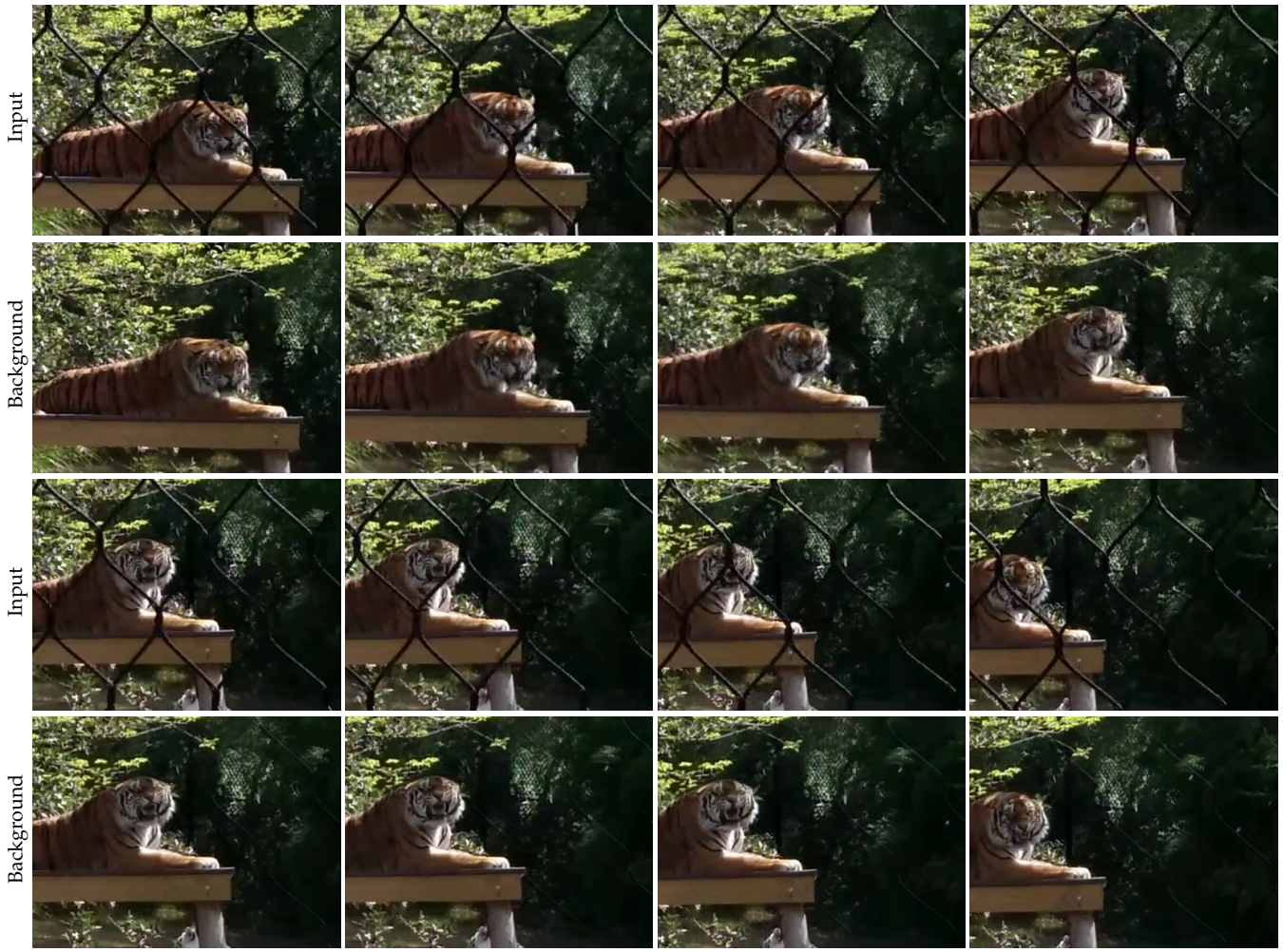


Fig. 9: **Video results for fence removal.** Our method can still generate temporally consistent results when there are moving objects in the scene, e.g. the tiger in this example.

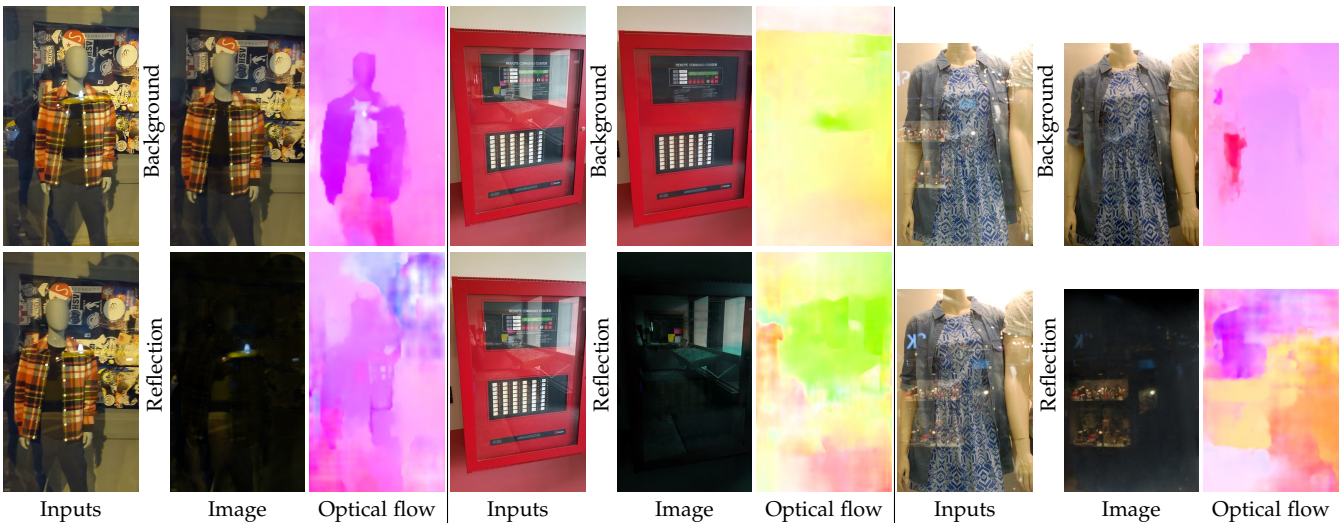


Fig. 10: **Predicted optical flows.** On the left, we show two representative input frames of the sequence. The middle shows the recovered background and reflection. On the right, we show the predicted flows for the background and reflection layers.

5 Temporal Coherency

The proposed method takes five 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 [4], [5], [6]. Both methods by Xue *et al*. [4] and Yang *et al*. [5] use multiple frames as input and generates the middle frame, similar to our model. Xue *et al*.+ [4] is an extension of [4] which uses the moving window strategy in [5] to improve the temporal consistency. Both Xue *et al*.++ [4] and Yang *et al*.++ [5] adopt a temporal average filtering to reduce the temporal flickering. Nandoriya *et al*. [6] use a spatio-temporal optimization to process the entire video sequence jointly.

We evaluate temporal consistency of each method on a controlled synthetic video sequence provided by [6], 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. In Figure 8, 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. Figure 9 shows another example that our method can deal with moving scenes objects.

References

[1] Y.-L. Liu, W.-S. Lai, M.-H. Yang, Y.-Y. Chuang, and J.-B. Huang, "Learning to see through obstructions," in *CVPR*, 2020.

[2] Q. Fan, J. Yang, G. Hua, B. Chen, and D. Wipf, "A generic deep architecture for single image reflection removal and image smoothing," in *ICCV*, 2017.

[3] X. Zhang, R. Ng, and Q. Chen, "Single image reflection separation with perceptual losses," in *CVPR*, 2018.

[4] T. Xue, M. Rubinstein, C. Liu, and W. T. Freeman, "A computational approach for obstruction-free photography," *ACM TOG*, vol. 34, no. 4, p. 79, 2015.

[5] J. Yang, H. Li, Y. Dai, and R. T. Tan, "Robust optical flow estimation of double-layer images under transparency or reflection," in *CVPR*, 2016.

[6] A. Nandoriya, M. Elgharib, C. Kim, M. Hefeeda, and W. Matusik, "Video reflection removal through spatio-temporal optimization," in *ICCV*, 2017.