SUPPLEMENTARY MATERIAL OF
SENSE: a Shared Encoder Network for Scene-flow Estimation

Huaizu Jiang[1†] Deqing Sun[2*] Varun Jampani[2*]
Zhaoyang Lv[3†] Erik Learned-Miller[1] Jan Kautz[2]
[1]UMass Amherst [2]NVIDIA [3]Georgia Tech

In this supplementary material, we provide more details about network training, rigidity-based warped disparity refinement for scene flow estimation, the proposed network architecture, and additional ablation as well well visual results.

1. Training Details

We perform pre-training on the synthetic SceneFlow dataset and fine-tuning on Sintel and KITTI, respectively.

Synthetic SceneFlow Dataset. We use the subset of FlyingThings3D used in [2], Monkaa, and Driving for pre-training. We remove images whose maximum optical flow magnitude is greater than 500. We end up using 128,753 samples for training.

Supervised training is performed for the pre-training with ground-truth annotations of optical flow, disparity, and their associated occlusions. The loss function is defined as

$$\mathcal{L}_{sp} = (\mathcal{L}_F + \mathcal{L}_{O_F}) + 0.25 \times (\mathcal{L}_D + \mathcal{L}_{O_D})\,. \tag{1}$$

For Monkaa and Driving, since only optical flow and disparity annotations are available, we only set $\mathcal{L}_{O_D}$ and $\mathcal{L}_{O_F}$ to 0 for training data sampled from Monkaa and Driving.

During training, we use color jittering, including randomly changing gamma value, changing brightness, changing contrast, and adding Gaussian noise, for both optical flow and disparity training. Additionally, we use random crops and vertical flips for stereo training images. The crop size is 256×512. For optical flow training images, we perform extensive data augmentations including random crop, translation, rotation, zooming, squeezing, and horizontal and vertical flip, where the crop size is 384×640. The network is trained for 100 epochs with a batch size of 8 using the Adam optimizer [3]. We use synchronized Batch Normalization [5] to ensure there are enough training samples for estimating Batch Normalization layers' statistics when using multiple GPUs. The initial learning rate is 0.001 and decreased by factor of 10 after 70 epochs.

Sintel. We fine-tune the pre-trained model on Sintel. Sintel training data provides optical flow, disparity, and their corresponding occlusion annotations. We therefore use the same loss function as used for the pre-training.

During training, we apply the same color jittering used for pre-training. Similarly we use random crops and vertical flips for stereo training images with crop size of 384×768. For optical flow training images, we perform extensive data augmentations as well including random crop, translation, rotation, zooming, squeezing, and horizontal and vertical flip, where the crop size is 384×768.

Synchronized Batch Normalization is used with batch size of 8. The model is first trained for 500 epochs using the Adam optimizer with an initial learning rate of 0.0005, which is decreased by factor of 2 after every 100 epochs. The weight decay is 0.0004. After 500-epoch training is finished, we keep fine-tuning the model for another 500 epochs using Adam with an initial learning rate of 0.0002, which is decreased by factor of 2 after every 100 epochs. The weight decay remains 0.0004.

KITTI. On KITTI (including KITTI2012 and KITTI2015), we use both supervised loss and semi-supervised loss. The final loss is defined as

$$\mathcal{L} = \underbrace{\mathcal{L}_F + \mathcal{L}_D}_{\text{supervised loss}} + \underbrace{\alpha_O\big(\mathcal{L}_{O_{Fd}} + \mathcal{L}_{O_{Dd}}\big) + \alpha_{S_d}\mathcal{L}_{S_d}}_{\text{distillation loss}} + \underbrace{\alpha_{PC}\mathcal{L}_{PC} + \alpha_{SC}\mathcal{L}_{SC} + \mathcal{L}_{SS} + \mathcal{L}_{REG}}_{\text{self-supervised loss}}, \tag{2}$$

†The work was begun while the author was an intern at NVIDIA.
*Currently affiliated with Google.

where $\mathcal{L}_{O_{Fd}}$ and $\mathcal{L}_{O_{Dd}}$ are distillation loss for optical flow occlusion and disparity occlusion, respectively. They are defined as `smooth_L1` loss between the pseudo ground-truth (*i.e.*, estimations from a model pre-trained on synthetic SceneFlow dataset) and estimations from the model being trained. On the validation set, we empirically found $\alpha_O = 0.05, \alpha_{S_d} = 1, \alpha_{PC} = 0.5, \alpha_{SC} = 0.5$ work well. For the SSIM loss, we use $\gamma_D = 0.005 \times C_H \times C_W$ and $\gamma_F = 0.01 \times C_H \times C_W$[1], where C_H and C_W are crop height and width, respectively. For the regularization term, we empirically set $\beta_F = \beta_D = 0.5$.

During training, we use similar color jittering used in pre-training but with a probability of 0.5. Similarly we use random crops and vertical flips for stereo training images with crop size of 320×768. For optical flow training images, we perform extensive data augmentations as well including random crop, translation, rotation, zooming, squeezing, and horizontal and vertical flip, where the crop size is 320×768.

Synchronized Batch Normalization is used with batch size of 8. The model is fine-tuned for 1,500 epochs using the Adam optimizer with an initial learning rate of 0.001, which is decreased by factor of 2 at epochs of 400, 800, 1,000, 1200, and 1,400. The weight decay is 0.0004. Another round of fine-tuning is followed with an initial learning rate of 0.0002, which is decreased by factor of 2 at epochs of 400, 600, 800, and 900.

2. Rigidity-based Warped Disparity Refinement for Scene Flow Estimation

Determine rigidity area. Given the estimated semantic segmentation labels of the first left frame $\mathbf{S}^{1,l}$, we select pixels as static rigid regions by removing pixels which have a semantic label of vehicle, pedestrian, cyclist, or sky. This step gives a conservative selection of static regions with points not at infinity. The output is a binary mask $\mathbf{B}$ with the label 1 indicating static rigid region. Since the semantic segmentation can be inaccurate at object boundary, we further perform an erosion operation with a size of 10 on the static rigid region mask $\mathbf{B}$.

Estimate rigid flow induced by camera motion. Given the estimated flow $\mathbf{F}^1$ and disparity $\mathbf{D}^1$ of the left frame, we calculate the ego-motion flow induced by the rigid camera motion by minimizing the weighted errors between predicted rigid flow $\mathbf{F}^1_{\mathbf{R}}$ and optical flow $\mathbf{F}^1$ in the background region pixels $\mathbf{x} \in \mathbf{R}^2$:

$$\arg\min_{\xi} \quad \mathbf{r}^T(\xi;\mathbf{x})\mathbf{W}\mathbf{r}(\xi;\mathbf{x}) \tag{3}$$

$$\mathbf{r}(\xi;\mathbf{x}) = \mathbf{F}^1(\mathbf{x}) - \mathbf{F}^1_{\mathbf{R}}(\xi;\mathbf{x}) \tag{4}$$

$$\mathbf{F}^1_{\mathbf{R}}(\xi;\mathbf{x}) = \mathcal{W}(\xi;\mathbf{x},\mathbf{D}^1(\mathbf{x})) - \mathbf{x} \tag{5}$$

where $\mathbf{x} \in \mathbf{R}^2$ denotes the pixels in 2D image space which are within the rigid areas $\mathbf{B}$. $\mathcal{W}(\xi;\mathbf{x},\mathbf{D}^1)$ is the warping function which transforms the pixels $\mathbf{x}$ and its corresponding disparity $\mathbf{D}^1(\mathbf{x})$ with an estimated transform $\xi \in \mathbf{SE}(3)$. $\mathbf{W}$ is a diagonal weight matrix that depends on residuals using Huber weight function.

We solve equation 3 as an iteratively reweighted least-square problem using Gauss-Newton update:

$$\delta\xi = (\mathbf{J}^T\mathbf{W}\mathbf{J})^{-1}\mathbf{J}^T\mathbf{W}\mathbf{r} \tag{6}$$

$$\xi = \xi \circ \delta\xi \tag{7}$$

where $\circ$ indicates the right composition of $\xi \in \mathbf{SE}(3)$. $\mathbf{J}$ is the Jacobian matrix of $\partial\mathbf{F}^1_{\mathbf{R}}(\xi)/\partial\xi$.

Suppose $\mathbf{K}$ is the intrinsic matrix for a pin-hole camera without distortion, which can be parameterized as (f_x, f_y, c_x, c_y) with f_x, f_y as its focal length and c_x, c_y as its offset along the two axes. The baseline of the stereo pair is b. We define the 3D point $\mathbf{p} = (p_x, p_y, p_z)$ as $\mathbf{p} = (f_x b/\mathbf{D}^1(\mathbf{x}))\mathbf{K}^{-1}\mathbf{x}$. Through chain-rule, we can derive the analytical form of the Jacobian matrix $\mathbf{J}$. To simplify the computation, we use the inverse depth parameterization $\mathbf{p} = (p_u/p_d, p_v/p_d, 1/p_d)$ in which $\mathbf{x} = (p_u, p_v) \in \mathbf{R}^2$ is the pixel of coordinate of $\mathbf{x}$ and p_d is the inverse depth as $p_d = \mathbf{D}^1(\mathbf{x})/(f_x b)$. Thus, we obtain the Jacobian matrix at a pixel $\mathbf{x}$ as:

$$\begin{bmatrix} -p_u p_v f_x & (1+p_u^2) f_x & -p_v f_x & p_d f_x & 0 & -p_u p_d f_x \\ -(1+p_v^2) f_y & p_u p_v f_y & p_u f_y & 0 & p_d f_y & -p_u p_d f_y \end{bmatrix} \tag{8}$$

We perform the Gauss-Newton update if the absolute residual error is bigger than 10^{-6} with a maximum of 20 iterations. All operations are implemented in Pytorch and executed in GPU. The running time of the total optimization varies between 0.03s and 0.2s, according to the number of iterations. In average, the optimization step takes 0.1s for KITTI image of resolution 375x1242.

[1] In our definition of SSIM loss, the function $SS(\cdot,\cdot)$ gives a single scalar value.

The final optical flow $\mathbf{F}$ is an element-wise linear composition of $\mathbf{F}^1$ and $\mathbf{F}^1_{\mathbf{R}}$ as:

$$\mathbf{F} = (\mathbf{1} - \mathbf{B}) \otimes \mathbf{F}^1 + \mathbf{B} \otimes \mathbf{F}^1_{\mathbf{R}} \tag{9}$$

where $\otimes$ indicates element-wise multiplications.

Estimate warped second frame rigid disparity. Given the estimated optimal $\xi^\star$, we define the disparity $\mathbf{D}^{1\to2}_{\mathcal{W},\mathbf{R}}$ of the second frame warped from the first frame following the optimal rigid transform $\xi^\star$:

$$\mathbf{D}^{1\to2}_{\mathcal{W},\mathbf{R}} = \mathcal{W}^{1\to2}_{\mathbf{D}}(\xi^\star; \mathbf{D}^1) \tag{10}$$

where $\mathcal{W}^{1\to2}_{\mathbf{D}}(\cdot)$ defines the disparity channel output from the warping function $\mathcal{W}(\cdot)$ through a forward warping. Given the forward optical flow $\mathbf{F}^1$, the warped disparity of the second frame can be computed through an inverse warping $\mathcal{W}^{2\to1}_{\mathbf{D}}$ as:

$$\mathbf{D}^{2\to1}_{\mathcal{W}} = \mathcal{W}^{2\to1}_{\mathbf{D}}(\mathbf{F}^1, \mathbf{D}^2) \tag{11}$$

We find that the disparity through forward warping $\mathbf{D}^{1\to2}_{\mathcal{W},\mathbf{R}}$ gives more accurate disparity in static region and can better handle occlusions. The final warped disparity $\mathbf{D}^2_{\mathcal{W}}$ is a element-wise linear composition of $\mathbf{D}^{2\to1}_{\mathcal{W}}$ and $\mathbf{D}^{1\to2}_{\mathcal{W},\mathbf{R}}$ as:

$$\mathbf{D}^2_{\mathcal{W}} = (\mathbf{1} - \mathbf{B}) \otimes \mathbf{D}^{2\to1}_{\mathcal{W}} + \mathbf{B} \otimes \mathbf{D}^{1\to2}_{\mathcal{W},\mathbf{R}} \tag{12}$$

Note that both warping function cannot deal with out-of-boundary pixels due to two-view occlusion. This can be resolved by the additional refinement network detailed in the following section.

Table 1. Definition of our shared encoder. H and W denote the height and width of the input images. $[\cdot]$ indicates a residual block [1]. We use convolution with stride of 2 to perform downsampling. The first downsampling is performed in the layer conv1_1.

layer name	output size	layer setting
input	$H \times W \times 3$	-
conv1_1 conv1_2 conv1_3	$\frac{H}{2} \times \frac{W}{2} \times 32$	$3 \times 3, 32$ $3 \times 3, 32$ $3 \times 3, 32$
conv2	$\frac{H}{4} \times \frac{W}{4} \times 32$	$\begin{bmatrix} 3 \times 3, 32 \\ 3 \times 3, 32 \end{bmatrix} \times 3$
conv3	$\frac{H}{8} \times \frac{W}{8} \times 64$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 16$
conv4	$\frac{H}{16} \times \frac{W}{16} \times 128$	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 3$
conv5	$\frac{H}{32} \times \frac{W}{32} \times 128$	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 3$

3. Details of Network Architecture

- Table 1 provides the detailed network architecture of our shared encoder, which is a ResNet-like [1] architecture.
- Table 2 provides details of the pyramid pooling module (PPM), which aggregates multi-scale feature maps to enhance disparity estimation and semantic segmentation.
- The Hourglass module used for disparity estimation refinement is illustrated in Table 3. The input is a concatenation of upsampled disparity (by a factor of 2), the feature map of the first image (128-dimensional), and the warped feature map of the second image (128-dimensional). The output is a residual disparity estimation that is added to the twice upsampled disparity.

Table 2. Definition of our PPM head, where branch1, branch2, branch3, and branch4 are all parallel branches on top of the conv5 layer in the encoder.

layer name	output size	layer setting
branch1	$\frac{H}{32} \times \frac{W}{32} \times 128$	1×1 adaptive avg. pool $1 \times 1, 128$ bilinear interpolation
branch2	$\frac{H}{32} \times \frac{W}{32} \times 128$	2×2 adaptive avg. pool $1 \times 1, 128$ bilinear interpolation
branch3	$\frac{H}{32} \times \frac{W}{32} \times 128$	3×3 adaptive avg. pool $1 \times 1, 128$ bilinear interpolation
branch4	$\frac{H}{32} \times \frac{W}{32} \times 128$	6×6 adaptive avg. pool $1 \times 1, 128$ bilinear interpolation
fusion	$\frac{H}{32} \times \frac{W}{32} \times 128$	concat of conv5, branch1 branch2, branch3, and branch4 $3 \times 3, 128$

Table 3. Definition of our disparity Hourglass refinement model.

layer name	output size	layer setting
input	$\frac{H}{2} \times \frac{W}{2} \times 257$	-
conv1	$\frac{H}{4} \times \frac{W}{4} \times 514$	$3 \times 3, 514$
conv2	$\frac{H}{8} \times \frac{W}{8} \times 514$	$3 \times 3, 514$
conv3	$\frac{H}{8} \times \frac{W}{8} \times 514$	$3 \times 3, 514$
conv4	$\frac{H}{4} \times \frac{W}{4} \times 514$	bilinear interpolation $3 \times 3, 514$
conv5	$\frac{H}{2} \times \frac{W}{2} \times 257$	bilinear interpolation $3 \times 3, 257$
output	$\frac{H}{2} \times \frac{W}{2} \times 1$	$3 \times 3, 1$

- The refinement network for warped disparity, which is used for scene flow estimation, can be found in Table 4. The input is a concatenation of $\mathbf{I}^{1,l}$, $\mathbf{O}_F^{1,l}$, $\mathbf{D}^{1,l}$, and $g(\mathbf{D}^{2,l}, \mathbf{F}^{1,l})$, with 6 (=3+1+1+1) channels in total. The network consists of an encoder, a decoder, and skip connections between them. It contains 9.1M parameters in total and takes 0.01s for inference for a KITTI image with resolution of 375×1242. We perform supervision for intermediate layers of the decoder, in a manner similar to optical flow and disparity estimations.

4. More Ablation Studies

New network design. As shown in Table 5, we study the effectiveness of new network designs for disparity estimation. The baseline model has exactly the same architecture as PWC-Net [4] for optical flow estimation, except we construct a 1D cost volume for disparity estimation. It is a compact model with 7.1M parameters. However, most of the parameters concentrate in the decoder due to DenseNet blocks. By removing the last pyramid in both encoder and decoder and adding Batch Normalization layers, we obtain significant improvement in disparity while halving the parameters. By replacing the original encoder consisting of plain CNN layers with deeper residual blocks, we obtain further improvements and yet still have fewer parameters. Adding PPM and hourglass refinement keeps improving the accuracy. Our final model has slightly more parameters than the baseline, but the performance on both synthetic and real-world benchmark datasets increases substantially.

Table 4. Definition of our warped disparity refinement model. output5 is on top of decoder_layer_5. output4, output3, and output2 are computed similarly. output1 is on top of decoder_layer1_2. We compute loss for all five output during training and sum them up as the final loss. During inference, we only compute output1.

layer name	output size	layer setting
input	$H \times W \times 6$	-
encoder_layer1	$H \times W \times 32$	$3 \times 3, 32$
encoder_layer2	$\frac{H}{2} \times \frac{W}{2} \times 32$	2×2 avg. pool, stride of 2 $3 \times 3, 32$
encoder_layer3	$\frac{H}{4} \times \frac{W}{4} \times 32$	2×2 avg. pool, stride of 2 $3 \times 3, 32$
encoder_layer4	$\frac{H}{8} \times \frac{W}{8} \times 32$	2×2 avg. pool, stride of 2 $3 \times 3, 32$
encoder_layer5	$\frac{H}{16} \times \frac{W}{16} \times 32$	2×2 avg. pool, stride of 2 $3 \times 3, 32$
bottleneck	$\frac{H}{32} \times \frac{W}{32} \times 32$	2×2 avg. pool, stride of 2 $3 \times 3, 32$
decoder_layer5	$\frac{H}{16} \times \frac{W}{16} \times 512$	$3 \times 3, 512$ $2\times$ bilinear interpolation
decoder_layer4	$\frac{H}{8} \times \frac{W}{8} \times 256$	concat. with encoder_layer_5 $3 \times 3, 256$ $2\times$ bilinear interpolation
decoder_layer3	$\frac{H}{4} \times \frac{W}{4} \times 128$	concat. with encoder_layer_4 $3 \times 3, 128$ $2\times$ bilinear interpolation
decoder_layer2	$\frac{H}{2} \times \frac{W}{2} \times 64$	concat. with encoder_layer_3 $3 \times 3, 64$ $2\times$ bilinear interpolation
decoder_layer1_1	$H \times W \times 32$	concat. with encoder_layer_2 $3 \times 3, 32$ $2\times$ bilinear interpolation
decoder_layer1_2	$H \times W \times 32$	concat. with encoder_layer_1 $3 \times 3, 32$
output5	$\frac{H}{16} \times \frac{W}{16} \times 1$	$3 \times 3, 1$
output4	$\frac{H}{8} \times \frac{W}{8} \times 1$	$3 \times 3, 1$
output3	$\frac{H}{4} \times \frac{W}{4} \times 1$	$3 \times 3, 1$
output2	$\frac{H}{2} \times \frac{W}{2} \times 1$	$3 \times 3, 1$
output1	$H \times W \times 1$	$3 \times 3, 1$

Table 5. Ablation study of design choices for disparity.

5 layers+BN	ResNet encoder	PPM	hourglass	#params	FlyThings3D (EPE)	KITTI2015 (EPE)
				7.1M	2.10	1.01
✓				3.6M	1.61	0.85
✓	✓			6.9M	1.40	0.77
✓	✓	✓		7.7M	1.32	0.77
✓	✓	✓	✓	8.3M	1.15	0.71

Shared encoder. We report optical flow and disparity errors on Sintel, KITTI 2012, and KITTI 2015 using both separate and shared encoders in Table 6, where a model trained on synthetic SceneFlow dataset is used. As we can see, a shared encoder

Table 6. Effectiveness of the shared encoder for optical flow and disparity estimations.

		Sintel (EPE)		KITTI 2012		KITTI 2015	
		clean	final	EPE	D1/F1-occ	EPE	D1/F1-occ
optical flow	separate encoder	1.97	3.34	2.63	11.72%	6.37	21.15%
	shared encoder	1.91	3.78	2.55	12.56%	6.23	23.29%
disparity	separate encoder	1.56	2.99	1.09	6.17%	1.26	6.62%
	shared encoder	1.70	3.20	1.04	5.42%	1.22	6.38%

leads to better EPE metrics on both KITTI 2012 and KITTI 2015.

5. Visual Results of Optical Flow and Disparity Estimations

We provide more visual results of optical flow and disparity estimations of the test set in Fig. 1 and Fig. 2 for KITTI 2012 and in Fig. 3 and Fig. 4 for KITTI 2015.

We can clearly see our full model (supervised loss plus semi-supervised loss) produces visually better results on both KITTI 2012 and KITTI 2015.

Acknowledgement

Huaizu Jiang and Erik Learned-Miller acknowledge support from AFRL and DARPA (#FA8750- 18-2-0126) and the MassTech Collaborative grant for funding the UMass GPU cluster. The U.S. Gov. is authorized to reproduce and distribute reprints for Gov. purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the AFRL and DARPA or the U.S. Gov.

References

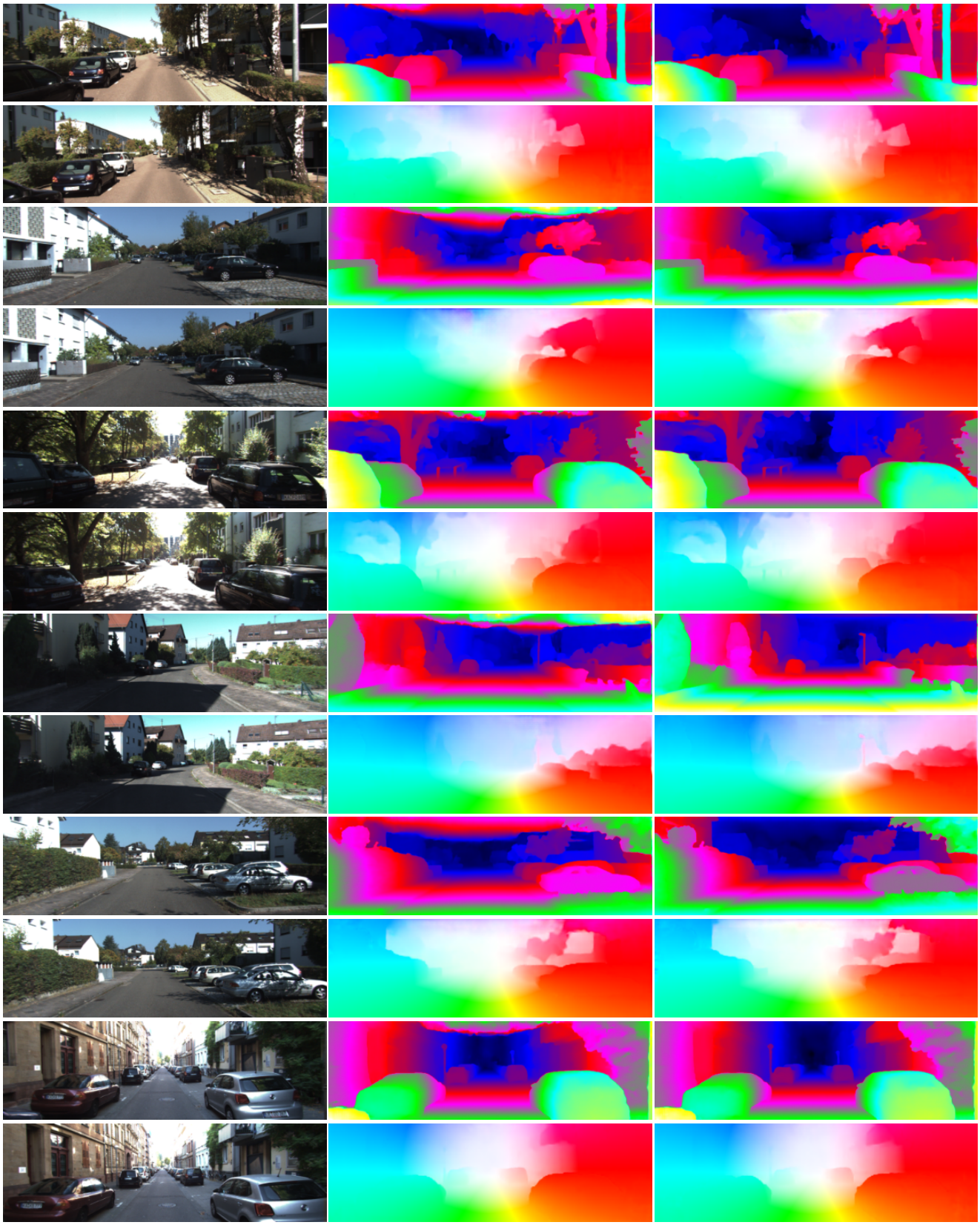
[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 3

[2] Eddy Ilg, Tonmoy Saikia, Margret Keuper, and Thomas Brox. Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation. In *ECCV*, 2018. 1

[3] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 1

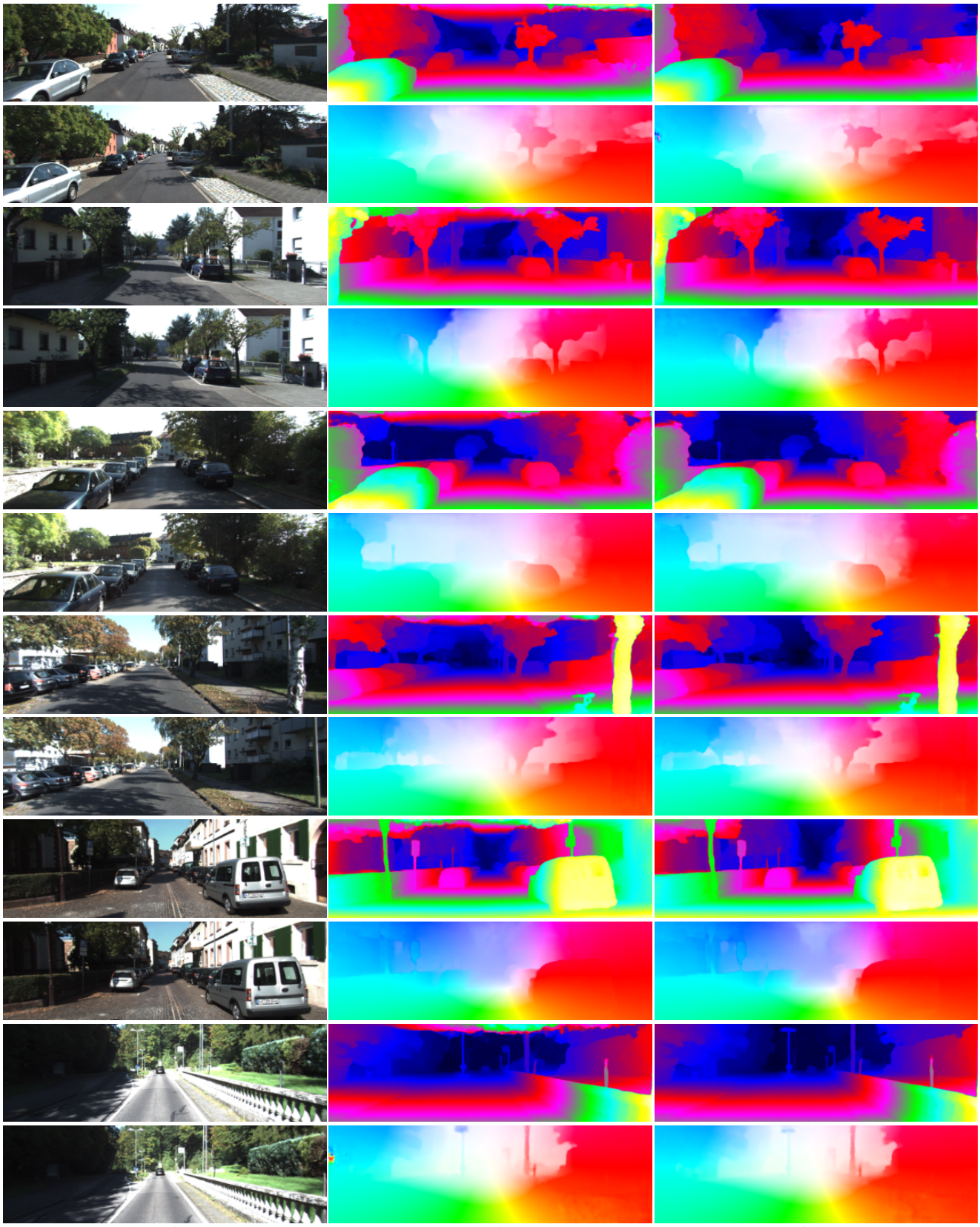
[4] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *CVPR*, June 2018. 4

[5] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *ECCV*, 2018. 1



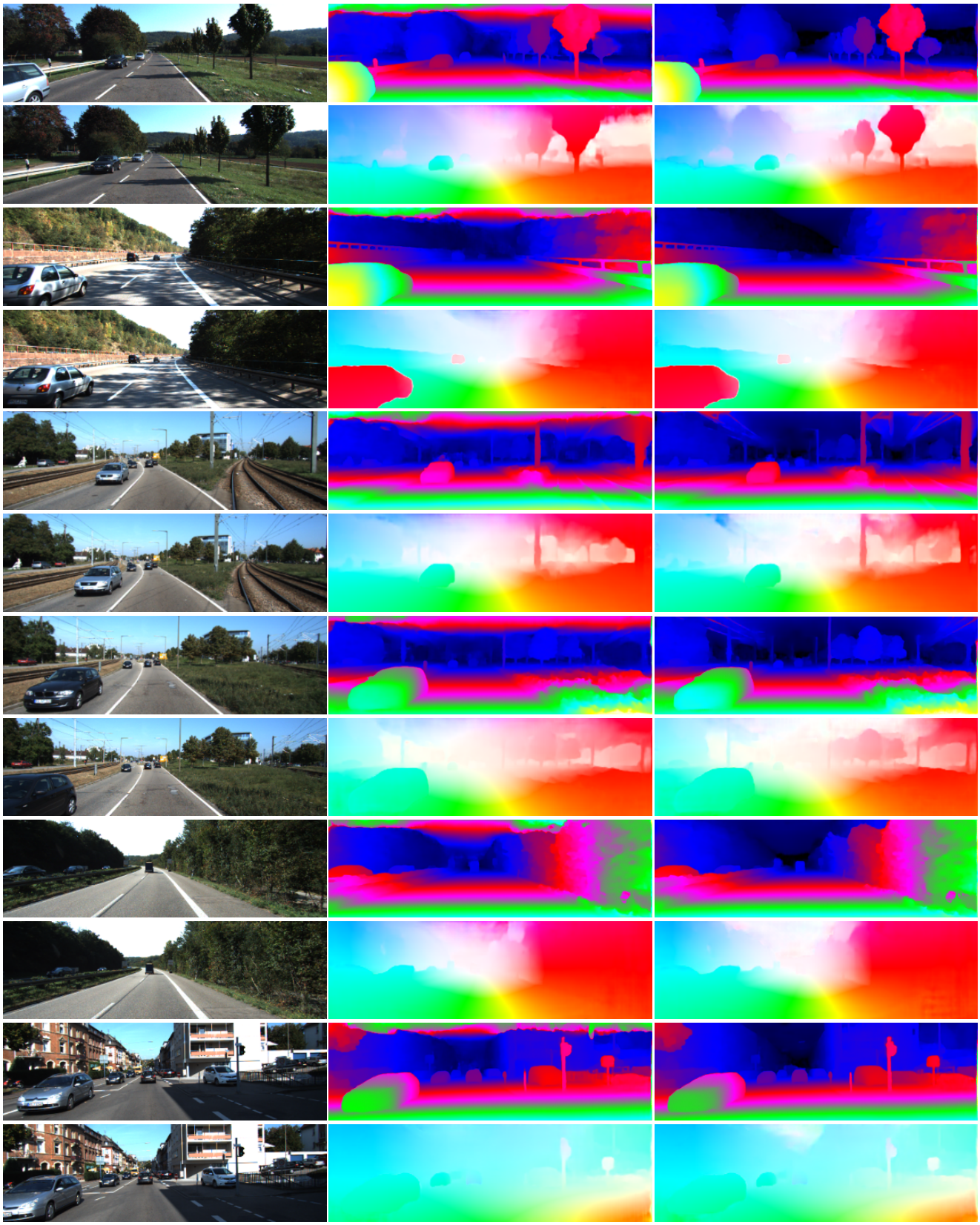
(a) input images (b) supervised loss (c) full loss

Figure 1. Visual results on the test set of KITTI 2012. We show two consecutive video frames in the first column. In the second and third columns, we show disparity in every first row and optical flow in the other one. Best viewed in color.



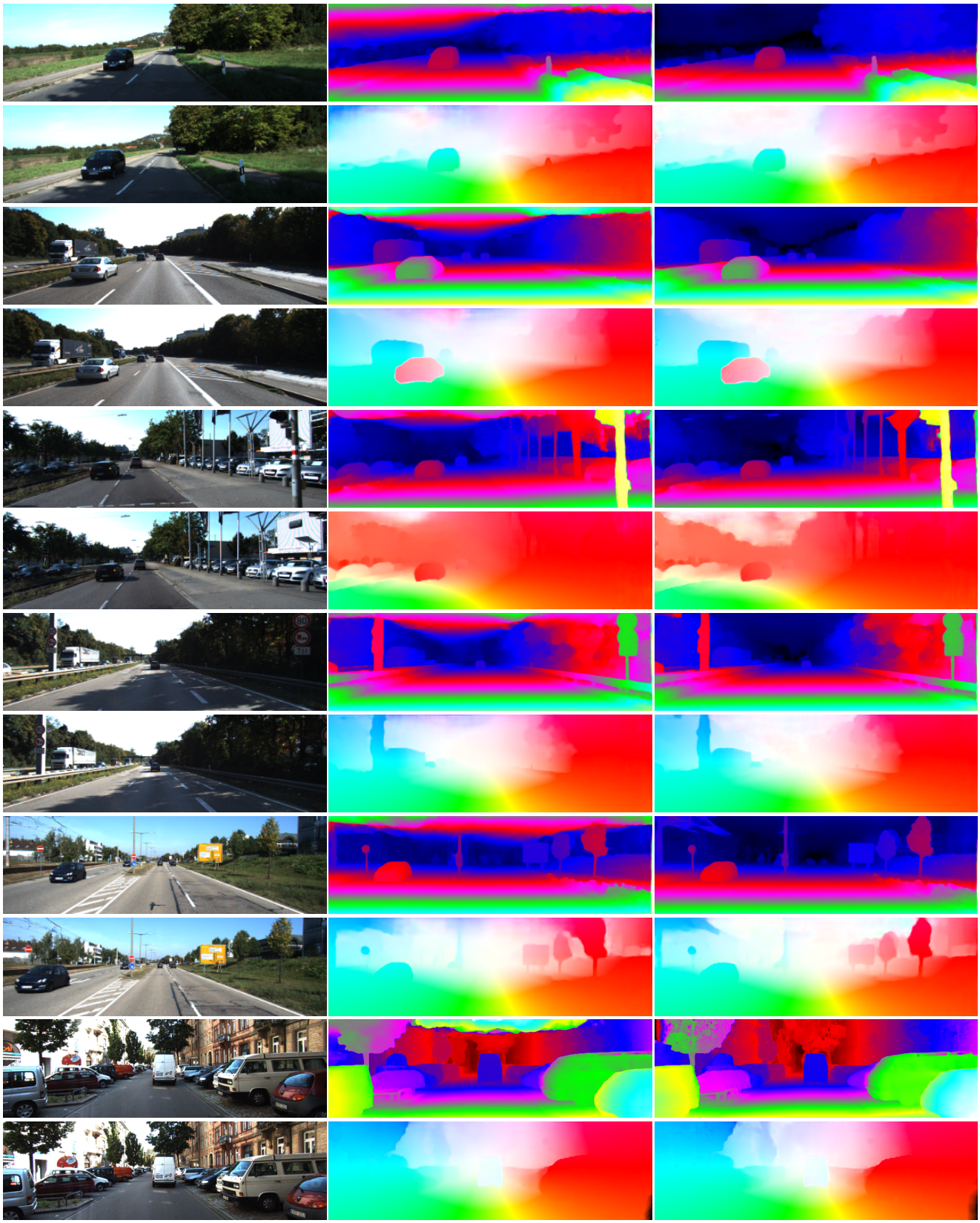
(a) input images (b) supervised loss (c) full loss

Figure 2. Visual results on the test set of KITTI 2012. We show two consecutive video frames in the first column. In the second and third columns, we show disparity in every first row and optical flow in the other one. Best viewed in color.



(a) input images (b) supervised loss (c) full loss

Figure 3. Visual results on the test set of KITTI 2015. We show two consecutive video frames in the first column. In the second and third columns, we show disparity in every first row and optical flow in the other one. Best viewed in color.



(a) input images (b) supervised loss (c) full loss

Figure 4. Visual results on the test set of KITTI 2015. We show two consecutive video frames in the first column. In the second and third columns, we show disparity in every first row and optical flow in the other one. Best viewed in color.