Semi-Supervised Learning for Optical Flow with Generative Adversarial Networks Supplementary Material

Wei-Sheng Lai[1] **Jia-Bin Huang**[2] **Ming-Hsuan Yang**[1,3]
[1]University of California, Merced [2]Virginia Tech [3]Nvidia Research
[1]{wlai24|mhyang}@ucmerced.edu [2]jbhuang@vt.edu

1 Overview

In this supplementary document, we present additional results to complement the paper. First, we provide the detailed configurations and parameters of the generator and discriminator in the proposed Generative Adversarial Network. Second, we present the qualitative comparisons with the state-of-the-art CNN-based optical flow methods. The complete results and source code are publicly available on `http://vllab.ucmerced.edu/wlai24/semiFlowGAN`.

2 Network Architectures

We describe the network configuration and parameter settings of the proposed method in this section.

2.1 Generator G

We adapt the architecture of our generator from SPyNet [8], which builds on a spatial pyramid to estimate optical flow in a coarse-to-fine manner. SPyNet has L pyramid levels, and each level has a CNN sub-network. We denote the input image pairs as I_1 and I_2. At pyramid level l, the input images are downsampled to the corresponding size, denoted by I_1^l and I_2^l, respectively. We define the warped image as $\tilde{I}_2^l = \mathbb{W}(I_2^l, F_i^l)$ where $\mathbb{W}$ is the bilinear warping function [6] that warps the second input image I_2^l according to the input flow F_i^l. The sub-network concatenates I_1^l, $\tilde{I}_2^l$ and F_i^l as a 8-channels input and predicts the *residual* flow F_r^l. The residual flow is then combined with the input flow to produce a complete output flow $F_o^l = F_i^l + F_r^l$. The output flow at the current level is upsampled $2\times$ and multiplied by 2 to be the input flow at the next level. We illustrate a two-level architecture of our generator in Figure 1. In this work, we build a 5-level SPyNet as our generator.

Instead of using stacks of convolutional layers [8], we adopt the encoder-decoder architecture with skip connections to effectively increase the receptive fields for each sub-network. Table 1 provides a detail configuration of the sub-network in our generator. For level $l = 0$, there is not input flow, and the input of the sub-network has 6 channels (i.e., the concatenation of I_1^0 and I_2^0).

2.2 Discriminator D

We use the convolutional PatchGAN [5] as the architecture of our discriminator. Given optical flow, we compute the flow warp error image, which is the intensity difference between the first input image and the warped second image. The discriminator learns to classify whether each $N \times N$ overlapping patch of the flow warp error image is produced by the ground truth flow (real sample) or the estimated flow (fake sample) from the generator. We use 3×3 convolution followed by the Leaky ReLU activation in our discriminator. The receptive field of the discriminator is determined by the number of strided convolutional layers.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

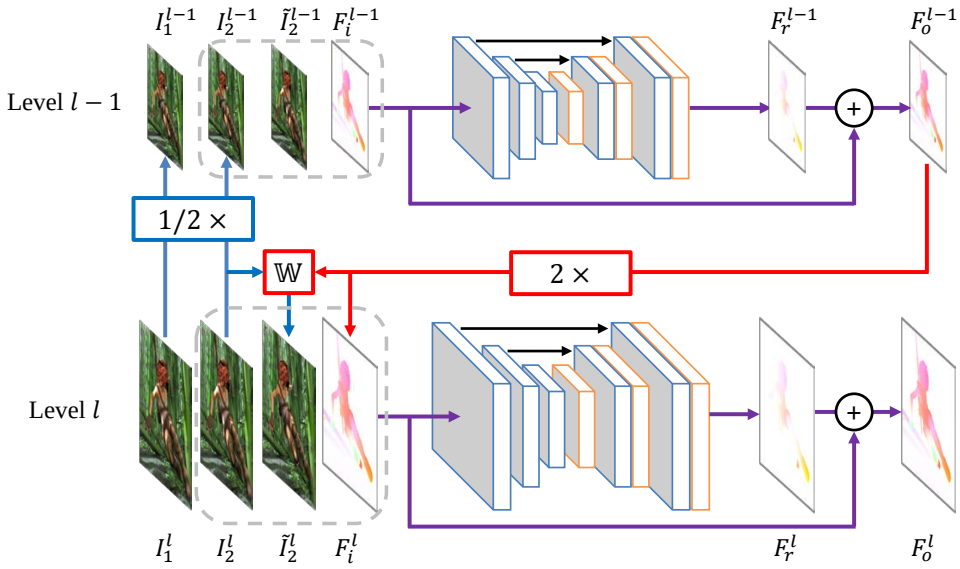


Figure 1: **Architecture of generator G.** We adapt the SPyNet [8] approach as our generator. We denote the input images and input flow at level l as I_1^l, I_2^l and F_i^l. The bilinear sampling function $\mathbb{W}$ warps I_2^l according to F_i^l and produce $\tilde{I}_2^l$. The sub-network concatenates I_1^l, $\tilde{I}_2^l$ and F_i^l as input and predicts the residual flow F_r^l. The output flow F_o^l is the summation of F_i^l and F_r^l.

Table 1: **Detailed configuration of a sub-network in our generator G.** The input of the sub-network is a concatenation of the first input image I_1^l, the warped second image $\tilde{I}_2^l$, and the input flow field F_i^l. The output is the residual flow F_r^l.

Input name	Output name	Kernel size	Stride	Input channels	Output channels	Input resolution	Output resolution
I_1^l, $\tilde{I}_2^l$, F_i^l	conv1a	3×3	1	8	64	$1\times$	$1\times$
conv1a	conv1b	3×3	2	64	128	$1\times$	$1/2\times$
conv1b	conv2a	3×3	1	128	128	$1/2\times$	$1/2\times$
conv2a	conv2b	3×3	2	128	256	$1/2\times$	$1/4\times$
conv2b	conv3a	3×3	1	256	256	$1/4\times$	$1/4\times$
conv3a	conv3b	3×3	2	256	512	$1/4\times$	$1/8\times$
conv3b	deconv3	4×4	2	512	256	$1/8\times$	$1/4\times$
upconv3, conv3a	conv3r	3×3	1	512	256	$1/4\times$	$1/4\times$
conv3r	upconv2	4×4	2	256	128	$1/4\times$	$1/2\times$
upconv2, conv2a	conv2r	3×3	1	256	128	$1/2\times$	$1/2\times$
conv2r	upconv1	4×4	2	128	64	$1/2\times$	$1\times$
upconv1, conv1a	F_r^l	3×3	1	128	2	$1\times$	$1\times$

Table 2: **Detailed configuration of our discriminator D.** Our discriminator takes a flow warp error image as input and output a probability map for predicting each $N \times N$ patch is real or fake sample.

Input name	Output name	Kernel size	Stride	Input channels	Output channels	Input resolution	Output resolution
flow warp error	conv1	3×3	2	3	64	$1\times$	$1/2\times$
conv1	conv2	3×3	2	64	128	$1/2\times$	$1/4\times$
conv2	conv3	3×3	2	128	256	$1/4\times$	$1/8\times$
conv3	conv4	3×3	1	256	512	$1/8\times$	$1/8\times$
conv4	probability map	3×3	1	512	1	$1/8\times$	$1/8\times$

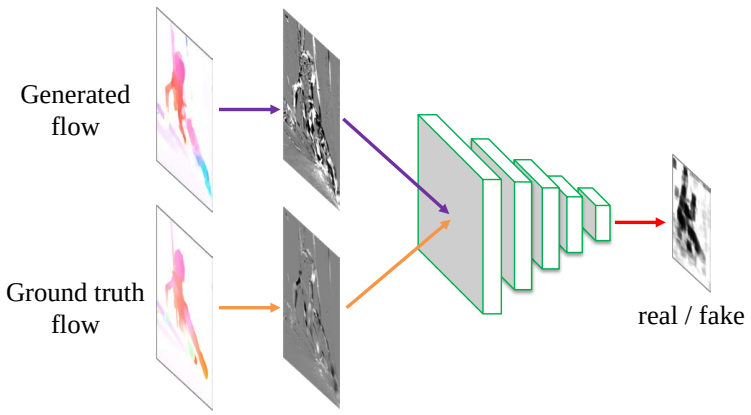


Figure 2: **Architecture of discriminator D.** We use the convolutional PatchGAN [5] as our discriminator. The discriminator learns to classify whether each $N \times N$ patch of the flow warp error image comes from the ground truth flow (real sample) or the estimated flow (fake sample).

3 Training on Virtual KITTI Dataset

In this section, we provide additional results by training the proposed method with labeled and unlabeled data from the same domain. We use the virtual KITTI dataset, which is a synthetic dataset with 50 sequences of driving scenes, as the labeled dataset and the KITTI raw videos as the unlabeled dataset. We train our model using supervised, unsupervised, baseline semi-supervised and the proposed semi-supervised settings and provide the quantitative results in Table 3.

Comparing the the models trained on the FlyingChair and KITTI raw datasets, training on the virtual KITTI dataset significantly reduce the error on the KITTI2012 and KITTI2015 datasets. The proposed semi-supervised method performs favorably against the purely supervised and baseline semi-supervised approaches. However, on the Sintel and FlyingChairs datasets, the supervised and proproposed semi-supervised methods perform even worse than the unsupervised approach. The reason lies in the fact that the scene and motion of the training datasets are much different from that of the test sets. Both the virtual KITTI and KITTI raw datasets have a strong prior of forward camera motion, while the Sintel and FlyingChairs datasets usually contain a main foreground object and small camera motion. Therefore, the models trained on the virtual KITTI and KITTI raw datasets cannot generalize well to other test datasets.

Table 3: **Training with datasets from the same domain.** "VKITTI" represents the Virtual KITTI dataset and "KITTI" denotes the KITTI raw dataset.

Setting	Training Datasets	Sintel-Clean EPE	Sintel-Final EPE	KITTI 2012 EPE	KITTI 2015 EPE	KITTI 2015 Fl-all	FlyingChairs EPE
Supervised	VKITTI	16.24	15.89	2.65	7.46	19.62%	14.95
Unsupervised	KITTI	**8.01**	**8.97**	16.54	25.53	54.40%	**6.66**
Baseline semi-supervised	VKITTI + KITTI	8.75	9.63	9.10	16.29	35.64%	8.85
Proposed semi-supervised	VKITTI + KITTI	14.08	14.48	**2.50**	**7.30**	**19.30%**	13.08

4 Qualitative Comparisons

In this section, we present visual comparisons with FlowNetS [3], FlowNetC [3] and SPyNet [8] on the Sintel [2], KITTI 2012 [4], KITTI 2015 [7], FlyingChairs [3] and Middlebury [1] datasets.

4.1 Visual comparisons on the Sintel training dataset

Figure 3: **Visual comparisons on the Sintel dataset.**

Figure 4: **Visual comparisons on the Sintel dataset.**

Figure 5: **Visual comparisons on the Sintel dataset.**

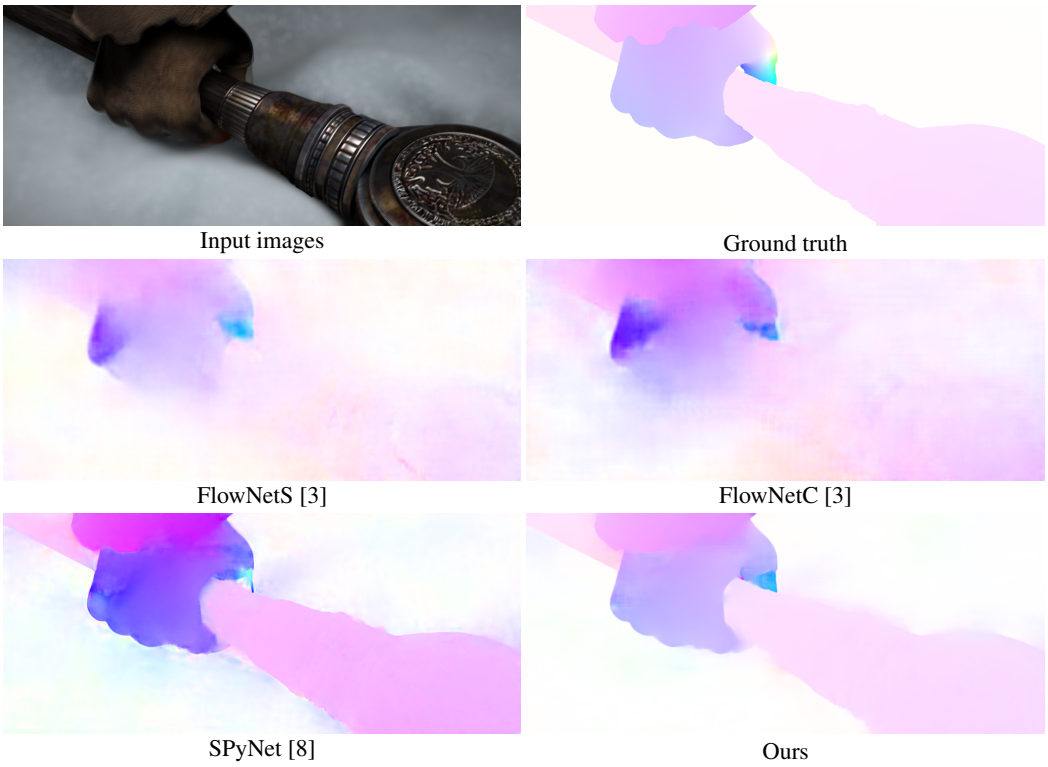


Figure 6: **Visual comparisons on the Sintel dataset.**

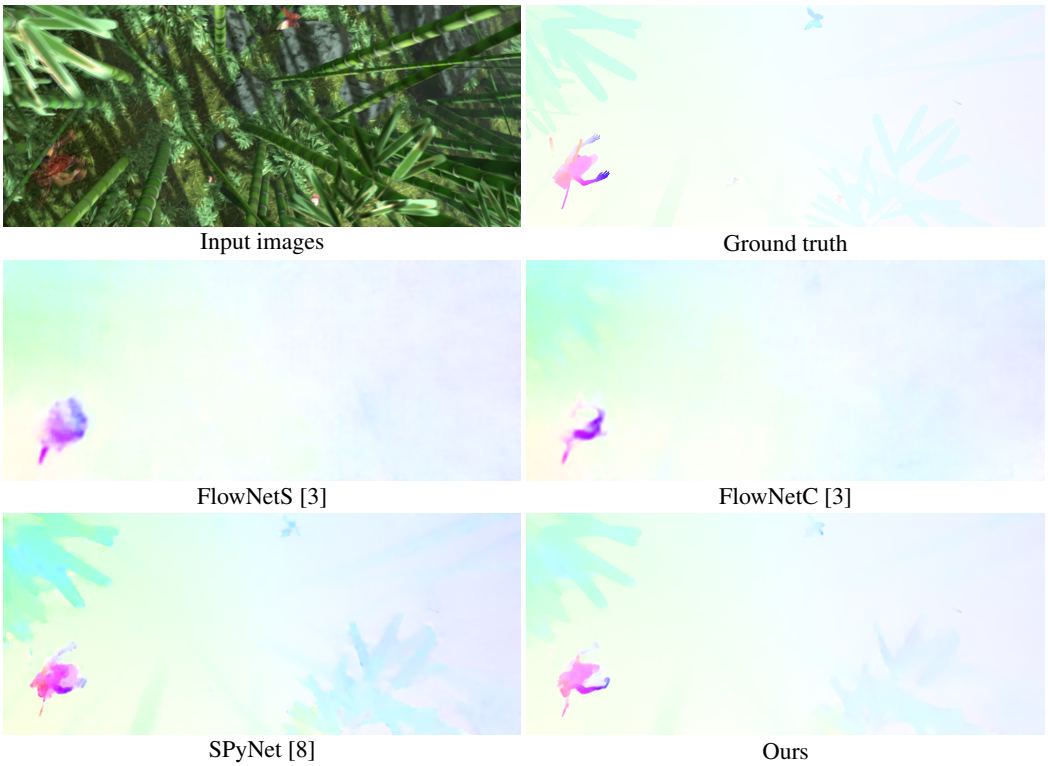


Figure 7: **Visual comparisons on the Sintel dataset.**

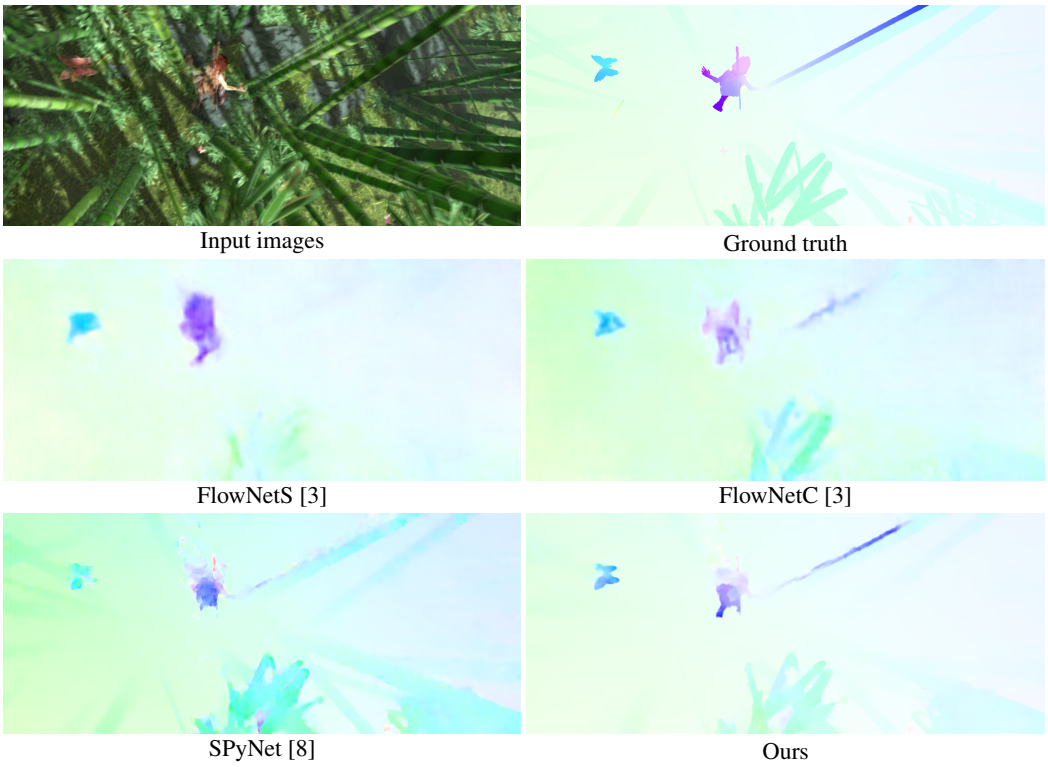


Figure 8: **Visual comparisons on the Sintel dataset.**

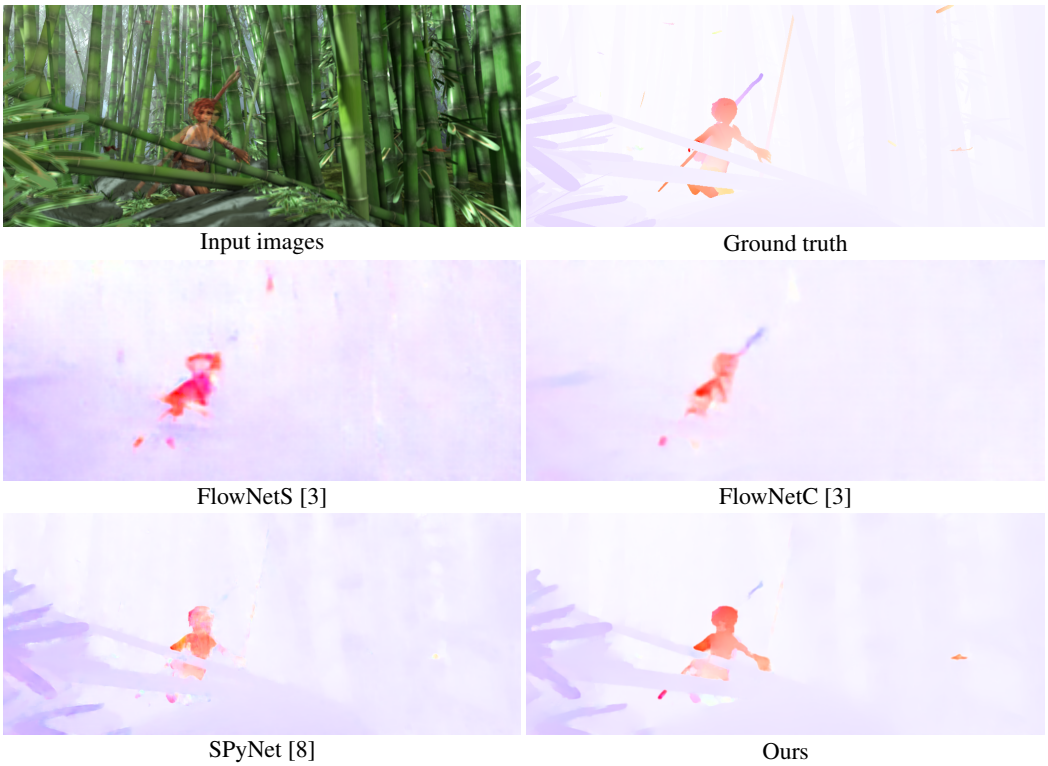


Figure 9: **Visual comparisons on the Sintel dataset.**

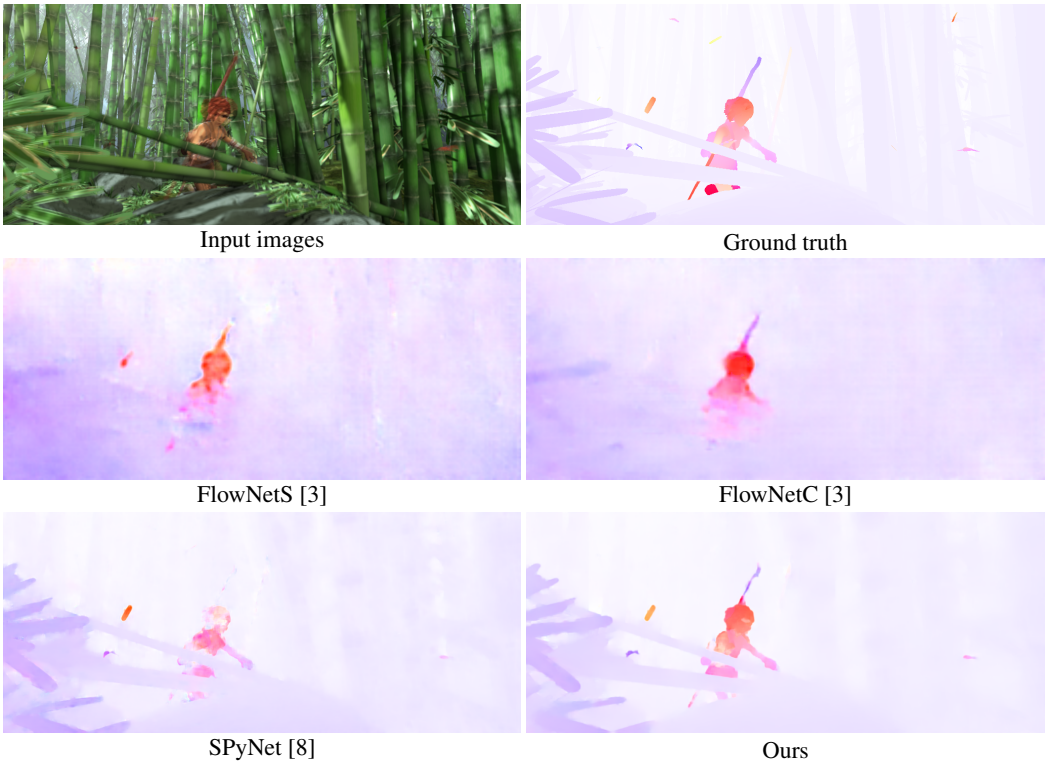


Figure 10: **Visual comparisons on the Sintel dataset.**

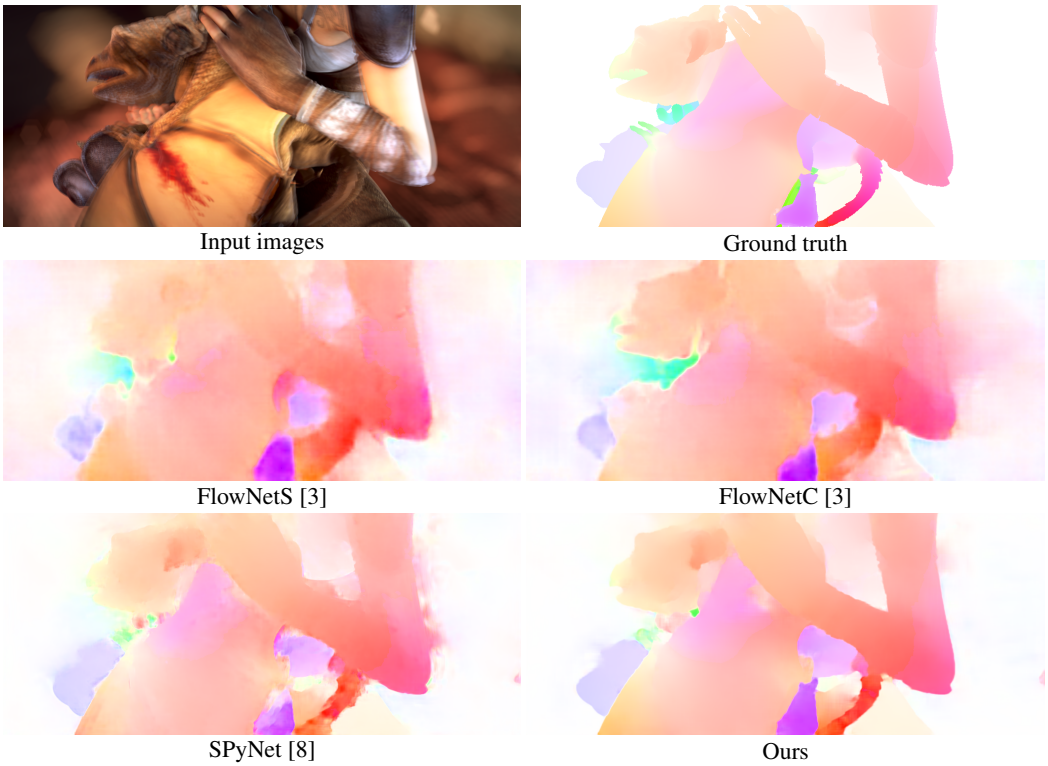


Figure 11: **Visual comparisons on the Sintel dataset.**

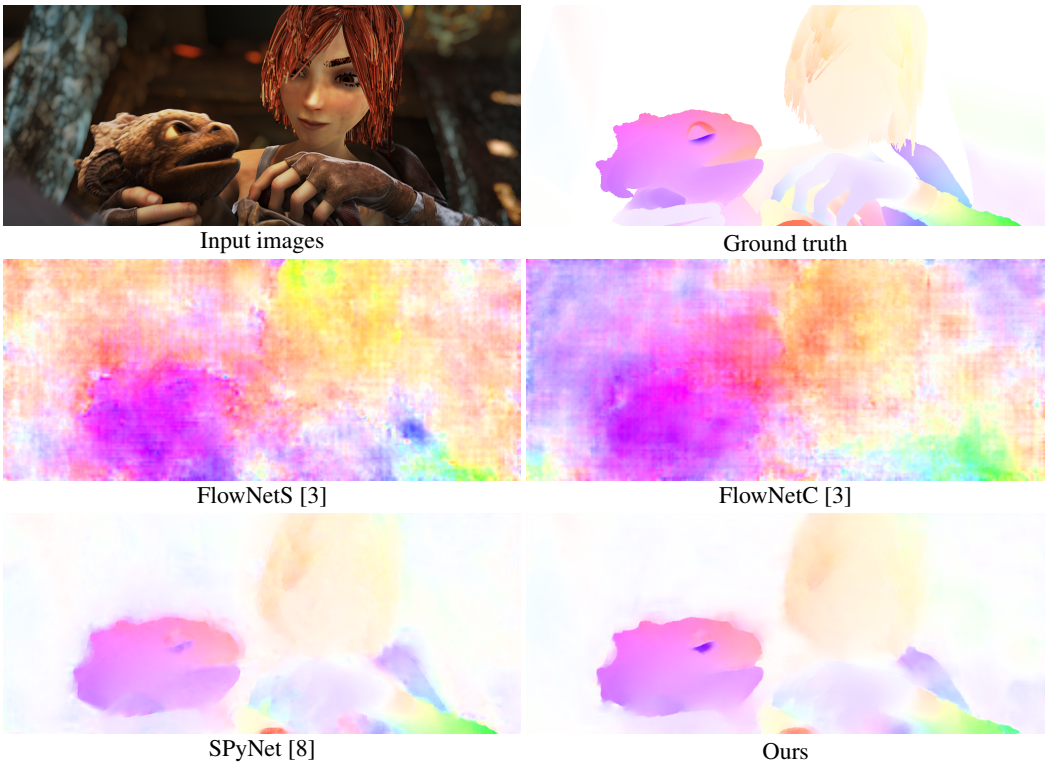


Figure 12: **Visual comparisons on the Sintel dataset.**

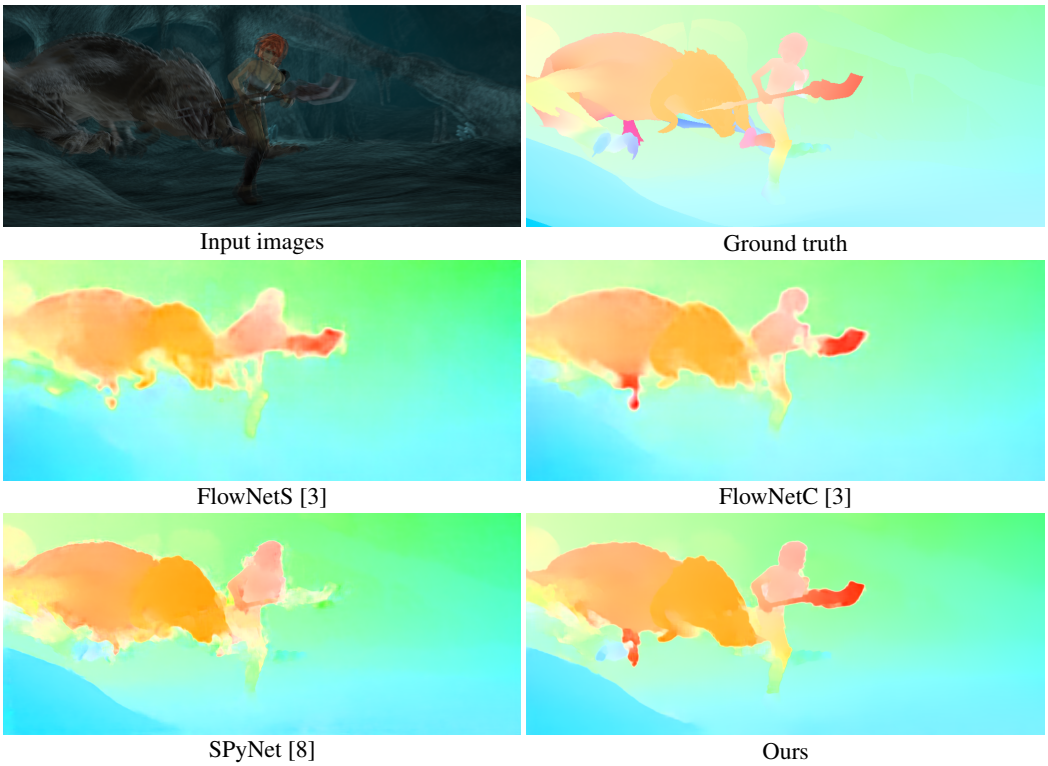


Figure 13: **Visual comparisons on the Sintel dataset.**

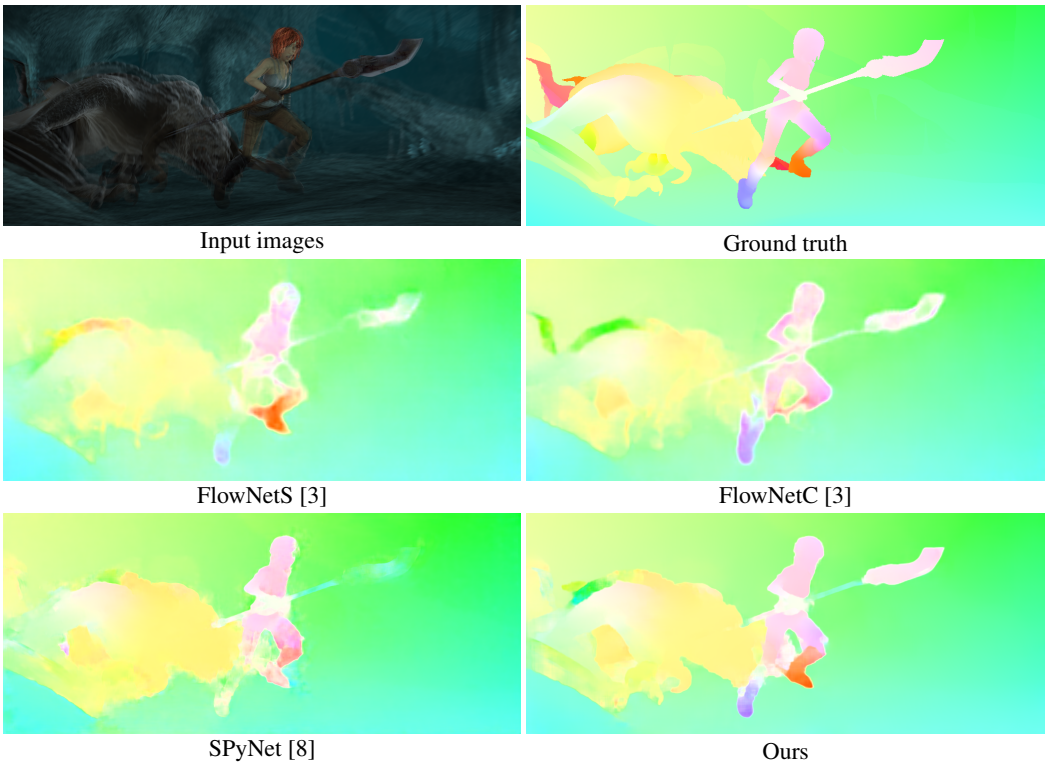


Figure 14: **Visual comparisons on the Sintel dataset.**

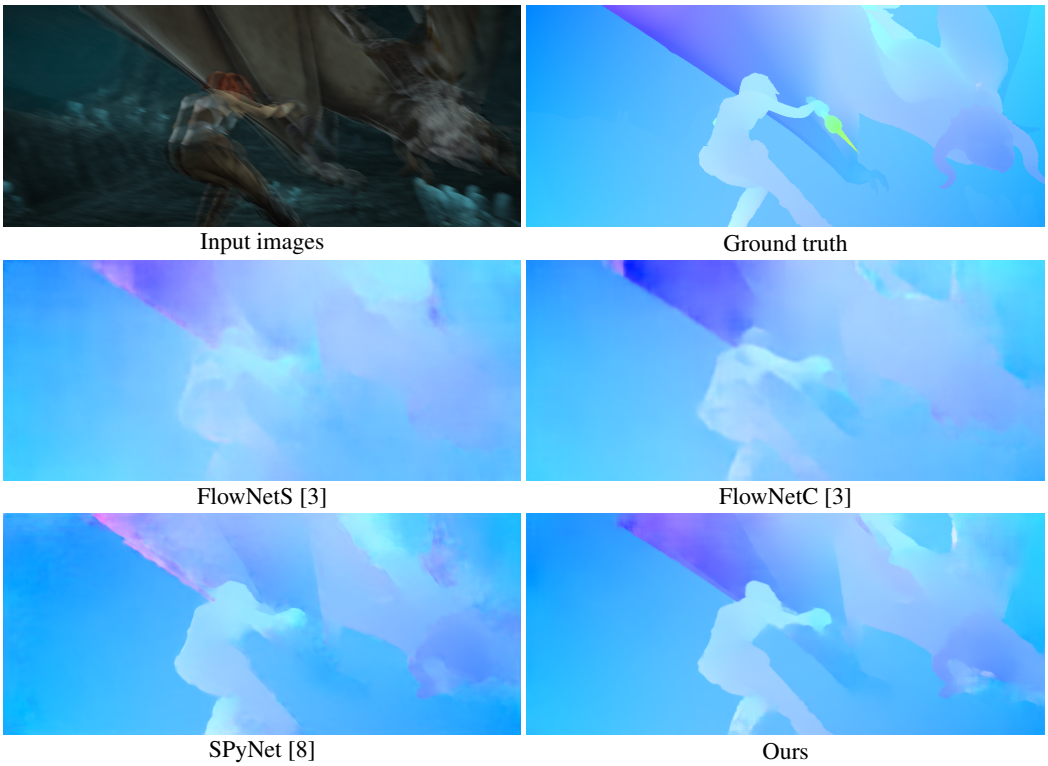


Figure 15: **Visual comparisons on the Sintel dataset.**

Figure 16: **Visual comparisons on the Sintel dataset.**

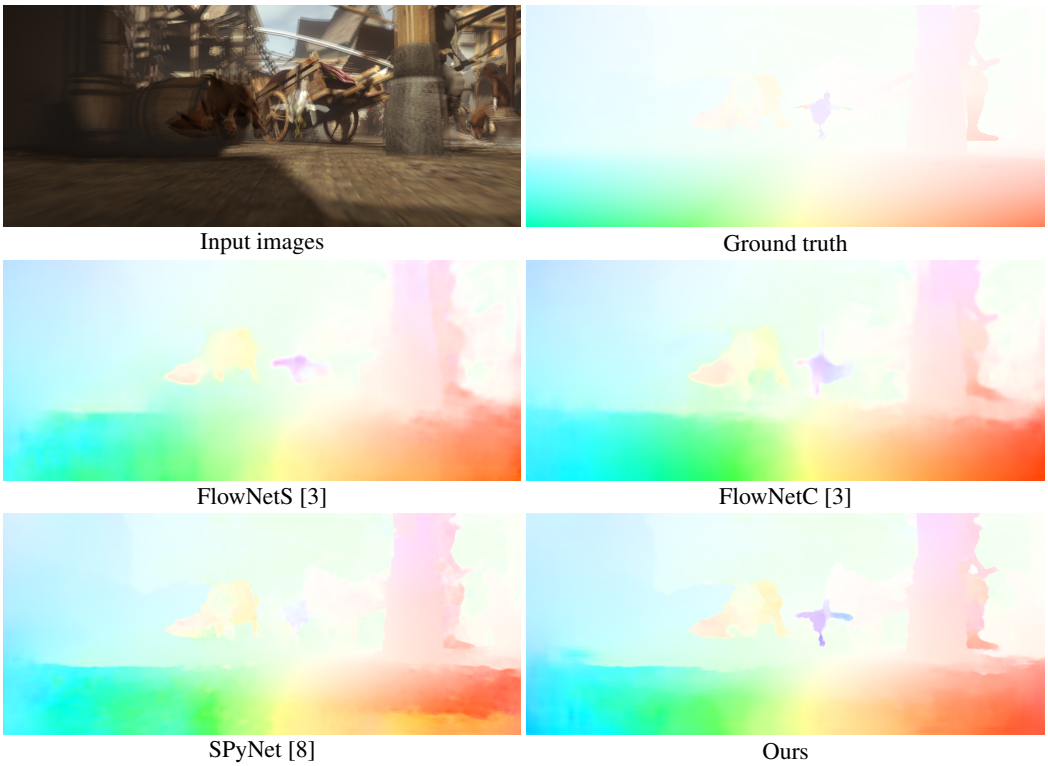


Figure 17: **Visual comparisons on the Sintel dataset.**

4.2 Visual comparisons on the KITTI 2012 training set

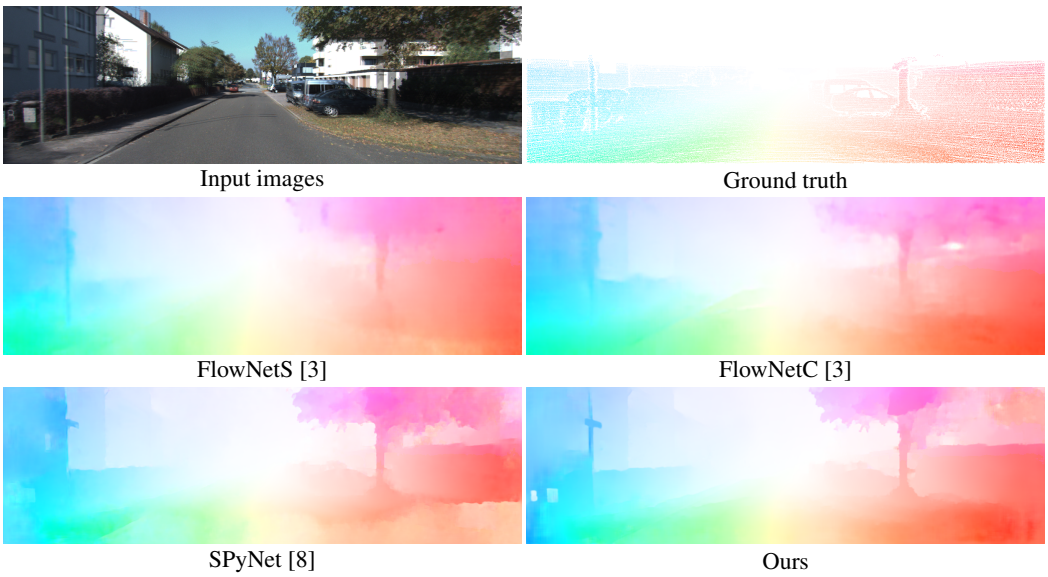


Figure 18: **Visual comparisons on the KITTI 2012 dataset.**

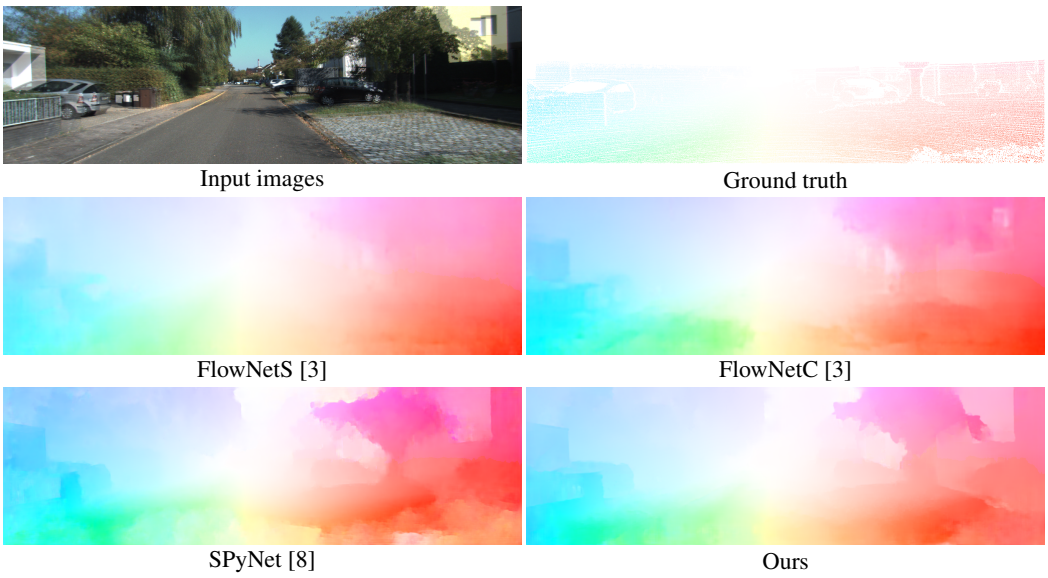


Figure 19: **Visual comparisons on the KITTI 2012 dataset.**

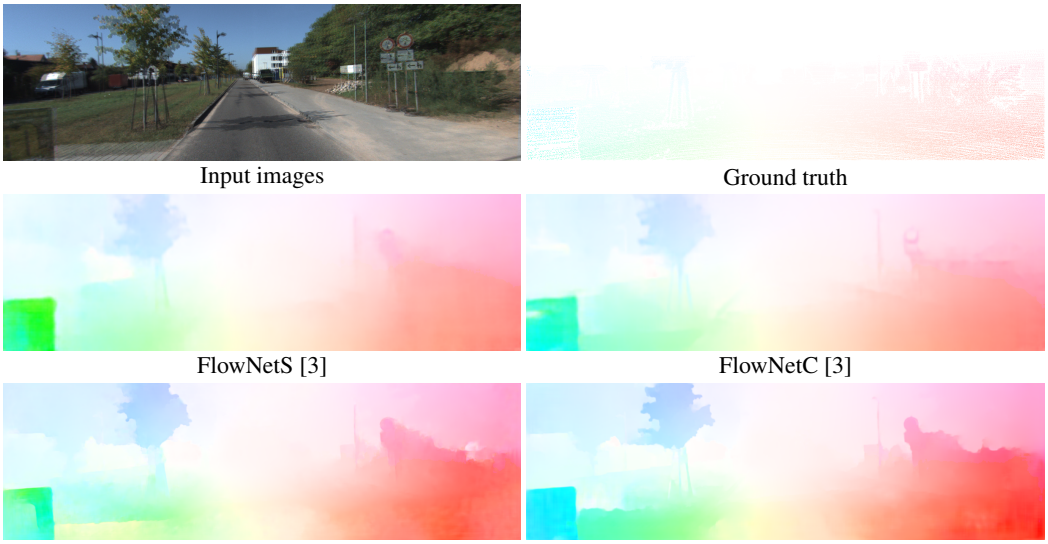


Figure 20: **Visual comparisons on the KITTI 2012 dataset.**

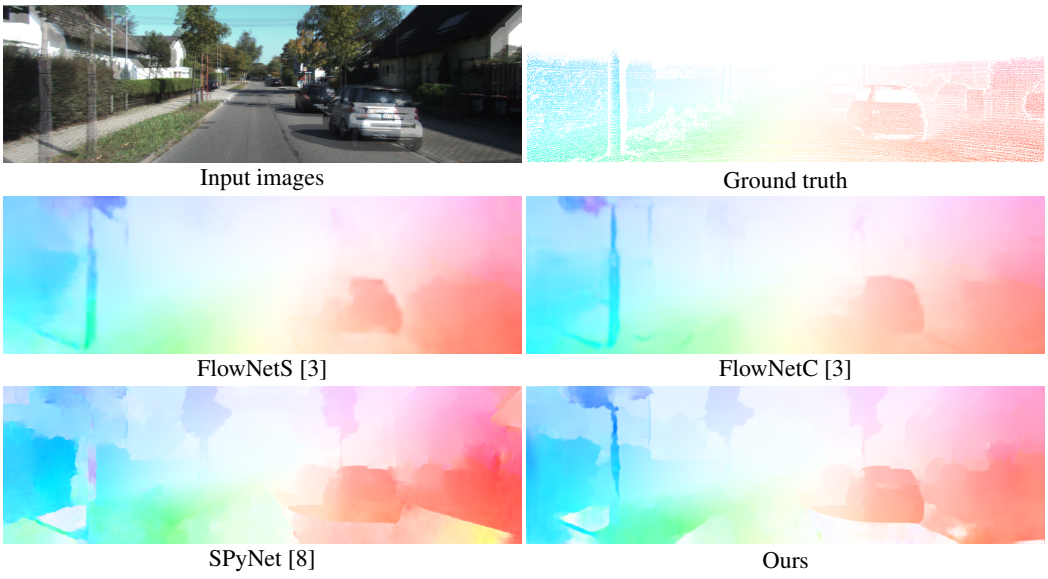


Figure 21: **Visual comparisons on the KITTI 2012 dataset.**

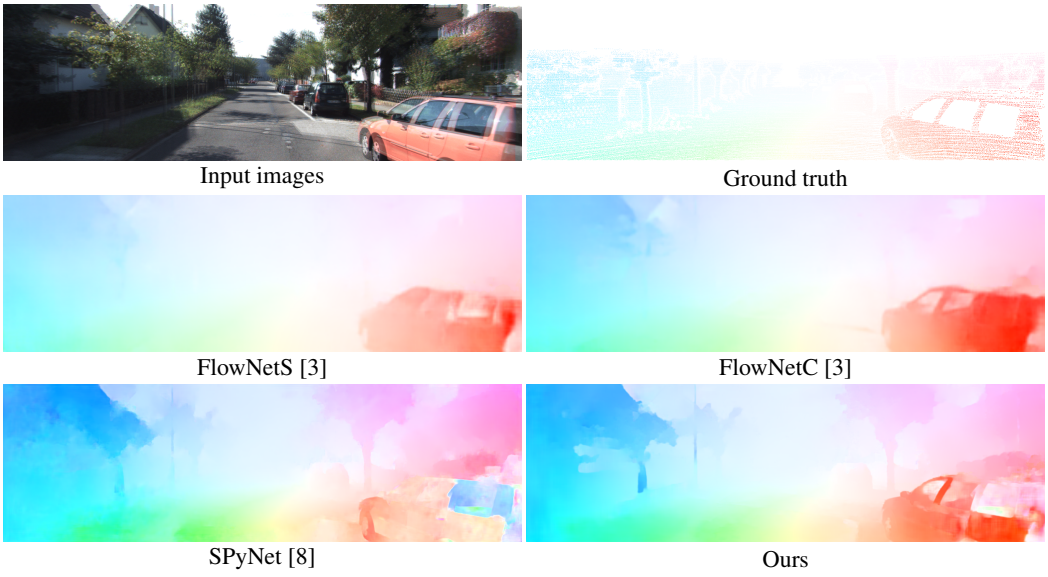


Figure 22: **Visual comparisons on the KITTI 2012 dataset.**

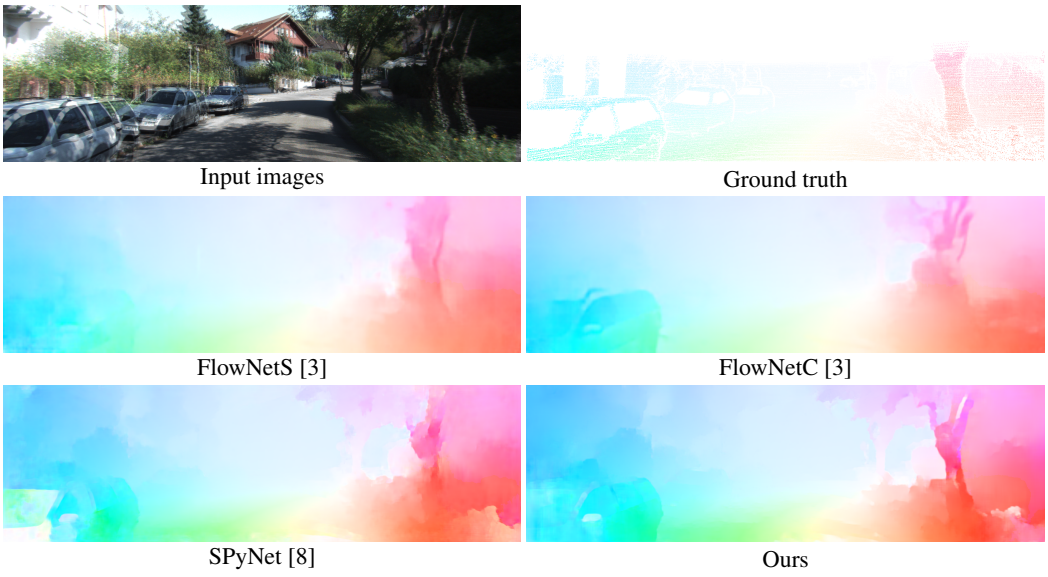


Figure 23: **Visual comparisons on the KITTI 2012 dataset.**

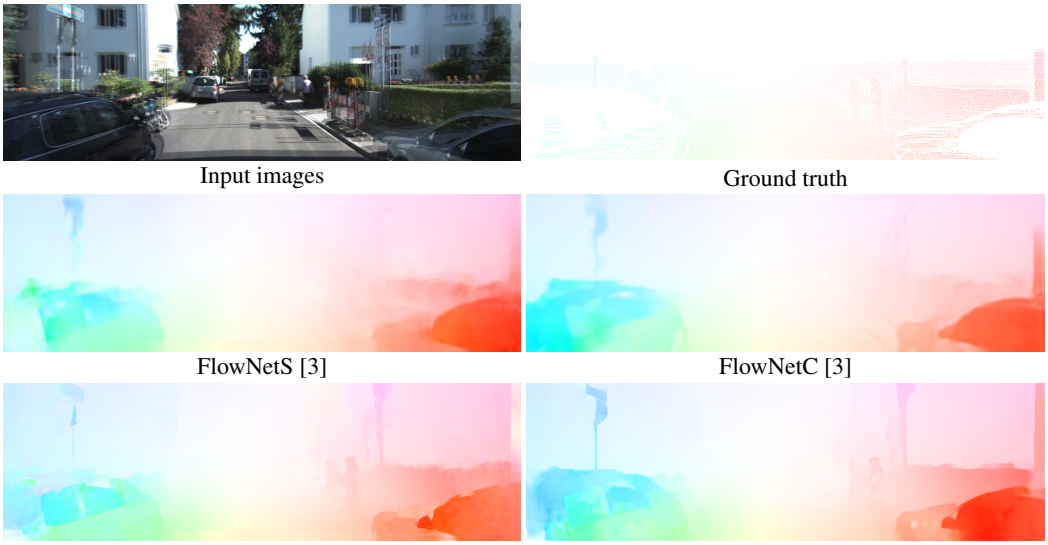


Figure 24: **Visual comparisons on the KITTI 2012 dataset.**

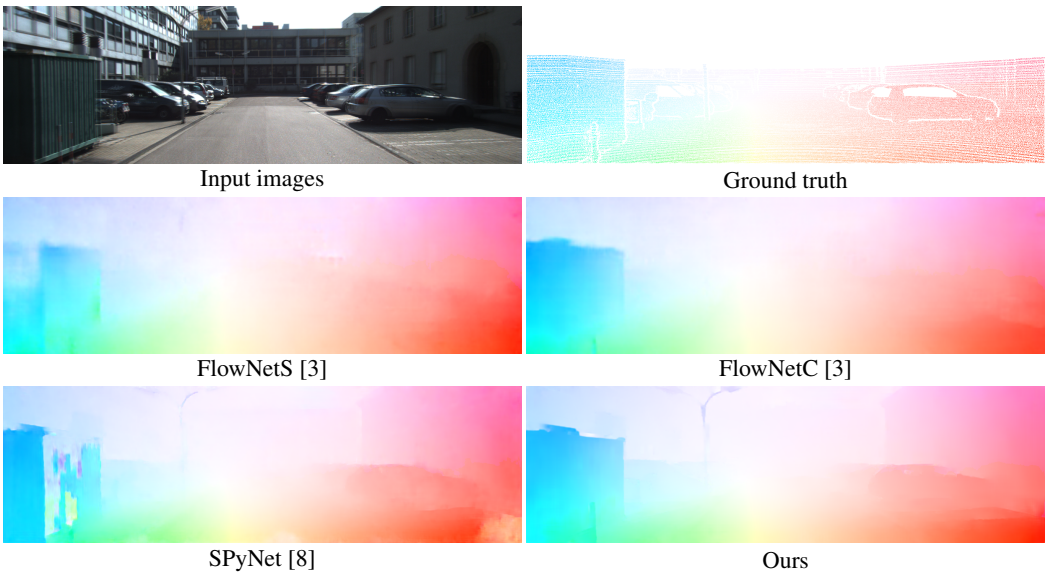


Figure 25: **Visual comparisons on the KITTI 2012 dataset.**

4.3 Visual comparisons on the KITTI 2015 training set

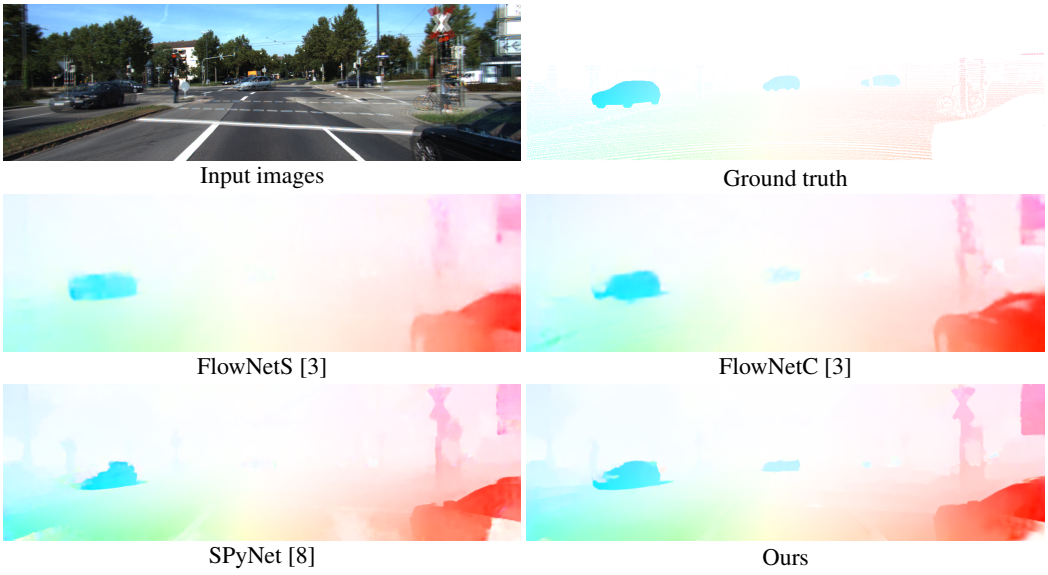


Figure 26: **Visual comparisons on the KITTI 2015 dataset.**

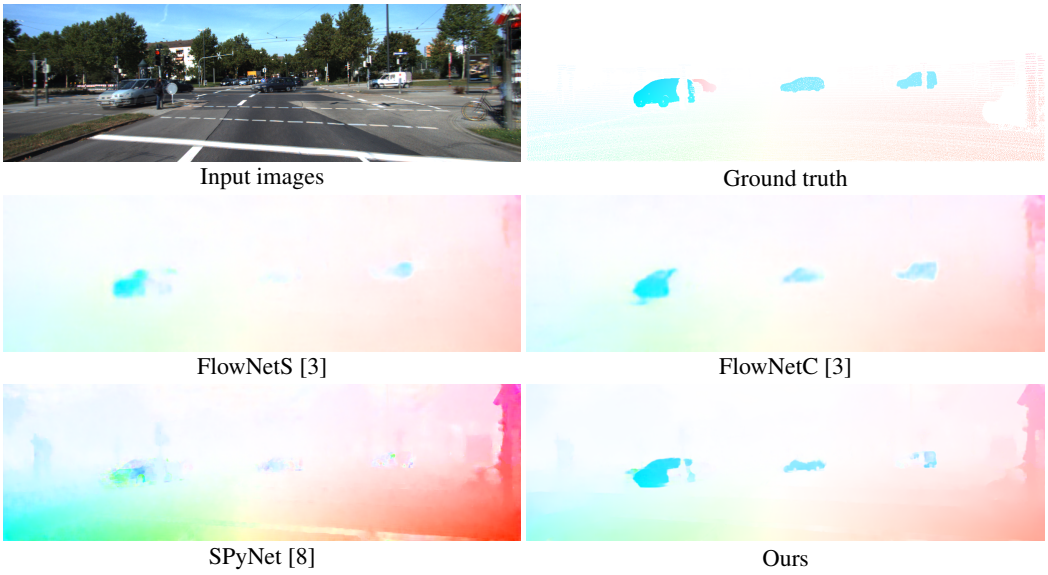


Figure 27: **Visual comparisons on the KITTI 2015 dataset.**

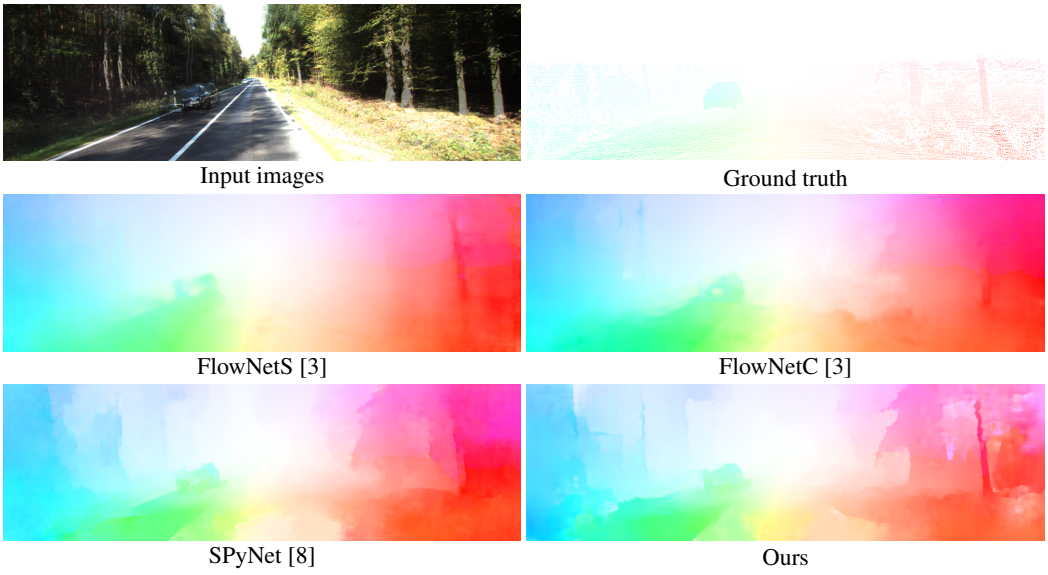


Figure 28: **Visual comparisons on the KITTI 2015 dataset.**

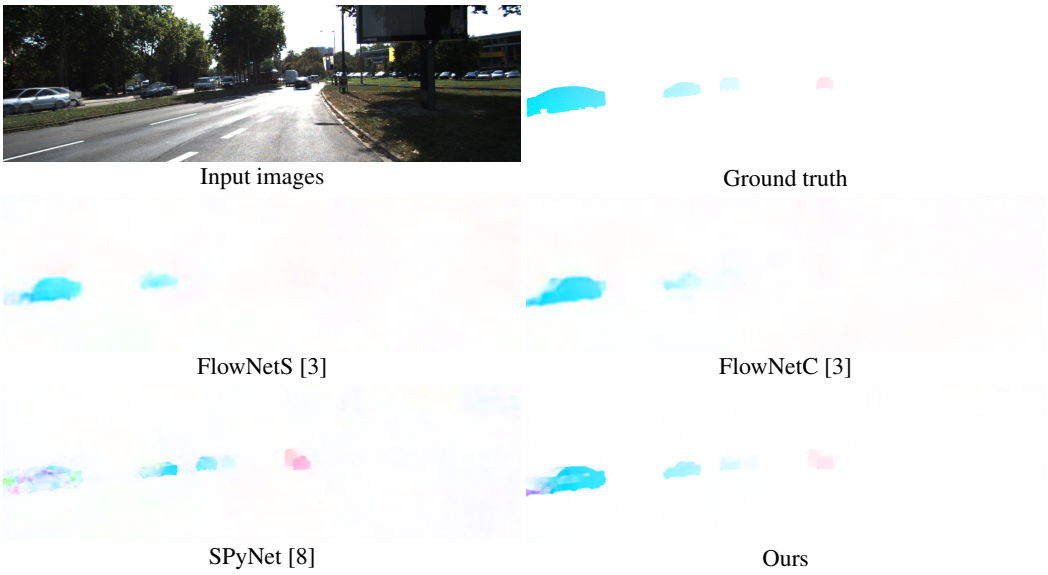


Figure 29: **Visual comparisons on the KITTI 2015 dataset.**

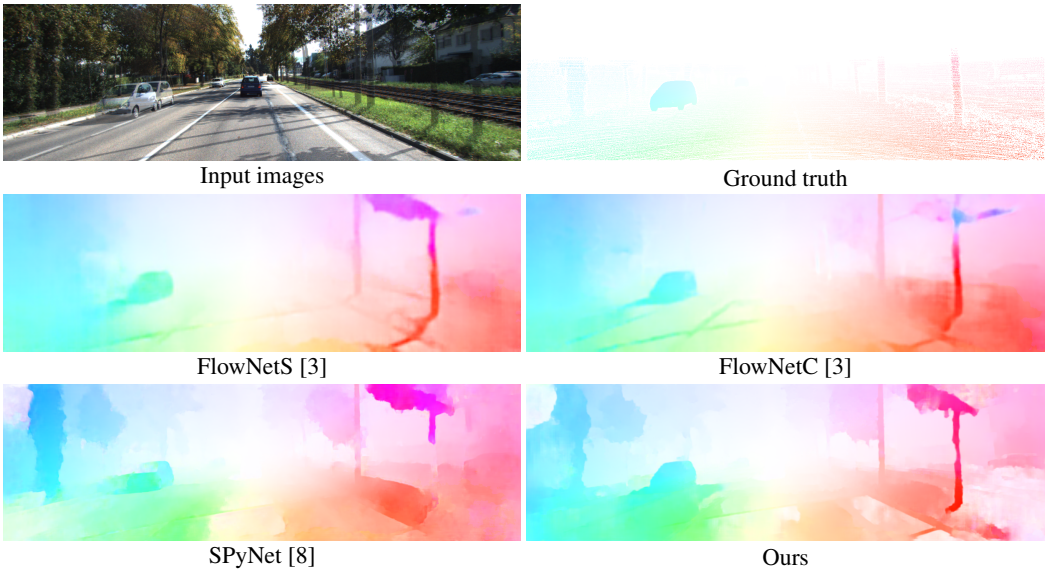


Figure 30: **Visual comparisons on the KITTI 2015 dataset.**

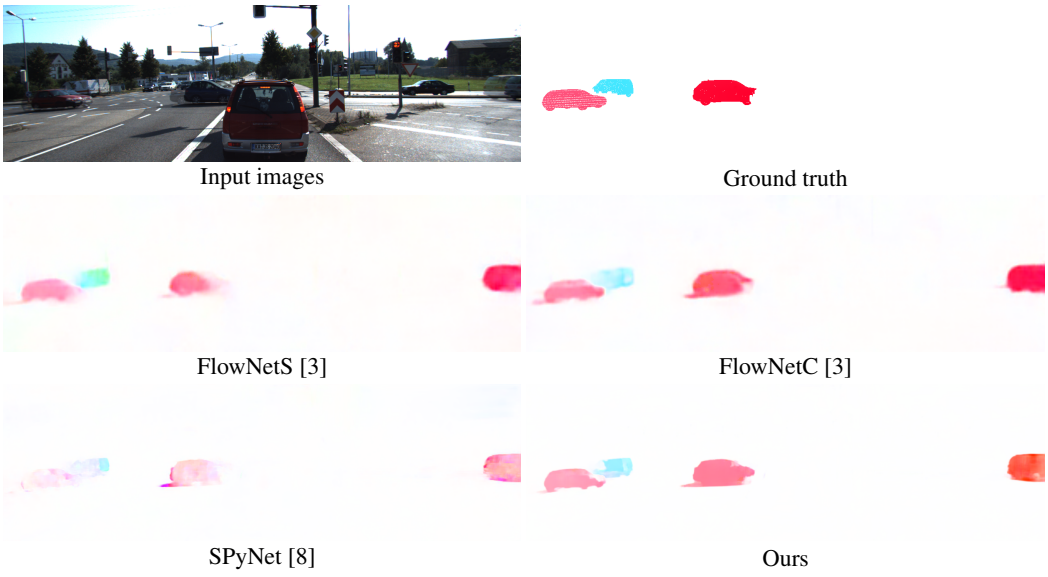


Figure 31: **Visual comparisons on the KITTI 2015 dataset.**

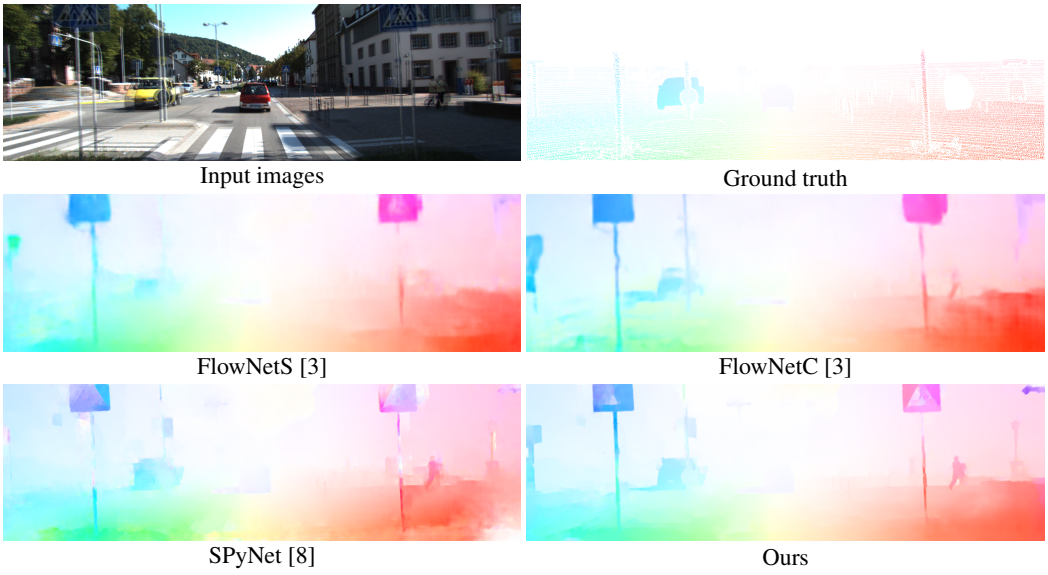


Figure 32: **Visual comparisons on the KITTI 2015 dataset.**

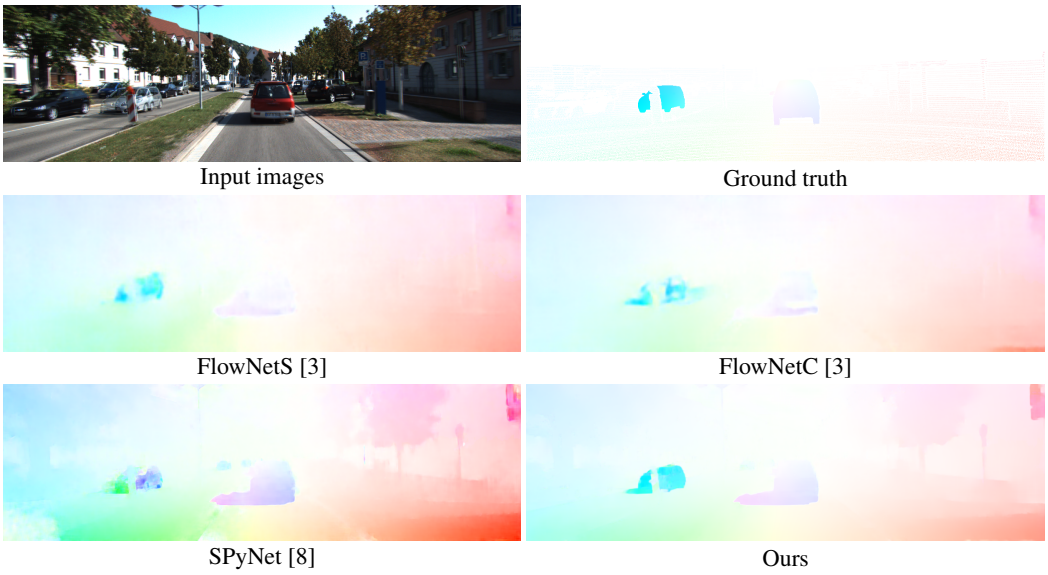


Figure 33: **Visual comparisons on the KITTI 2015 dataset.**

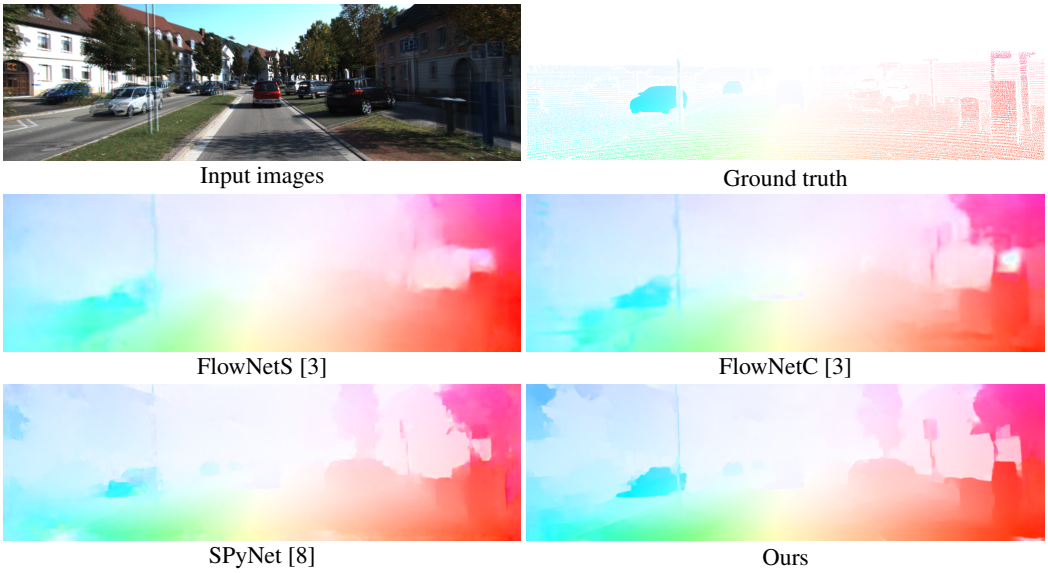


Figure 34: **Visual comparisons on the KITTI 2015 dataset.**

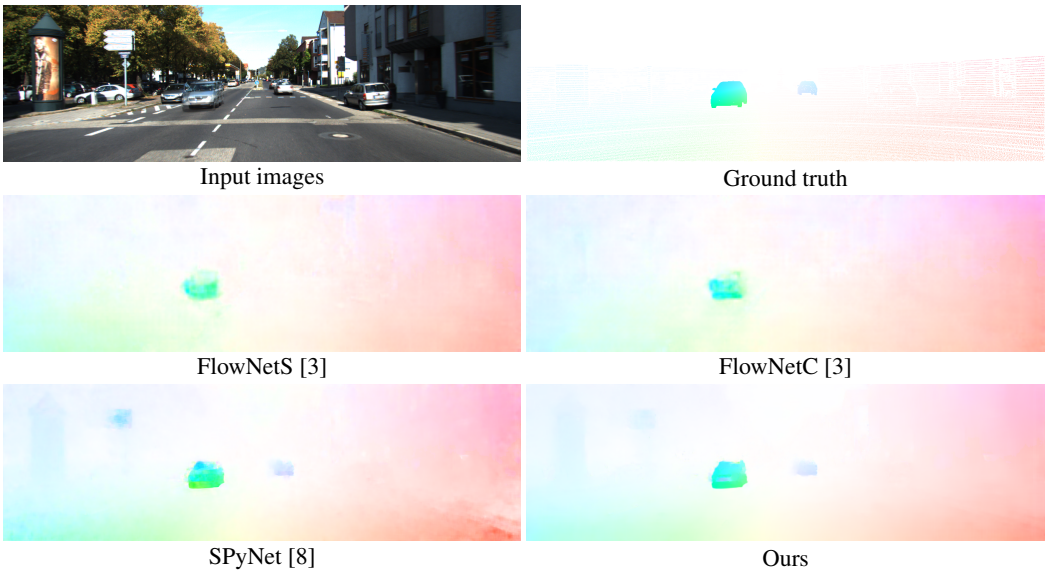


Figure 35: **Visual comparisons on the KITTI 2015 dataset.**

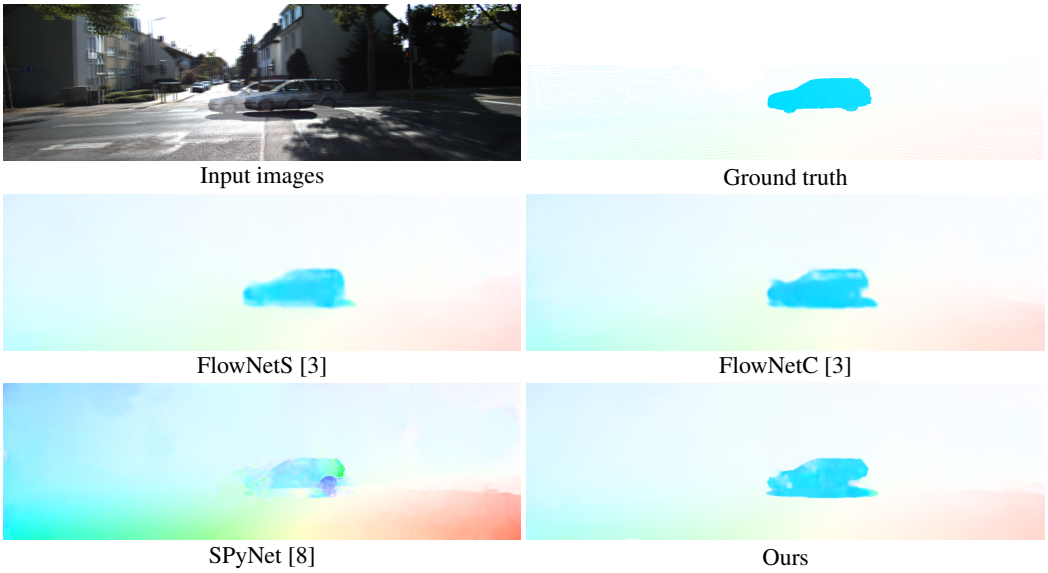


Figure 36: **Visual comparisons on the KITTI 2015 dataset.**

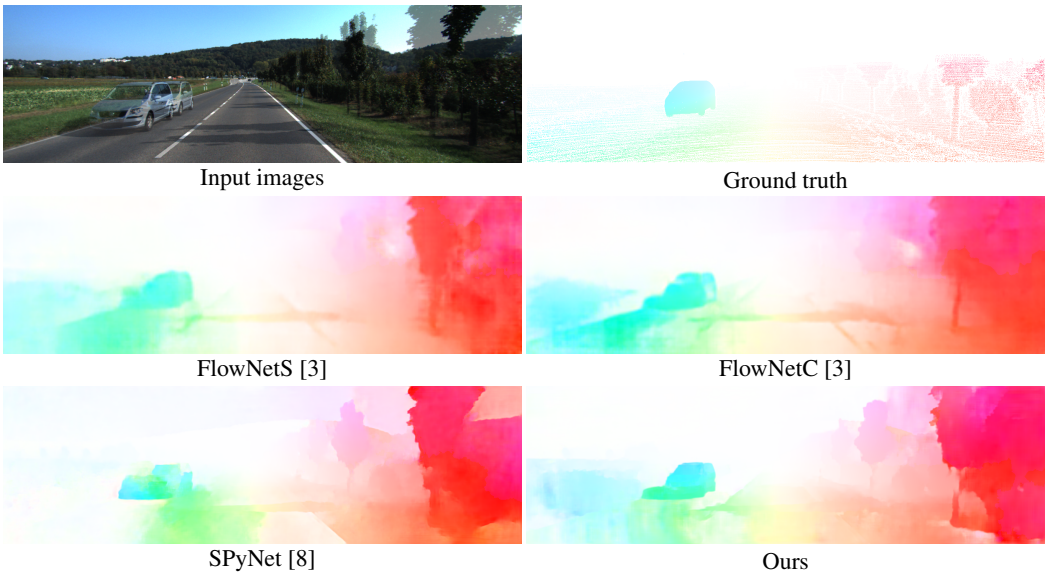


Figure 37: **Visual comparisons on the KITTI 2015 dataset.**

4.4 Visual comparisons on the FlyingChairs test set

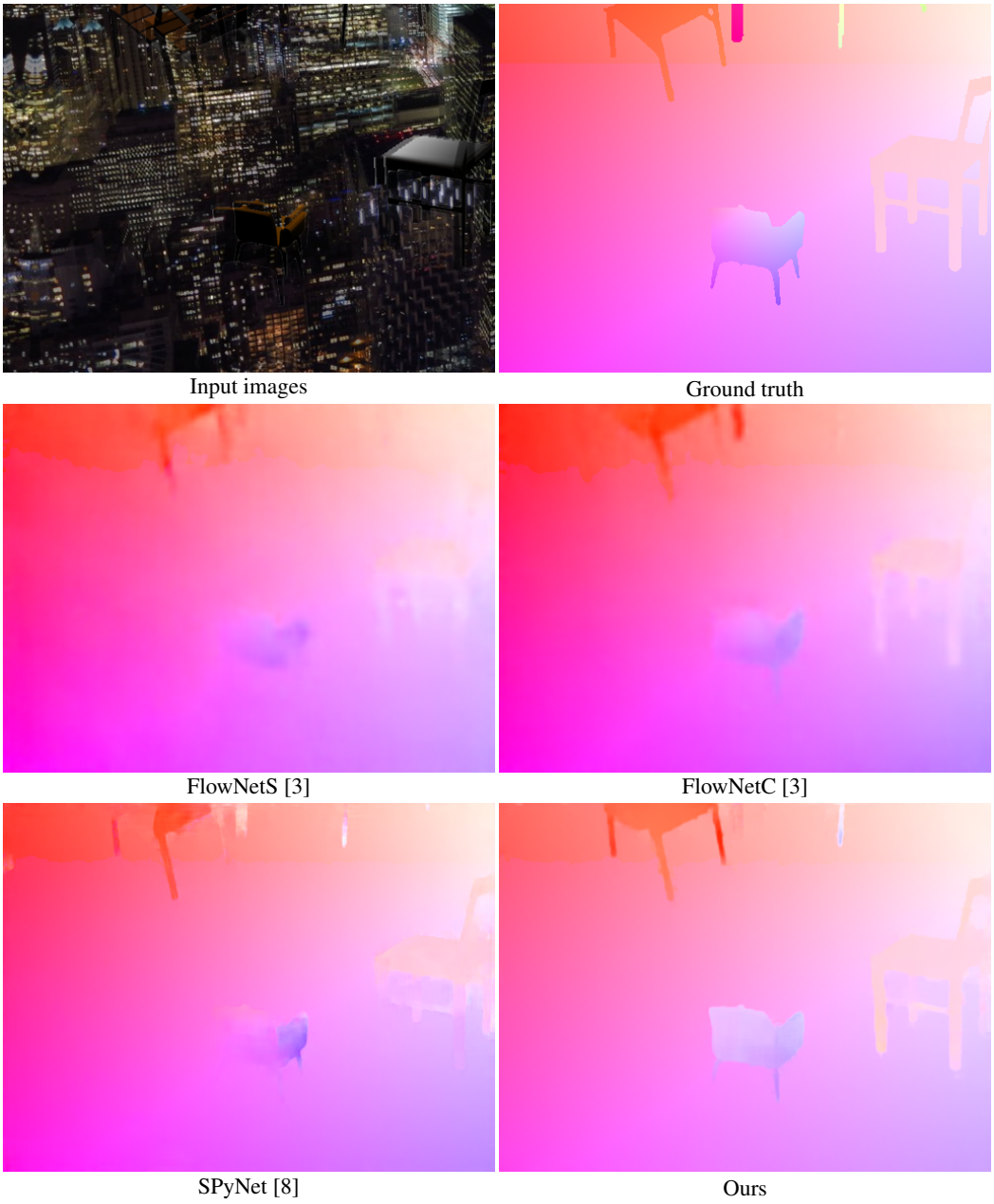
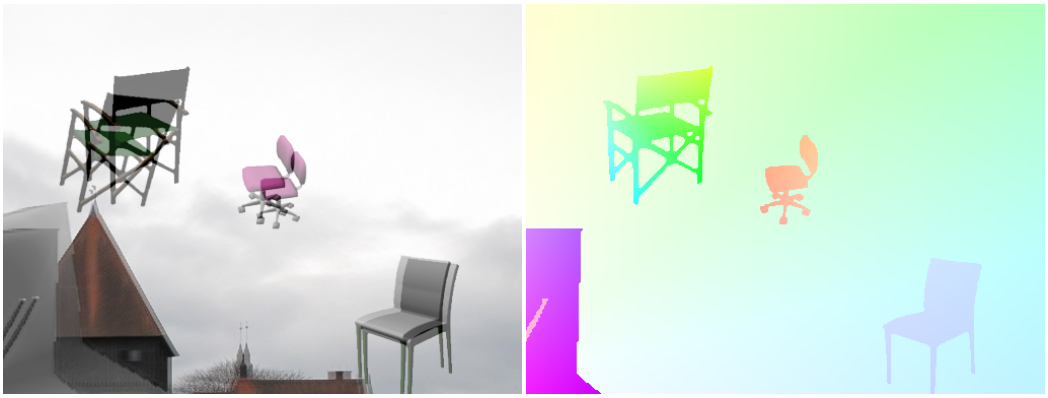
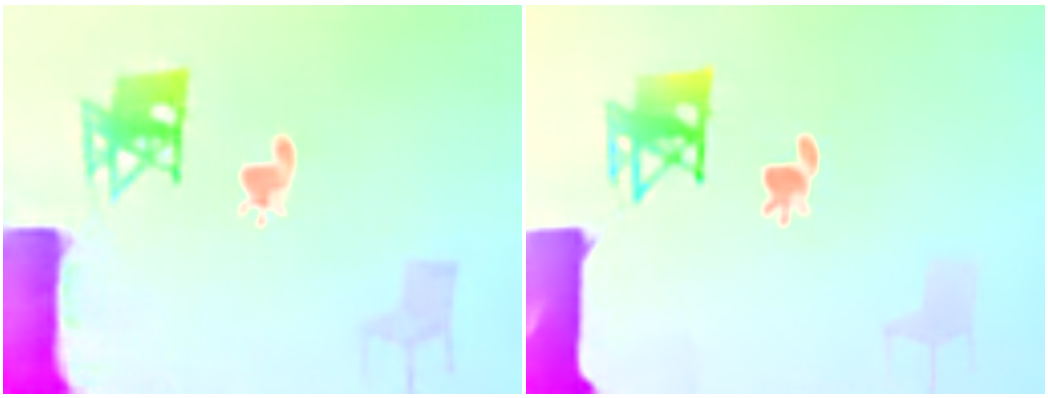


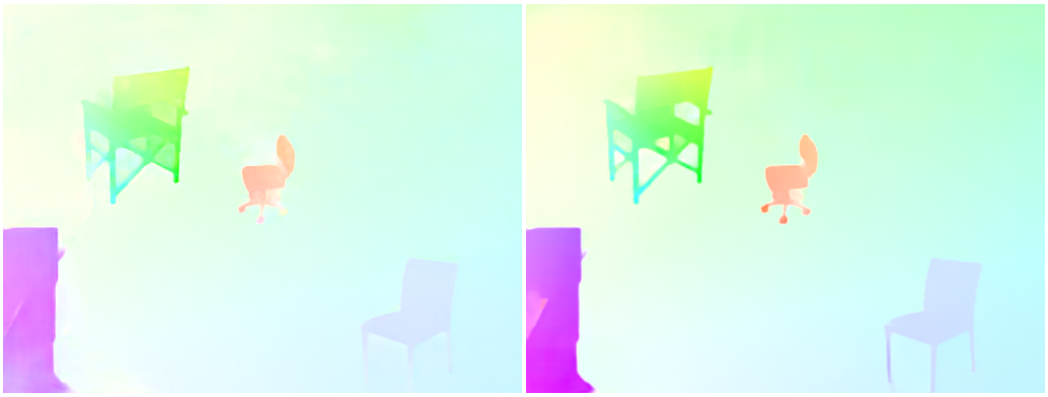
Figure 38: **Visual comparisons on the FlyingChairs dataset.**



Input images | Ground truth



FlowNetS [3] | FlowNetC [3]



SPyNet [8] | Ours

Figure 39: **Visual comparisons on the FlyingChairs dataset.**

Input images | Ground truth

FlowNetS [3] | FlowNetC [3]

SPyNet [8] | Ours

Figure 40: **Visual comparisons on the FlyingChairs dataset.**


Input images

Ground truth


FlowNetS [3]


FlowNetC [3]


SPyNet [8]


Ours

Figure 41: **Visual comparisons on the FlyingChairs dataset.**

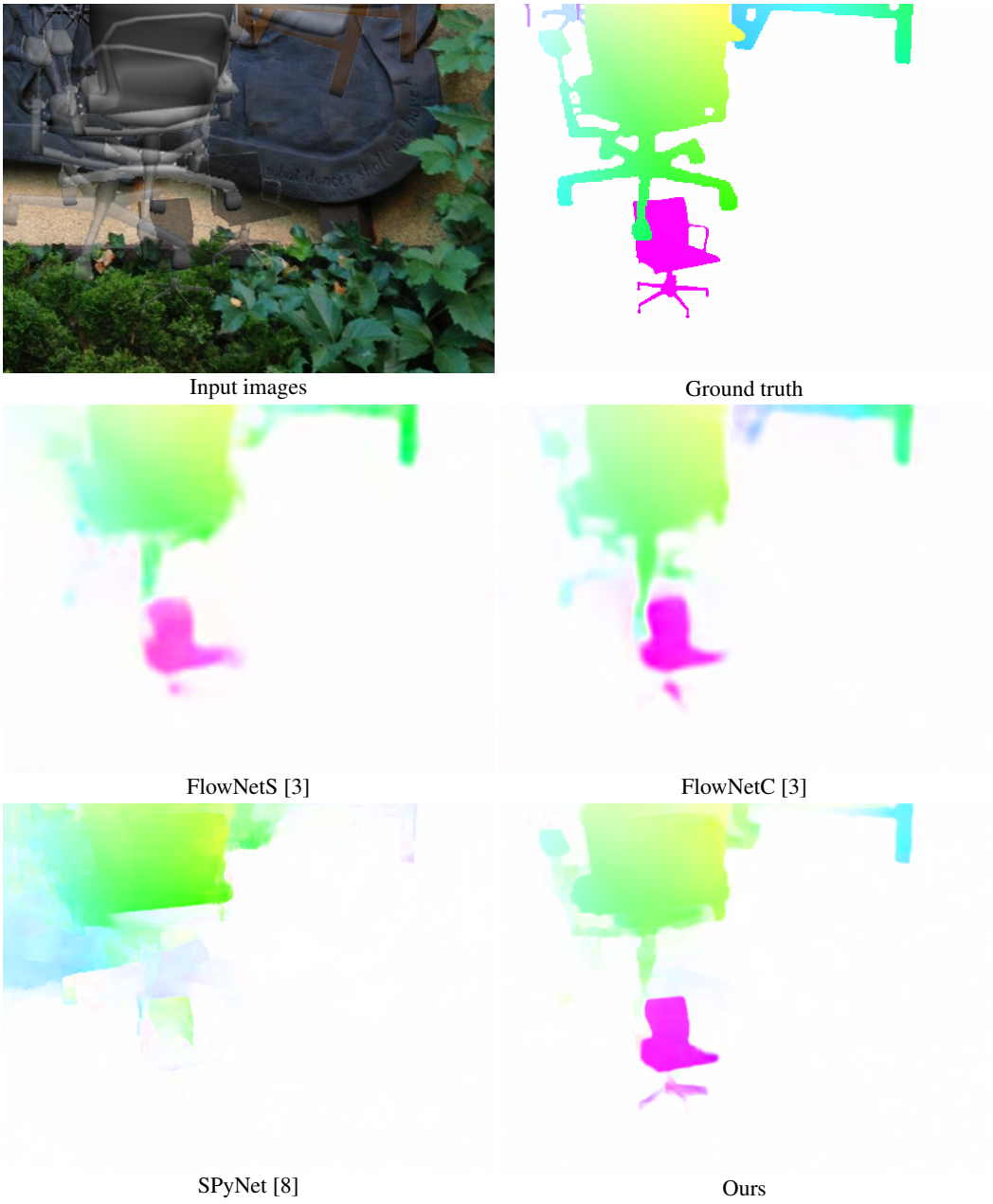


Figure 42: **Visual comparisons on the FlyingChairs dataset.**


Input images

Ground truth


FlowNetS [3]


FlowNetC [3]


SPyNet [8]

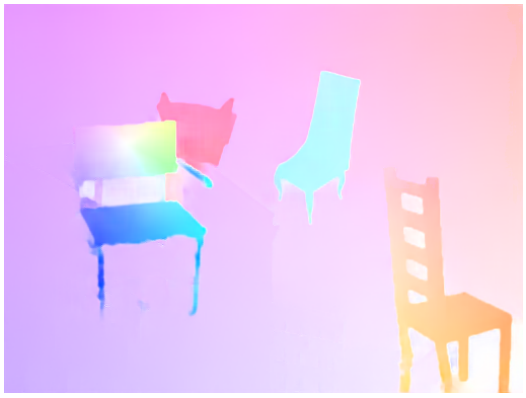
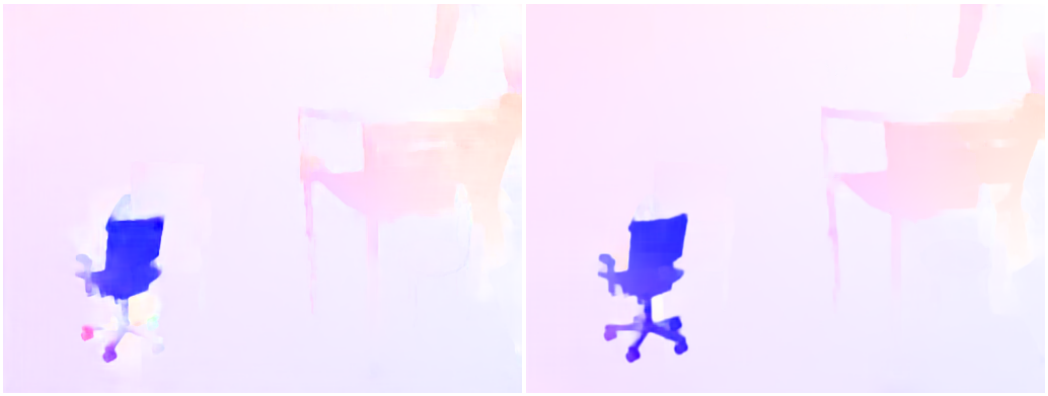

Ours

Figure 43: **Visual comparisons on the FlyingChairs dataset.**

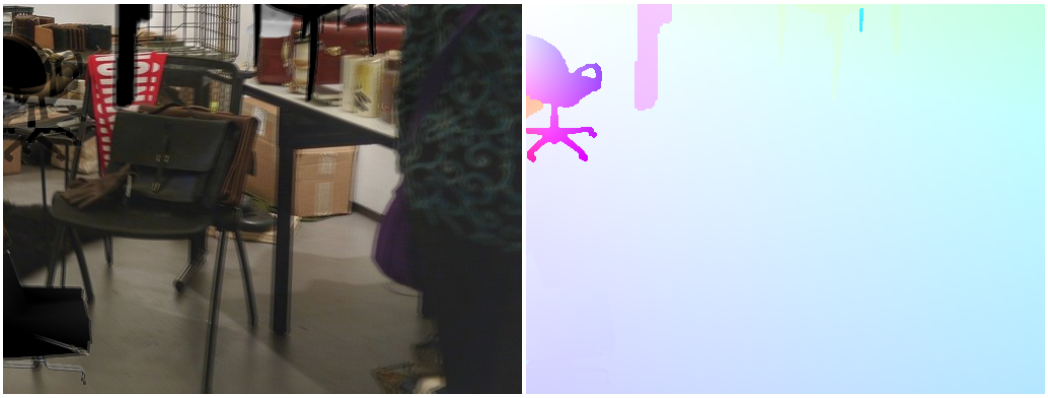
Input images | Ground truth

FlowNetS [3] | FlowNetC [3]



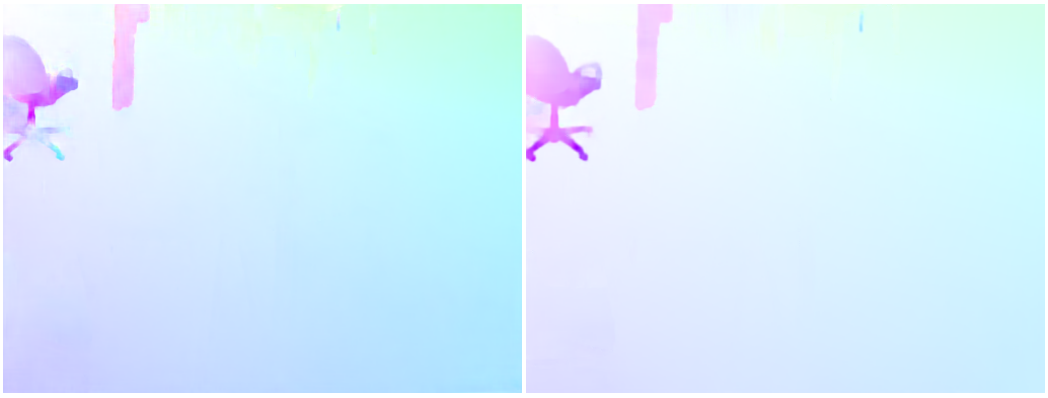
SPyNet [8] | Ours

Figure 44: **Visual comparisons on the FlyingChairs dataset.**



Input images | Ground truth

FlowNetS [3] | FlowNetC [3]



SPyNet [8] | Ours

Figure 45: **Visual comparisons on the FlyingChairs dataset.**

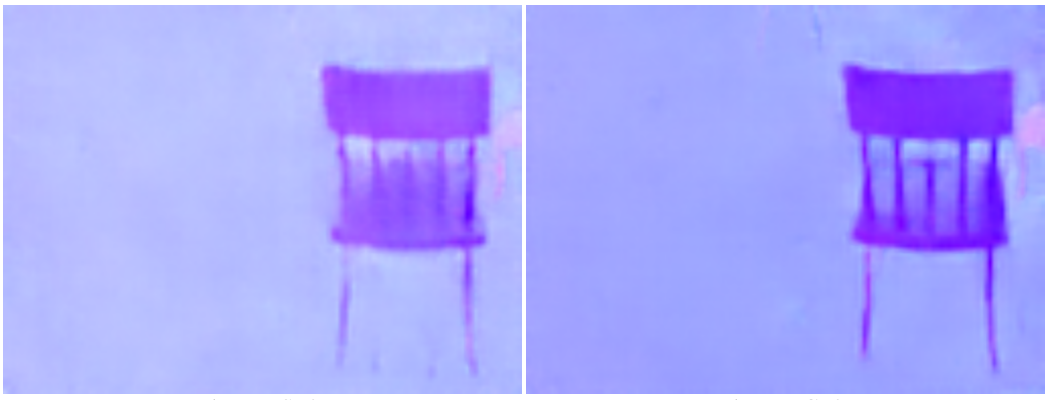
Input images | Ground truth

FlowNetS [3] | FlowNetC [3]

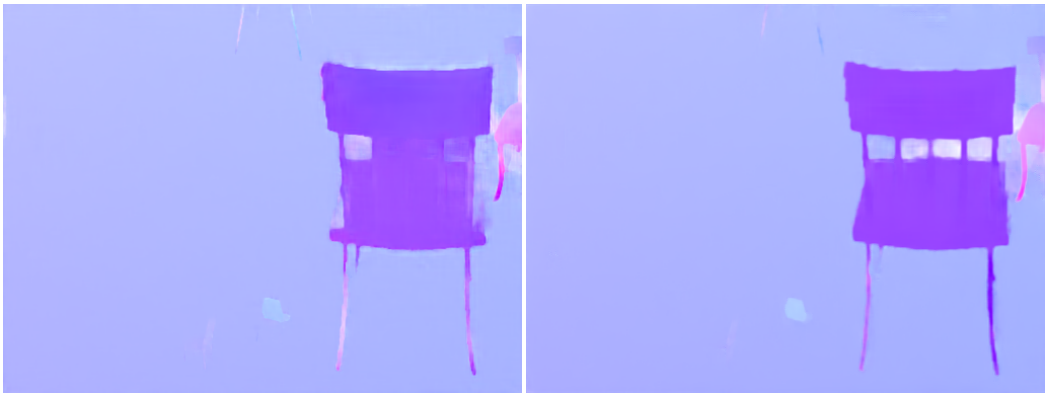
SPyNet [8] | Ours

Figure 46: **Visual comparisons on the FlyingChairs dataset.**

Input images | Ground truth



FlowNetS [3] | FlowNetC [3]

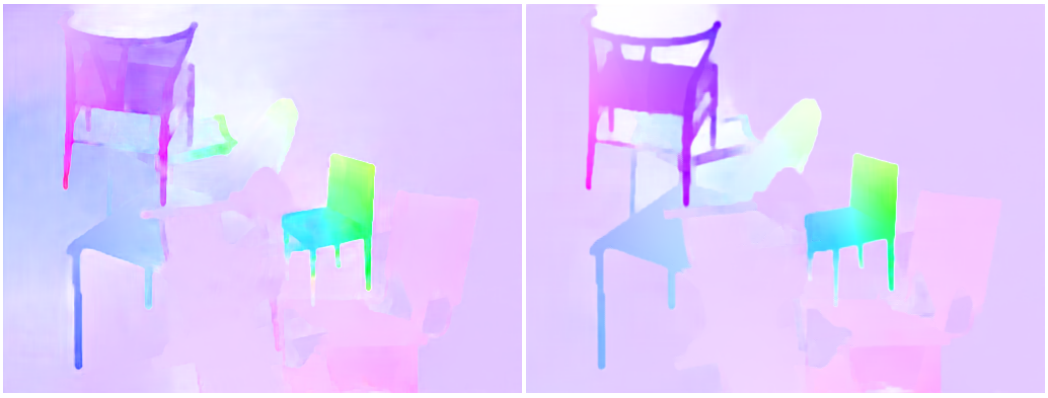


SPyNet [8] | Ours

Figure 47: **Visual comparisons on the FlyingChairs dataset.**

Input images | Ground truth

FlowNetS [3] | FlowNetC [3]



SPyNet [8] | Ours

Figure 48: **Visual comparisons on the FlyingChairs dataset.**

Input images | Ground truth

FlowNetS [3] | FlowNetC [3]

SPyNet [8] | Ours

Figure 49: **Visual comparisons on the FlyingChairs dataset.**

4.5 Visual comparisons on the Middlebury training set

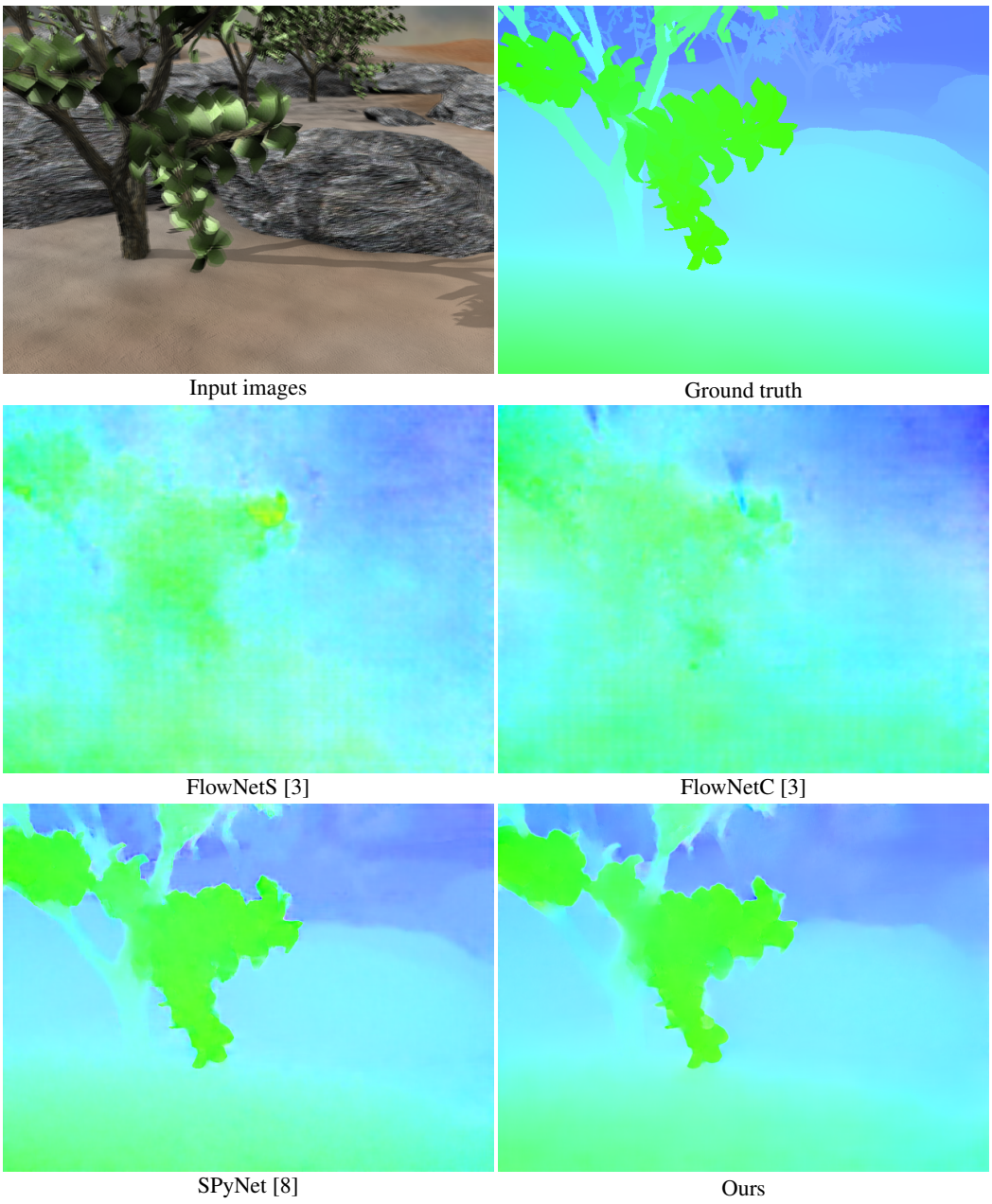


Figure 50: **Visual comparisons on the Middlebury dataset.**

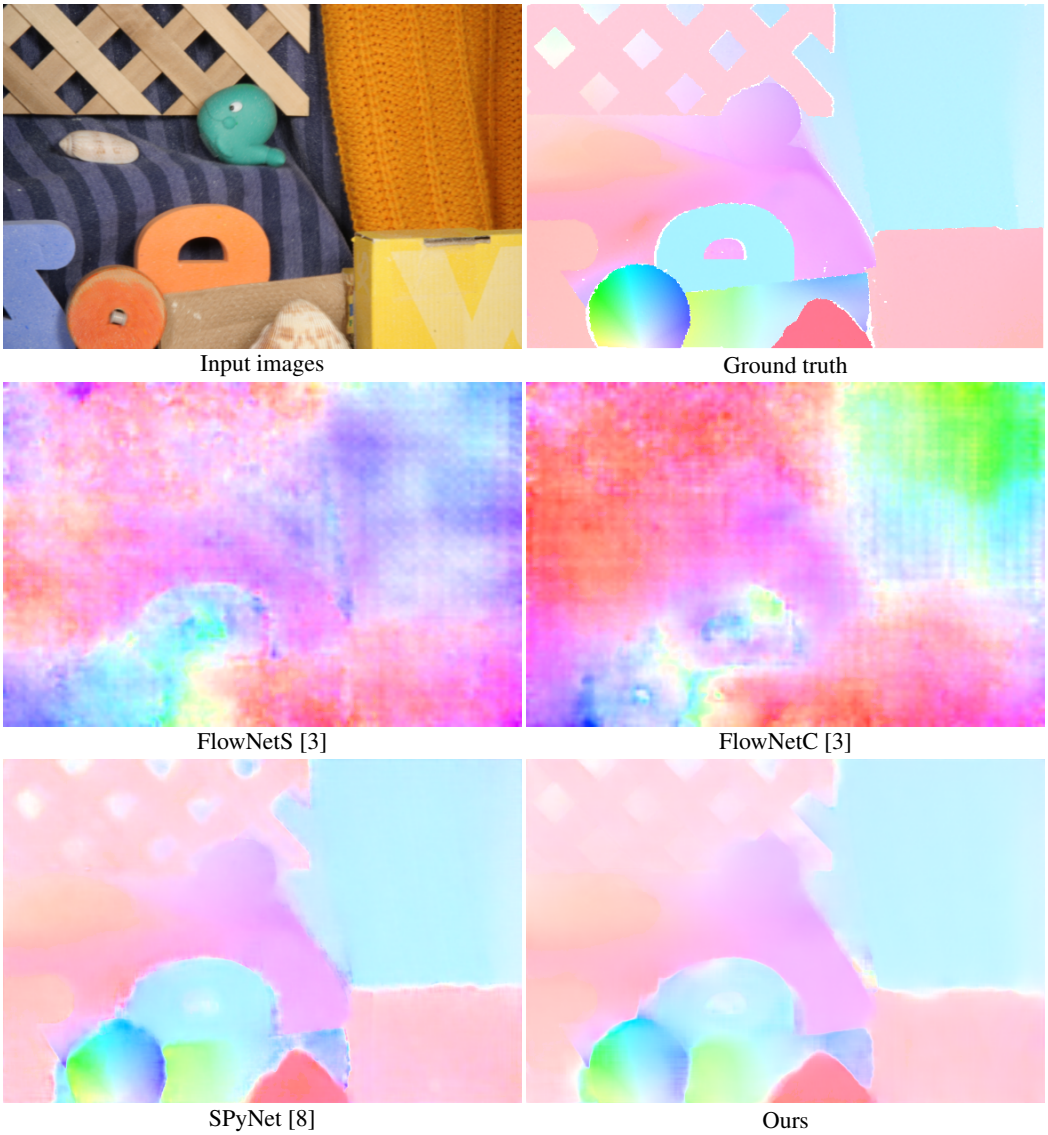
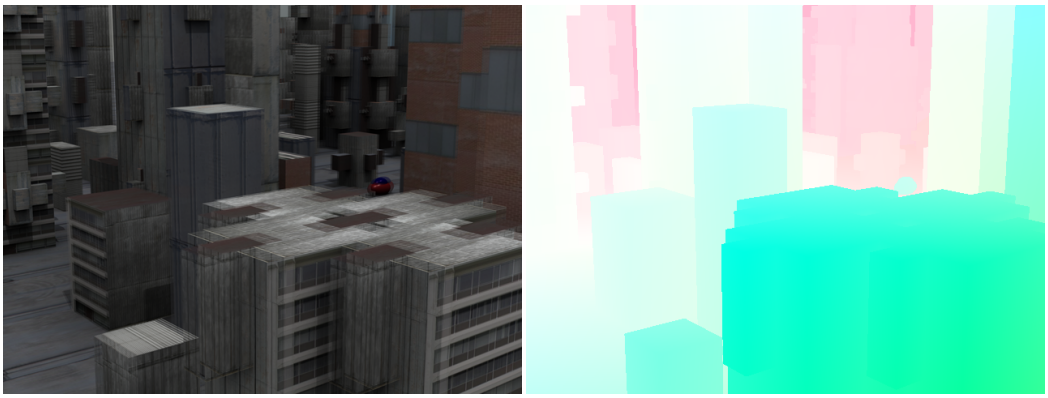


Figure 51: **Visual comparisons on the Middlebury dataset.**



Input images | Ground truth

FlowNetS [3] | FlowNetC [3]

SPyNet [8] | Ours

Figure 52: **Visual comparisons on the Middlebury dataset.**

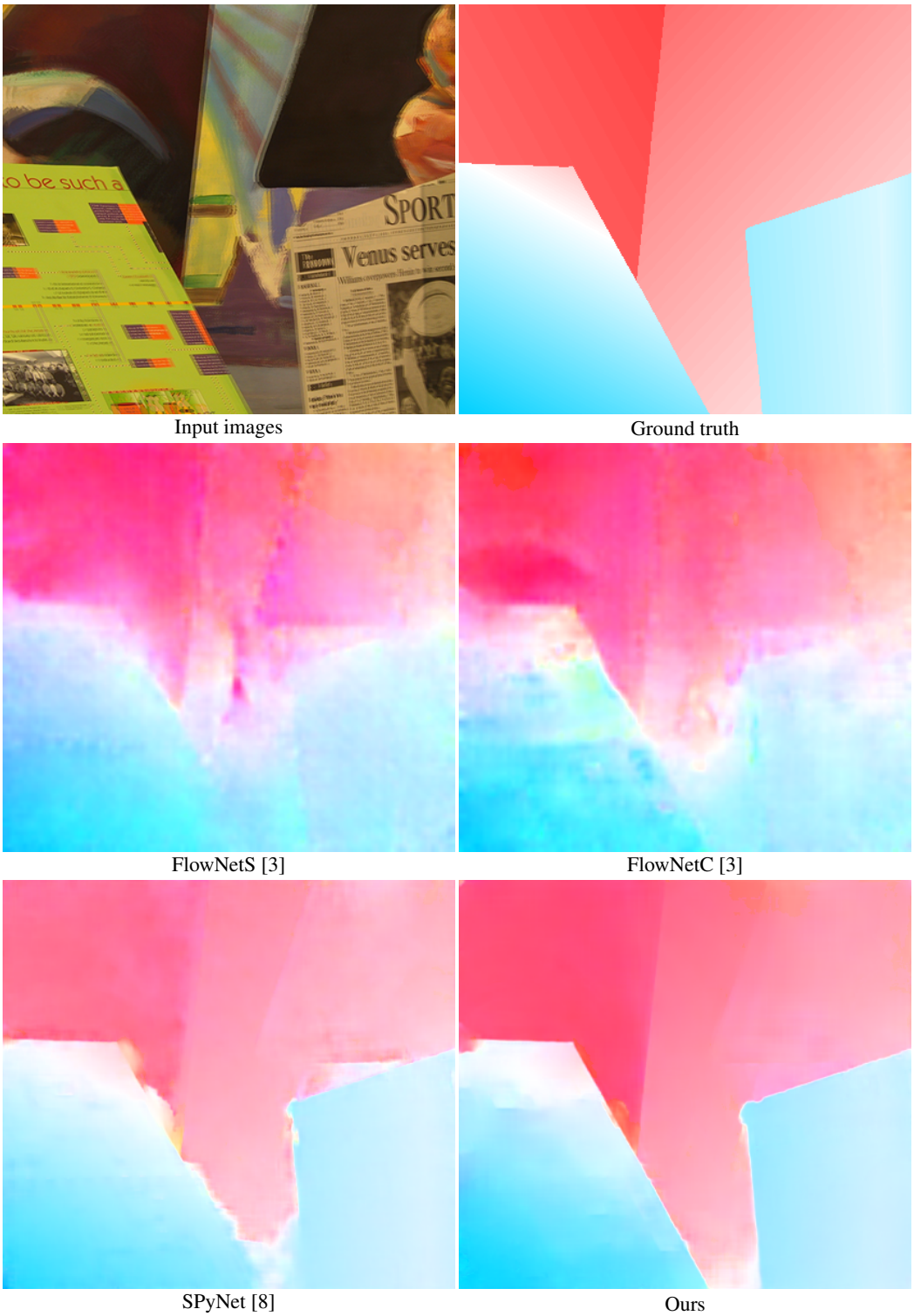


Figure 53: **Visual comparisons on the Middlebury dataset.**

References

[1] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. *IJCV*, 92(1):1–31, 2011.

[2] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, 2012.

[3] P. Fischer, A. Dosovitskiy, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox. FlowNet: Learning optical flow with convolutional networks. In *ICCV*, 2015.

[4] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *CVPR*, 2012.

[5] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017.

[6] M. Jaderberg, K. Simonyan, and A. Zisserman. Spatial transformer networks. In *NIPS*, 2015.

[7] M. Menze and A. Geiger. Object scene flow for autonomous vehicles. In *CVPR*, 2015.

[8] A. Ranjan and M. J. Black. Optical flow estimation using a spatial pyramid network. In *CVPR*, 2017.