Supplementary Material

SDC – Stacked Dilated Convolution: A Unified Descriptor Network for Dense Matching Tasks

René Schuster[1] Oliver Wasenmüller[1] Christian Unger[2] Didier Stricker[1]
[1]DFKI - German Research Center for Artificial Intelligence [2]BMW Group
firstname.lastname@{bmw,dfki}.de

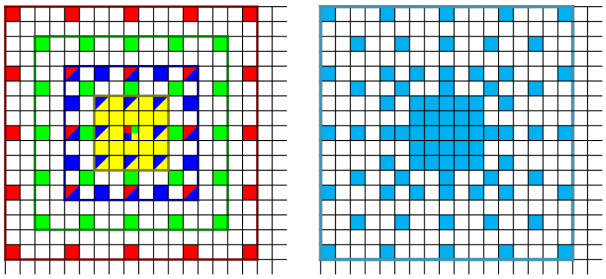
Introduction

This is the supplementary material to our paper *SDC – Stacked Dilated Convolution: A Unified Descriptor Network for Dense Matching Tasks*. It covers the following contents that are mentioned in the main paper:

- Comparison between multiple, parallel, dilated convolutions and a single convolution with a larger sparse kernel.
- Comparison of different hyper-parameters for our novel SDC network architecture.
- Detailed description for our sampling of training data with visual examples for sampled patch triplets.
- More visual results of SDC feature matching with different algorithms on different data sets.

1. Large Sparse Convolution

Multiple dilated convolutions in parallel are similar to a single larger convolution with a sparse kernel (see Figure 1). The difference is that pixels where the parallel kernels overlap are only considered once in a single kernel (for 3×3 kernels this will be the center pixel only) and that a single kernel will merge all information into a single output. With our parallel convolutions we have the choice to add them, stack them, or combine them as we want.



SDC Design Single Sparse Kernel

Figure 1: Two variants of sparse 17×17 kernels. 4 parallel dilated 5×5 kernels (left), and a single kernel (right).

For comparison of both approaches, we design two variants which use single, larger convolutions with the exact same receptive field as our SDC network. The first one produces the same output dimensions at each layer, *i.e.* $64, 64, 128, 256,$ and 128 feature channels. This results in a 4 times larger network compared to our SDC network approximately (disregarding overlapping pixel positions). The second variant is designed to achieve the same network size as our SDC network which results in 4 times less output channels per layer, *i.e.* $16, 16, 32, 64,$ and 32. We call these networks *Fake-big* and *Fake-small* respectively, because these networks try to imitate our original SDC design. The complete architectures look like this:

- *Fake-big*: SparseConv(17,64,1,1)–SparseConv(17,64,1,1)–SparseConv(17,128,1,1)–SparseConv(17,256,1,1)–SparseConv(17,128,1,1)
- *Fake-small*: SparseConv(17,16,1,1)–SparseConv(17,16,1,1)–SparseConv(17,32,1,1)–SparseConv(17,64,1,1)–SparseConv(17,32,1,1)

Sparsity is enforced according to the pattern shown in Figure 1. The numbers parameterizing the convolutions are explained in Section 4.1 of the main paper. Accuracy with network characteristics, ROC curves, and Robustness are evaluated in Table 1 and Figures 2 and 3. These three metrics are explained in Section 4.1 of the main paper.

Table 1: Comparison of the accuracy for our SDC networks and the *Fake* variants that use single, sparse convolutions along with receptive field size (RF), number of parameters (Size) and accumulated sub-sampling factor due to striding.

Network	Accuracy	RF	Size	Factor
SDC (**Ours**)	**97.2 %**	81	1.95 M	1
Fake-big	97.0 %	81	6.3 M	1
Tiny (**Ours**)	96.0 %	25	**0.12 M**	1
Fake-small	96.0 %	81	0.4 M	1

In all evaluations, our SDC network outperforms the *Fake* networks. Considering that *Fake-big* is a much bigger network, this is even more evidence that our design is very powerful. The concatenation of multi-scale features

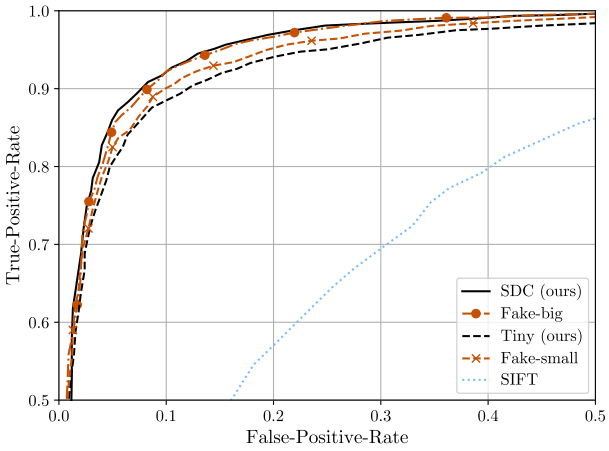


Figure 2: ROC curves for our SDC networks and the two *Fake* variants.

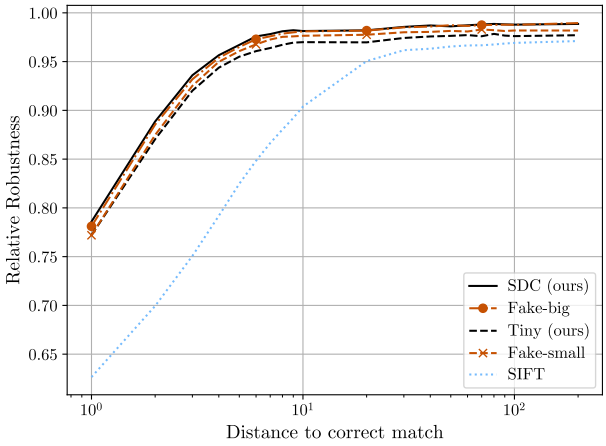


Figure 3: Robustness for our SDC networks and the two *Fake* variants.

and the mixture of multi-scale information at every level is beneficial for image description.

2. Design Decisions

Variation of depth and width of our SDC network structure is covered by our *Tiny* version. In the following, we will vary batch size, activation functions, and loss function and compare them. We will alter one parameter at a time and compare the robustness to our original design. For better comparison, we introduce relative robustness which is the robustness ratio of a model and a reference model. A relative robustness greater than 1 means that the model is better than the reference model. We will use our original SDC network as reference which will result in a baseline of 100 % of relative robustness for this model.

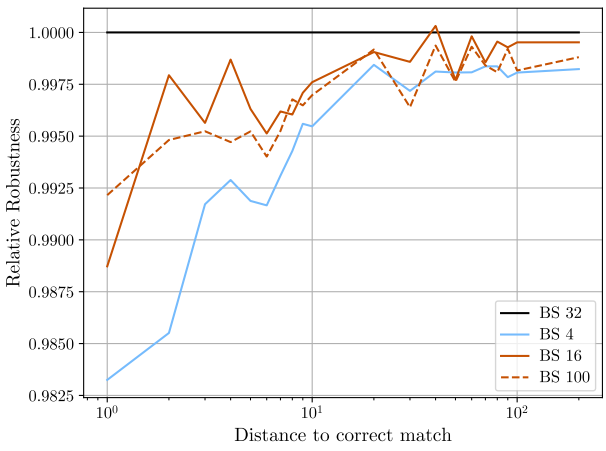


Figure 4: Relative robustness for our SDC network trained with different batch sizes (BS).

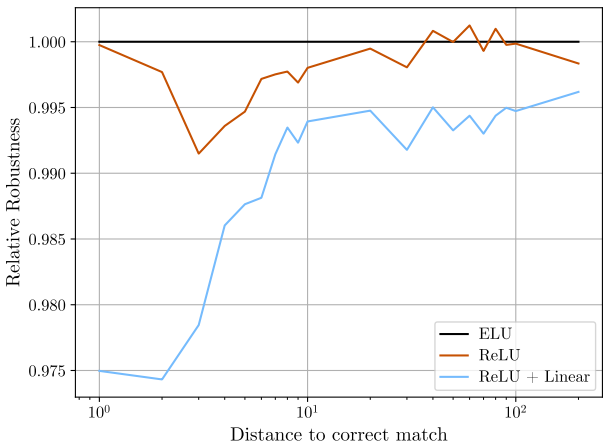


Figure 5: Relative robustness for our SDC network with different activation functions.

Results for different batch sizes are given in Figure 4. The impact of the batch size is minor. Even for very small mini batches, the relative robustness drops by less than 2 percent points. Increasing the batch size is not improving the performance either.

For alternative activation functions, we consider ReLU [12] instead of ELU [5]. Since our final feature vectors are normalized to unit range, the rectification of ReLU restricts the feature space to the non-negative orthant of the 128-dimensional hypercube which is only 2^{-128} of the full volume. Therefore, we also train and compare a network with ReLU and linear activation in the last layer. The comparison is shown in Figure 5.

As mentioned in the main paper where we use triplet training with a thresholded hinge embedding loss [1], we

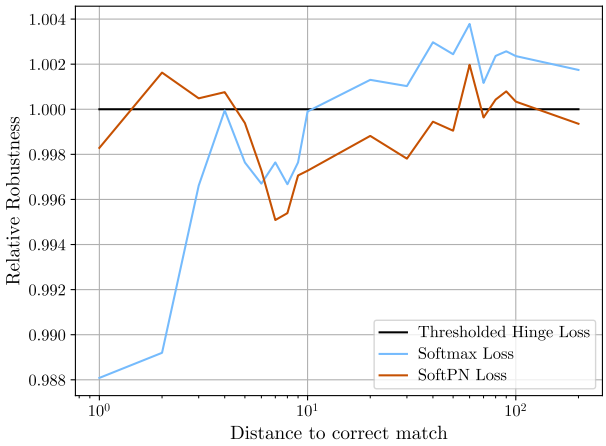


Figure 6: Relative robustness for our SDC network trained with different triplet loss functions.

have also experimented with the softmax loss of [8] and the PN loss of [3] (see Figure 6). Both alternative losses perform better for some displacements, and worse for others. However, the difference in robustness is not significant and training with these real triplet losses is much less stable forcing us to use a lower learning rate and thus increasing overall training time.

3. Training Data

To train a single unified descriptor network that can be used for any algorithm on any domain, we use multiple data sets with very diverse characteristics. They differ in number of sequences and sequence lengths, image resolution, available ground truth, ground truth masks, and color channels. We have tried to consider all these factors to avoid imbalance and over-representations in our training data. In the end, we sample reference images with the following probabilities for each data set:

- KITTI [11]: 0.5
- Sintel [4]: 0.175
- Middlebury Optical Flow[2]: 0.025
- Middlebury Stereo [13]: 0.05
- HD1K [10]: 0.175
- ETH3D [14]: 0,075

We do not sample the same image twice until all images of the respective data set were selected, *i.e*. within each data set, we use all images equally often. For each image in each epoch, 100 randomly sampled reference patches are selected. If a data set provides ground truth for multiple tasks (*e.g*. KITTI [11]), we randomly select one of the tasks from stereo, optical flow, or scene flow. The reference patches are sampled from pixel positions where ground truth exists and where the ground truth displacement points to a visible position in the corresponding view, *i.e*. occlusions and out-of-bound displacements are excluded wherever possible. For each of the 100 reference patches, we extract the corresponding matching patch according to the ground truth. We pad the image with reflection at image boundaries and use bilinear interpolation at sub-pixel positions. The negative patch is extracted by altering the ground truth displacement with a random offset. This random offset is at least 2 pixels and at most 100 pixels large in magnitude. For stereo correspondences, the displacement and the random offset are 1-dimensional along the horizontal direction according to the epipolar constraint. Other correspondences have 2-dimensional displacement, *i.e*. circular around the ground truth correspondence. Since close-by correspondences are harder to distinguish, we sample the random offset non-uniformly. In detail, we split the random offset into two ranges, a close one ($[2, 10]$ pixels) and a distant range ($]10, 100]$ pixels). The close range is selected 3 times more often than the far range and within each range, we sample uniformly. This leads to the overall probability distribution for our random negative displacements shown in Figure 7.

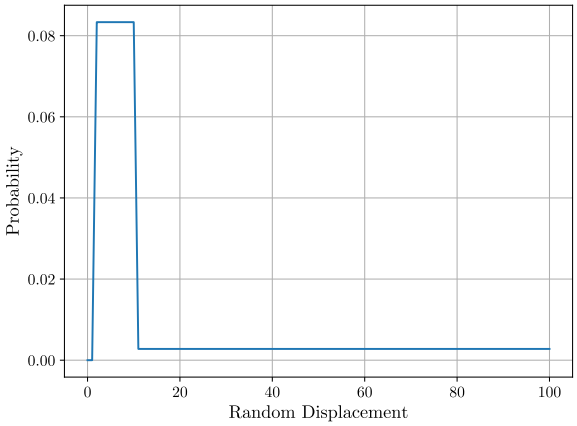


Figure 7: Probability distribution of the random offset used to generate the negative patch correspondence.

Each image is processed completely (all 100 patch triplets) before selecting the next image pair from one of the data sets to reduce IO operations in our training data pipeline. We then shuffle chunks of 3200 triplets to reintroduce randomness across data sets and images. As mentioned in the main paper, the input patches are normalized according to the mean pixel and variance of the complete training data. The mean pixel is [0.3534, 0.3448, 0.3295] and the mean standard deviation is [0.2492, 0.2465, 0.2446] for the red, green, and blue color channels respectively.

Smaller displacements (less than 2 pixels) are not considered for several reasons. Minimal changes in appearance might confuse the network, rounding and interpolation in-

troduce small inaccuracies, and most applications tolerate a matching accuracy less than 2 pixels endpoint error. Visual examples of our training triplets are given in Figure 9.

The 200 training images from KITTI [11], are randomly split into a subset for actual training (70 %), one for validation (20 %), and one for testing in our experiments section (10 %). This is the exact list of sequences for each subset:

- Training: 0, 1, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 18, 19, 20, 21, 22, 24, 25, 27, 28, 29, 30, 31, 33, 34, 35, 37, 38, 39, 40, 41, 43, 44, 45, 47, 49, 50, 51, 54, 55, 56, 58, 59, 62, 63, 66, 67, 68, 70, 71, 74, 75, 76, 78, 79, 82, 83, 85, 86, 87, 88, 89, 90, 93, 95, 96, 97, 99, 101, 102, 103, 104, 105, 107, 109, 111, 113, 114, 116, 117, 118, 120, 123, 125, 128, 129, 130, 132, 134, 135, 137, 138, 139, 140, 141, 143, 144, 145, 147, 148, 149, 150, 151, 152, 154, 155, 156, 157, 160, 161, 162, 163, 164, 165, 167, 168, 169, 170, 171, 172, 175, 177, 178, 179, 180, 182, 183, 185, 187, 188, 189, 191, 194, 195, 197, 198, 199
- Validation: 2, 16, 17, 23, 26, 32, 36, 48, 52, 53, 57, 60, 61, 64, 69, 72, 73, 77, 80, 81, 84, 91, 100, 108, 110, 112, 122, 126, 127, 131, 133, 136, 142, 153, 158, 159, 166, 176, 192, 196
- Testing: 4, 42, 46, 65, 92, 94, 98, 106, 115, 119, 121, 124, 146, 173, 174, 181, 184, 186, 190, 193

4. Visual Results

In addition to the quantitative results for ELAS [6], SGM [7], CPM [9], FlowFields++ [15], and SceneFlowFields [16] presented in the main paper in Tables 2 to 5, we provide visual examples on the different data sets in Figure 10. For each combination of matching algorithm and data set, we show the corresponding images, the estimated matches using the original feature descriptor, and matches computed with our SDC features. Please note that we do not tune any algorithm for our new descriptor. We replace the feature descriptor and change nothing else. For each matching result, we also give an error map indicating where the estimate exceeds an endpoint error of 3 pixels (>3px). The color encoding for these error maps is shown in Figure 8.

Figure 8: Color encoding of the endpoint error in the error maps in Figures 10a to 10o.

In case of scene flow (Figure 10o), we split the visualization of the estimated result into optical flow and two disparity maps. Since CPM [9] computes sparse matches in 3×3 blocks only, we dilate the visualization for matches and error maps. More results can be found in our supplementary video https://youtu.be/RoxfVdfqWpY.

References

[1] Christian Bailer, Kiran Varanasi, and Didier Stricker. CNN-based patch matching for optical flow with thresholded hinge embedding loss. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[2] Simon Baker, Daniel Scharstein, JP Lewis, Stefan Roth, Michael J Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision (IJCV)*, 2011. 3, 15, 16

[3] Vassileios Balntas, Edward Johns, Lilian Tang, and Krystian Mikolajczyk. PN-Net: Conjoined triple deep network for learning local image descriptors. *arXiv preprint arXiv:1601.05030*, 2016. 3

[4] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *European Conference on Computer Vision (ECCV)*, 2012. 3, 13, 14

[5] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (ELUs). *arXiv preprint arXiv:1511.07289*, 2015. 2

[6] Andreas Geiger, Martin Roser, and Raquel Urtasun. Efficient large-scale stereo matching. In *Asian Conference on Computer Vision (ACCV)*. 2010. 4, 7, 8, 10

[7] Heiko Hirschmüller. Stereo processing by semiglobal matching and mutual information. *Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2008. 4, 7, 9, 11

[8] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition (SIMBAD)*, 2015. 3

[9] Yinlin Hu, Rui Song, and Yunsong Li. Efficient coarse-to-fine patchmatch for large displacement optical flow. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 4, 12, 13, 15, 17

[10] Daniel Kondermann, Rahul Nair, Katrin Honauer, Karsten Krispin, Jonas Andrulis, Alexander Brock, Burkhard Gussefeld, Mohsen Rahimimoghaddam, Sabine Hofmann, Claus Brenner, et al. The HCI benchmark suite: Stereo and flow ground truth with uncertainties for urban autonomous driving. In *Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2016. 3, 17, 18

[11] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 3, 4, 7, 12, 19

[12] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *International Conference on Machine Learning (ICML)*, 2010. 2

[13] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision (IJCV)*, 2002. 3, 8, 9

[14] Thomas Schöps, Johannes L Schönberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3, 10, 11

[15] René Schuster, Christian Bailer, Oliver Wasenmüller, and Didier Stricker. FlowFields++: Accurate optical flow correspondences meet robust interpolation. In *International Conference on Image Processing (ICIP)*, 2018. 4, 12, 14, 16, 18

[16] René Schuster, Oliver Wasenmüller, Georg Kuschk, Christian Bailer, and Didier Stricker. SceneFlowFields: Dense interpolation of sparse scene flow correspondences. In *Winter Conference on Applications of Computer Vision (WACV)*, 2018. 4, 19

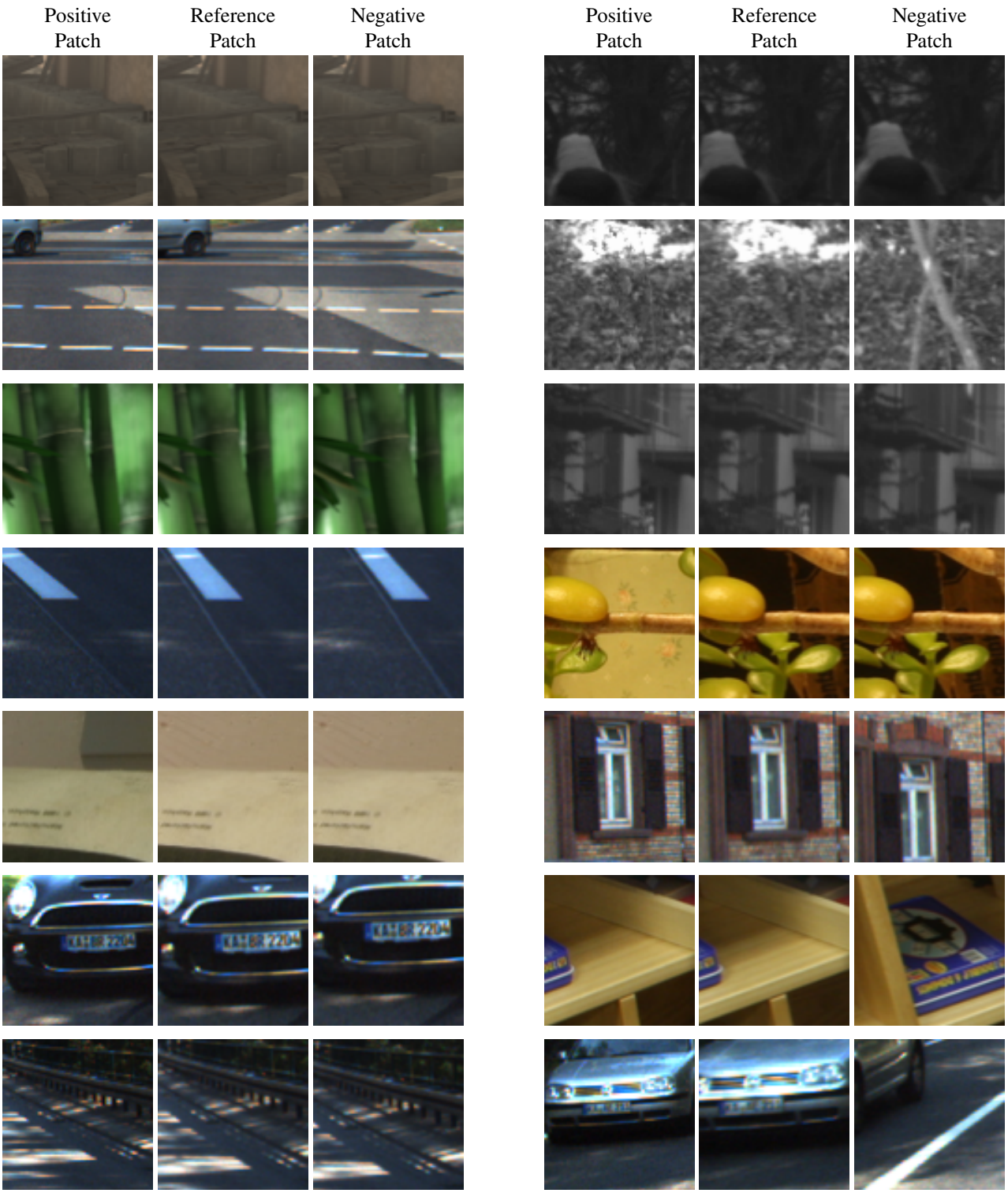
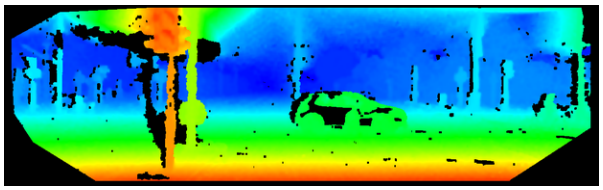


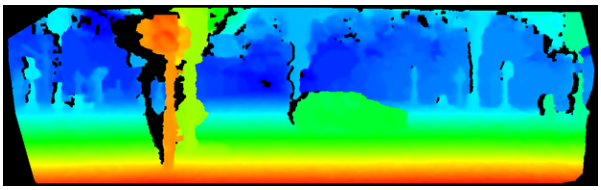
Figure 9: Randomly sampled training triplets. For each reference patch, we select a corresponding patch according to the ground truth displacement and a negative correspondence by adding a random offset to the ground truth.

Left Image

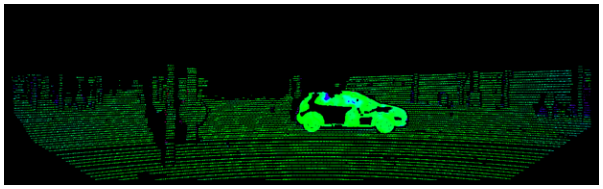
Right Image



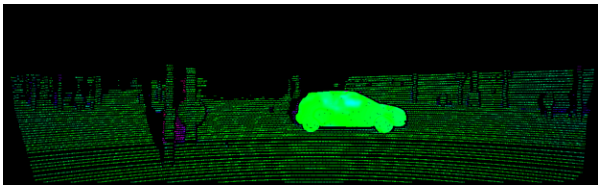
Original Result



Result with SDC



Error Map for Original Result

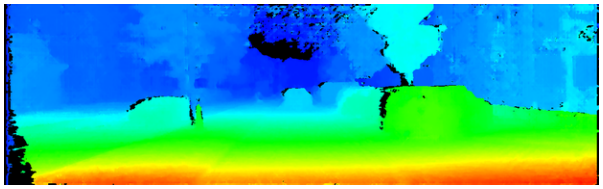


Error Map for Result with SDC

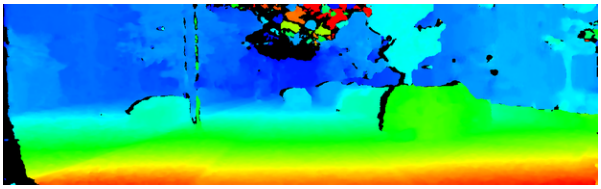
(a) Stereo matching result for ELAS [6] on KITTI [11]. Quantitative results are given in Table 2 of the main paper.

Left Image

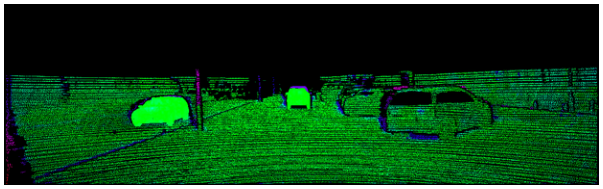
Right Image



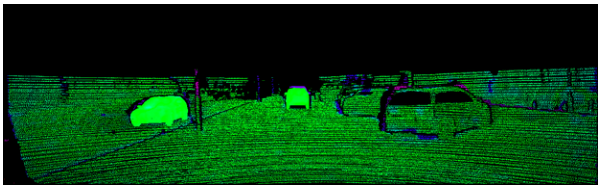
Original Result



Result with SDC



Error Map for Original Result

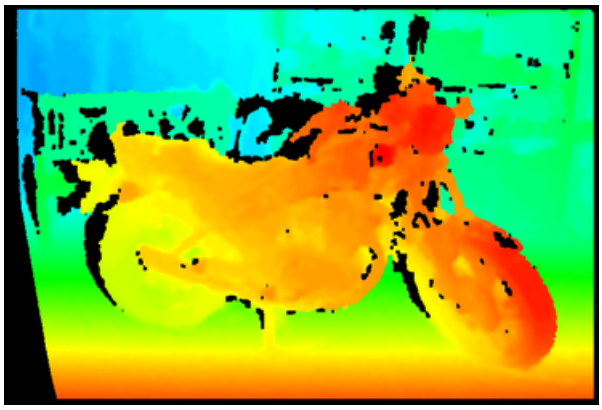


Error Map for Result with SDC

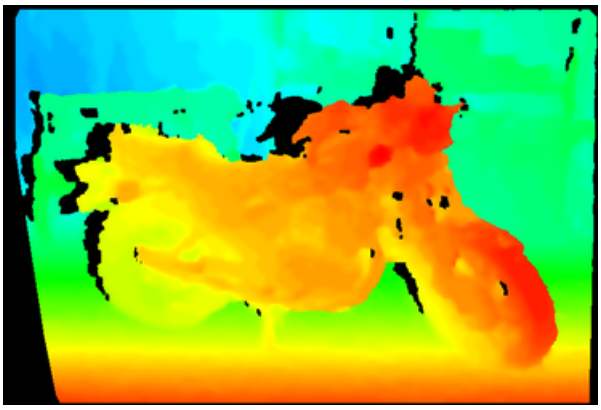
(b) Stereo matching result for SGM [7] on KITTI [11]. Quantitative results are given in Table 2 of the main paper.

Left Image

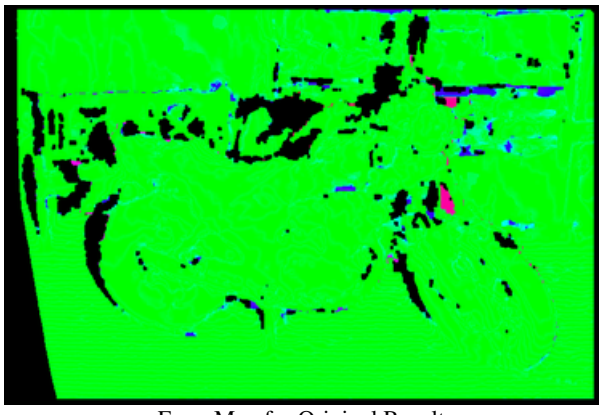
Right Image



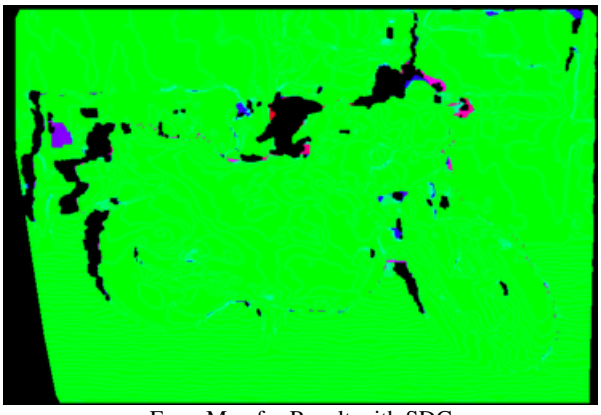
Original Result



Result with SDC



Error Map for Original Result

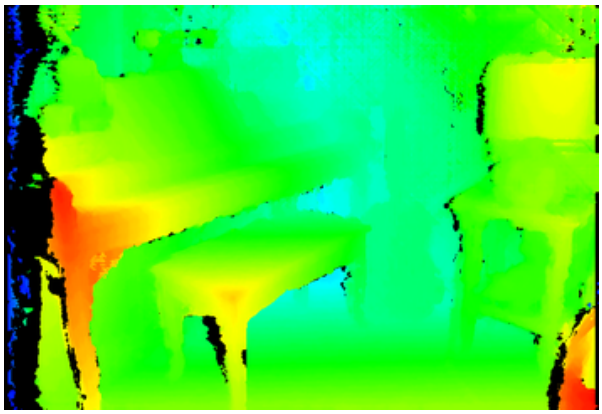


Error Map for Result with SDC

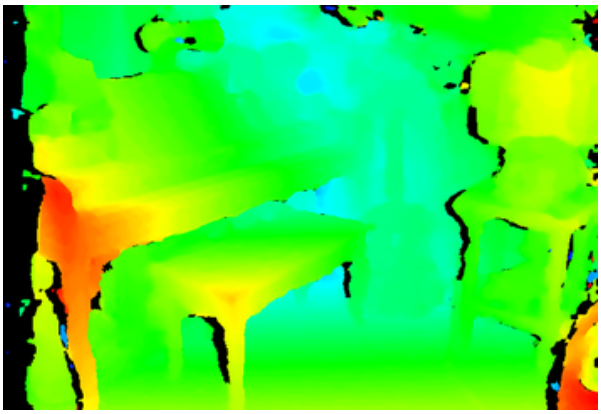
(c) Stereo matching result for ELAS [6] on Middlebury [13]. Quantitative results are given in Table 2 of the main paper.

Left Image

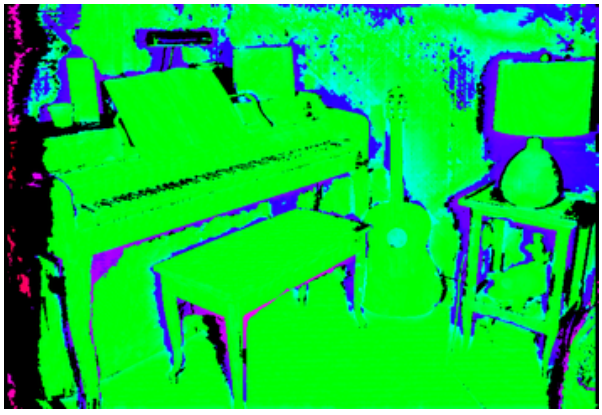
Right Image



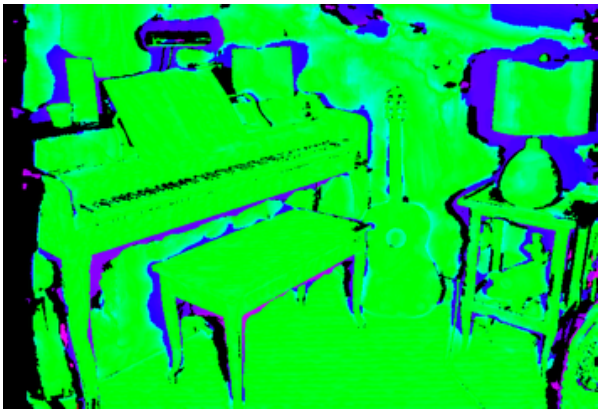
Original Result



Result with SDC



Error Map for Original Result

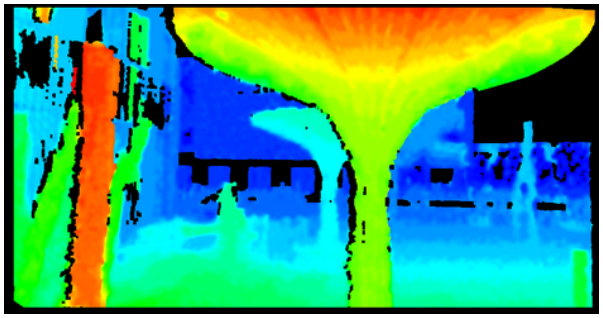


Error Map for Result with SDC

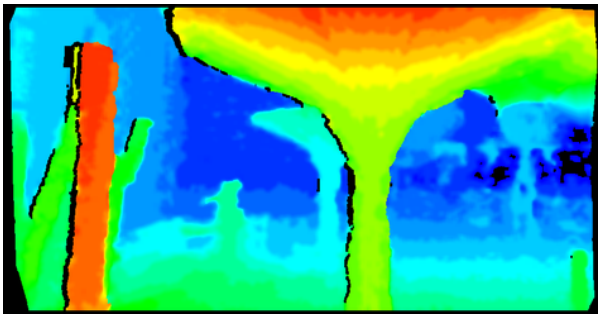
(d) Stereo matching result for SGM [7] on Middlebury [13]. Quantitative results are given in Table 2 of the main paper.

Left Image

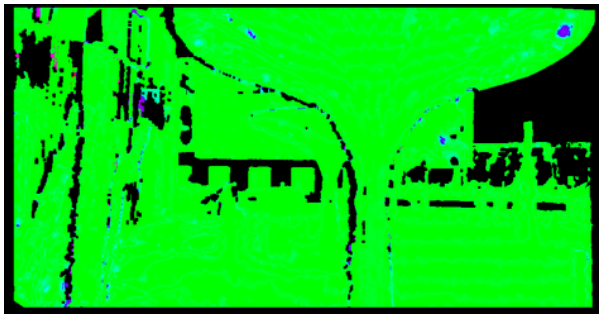
Right Image



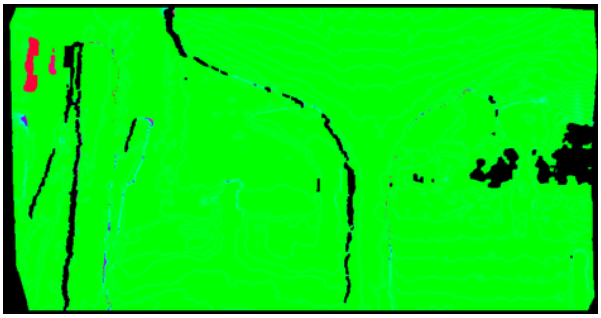
Original Result



Result with SDC



Error Map for Original Result

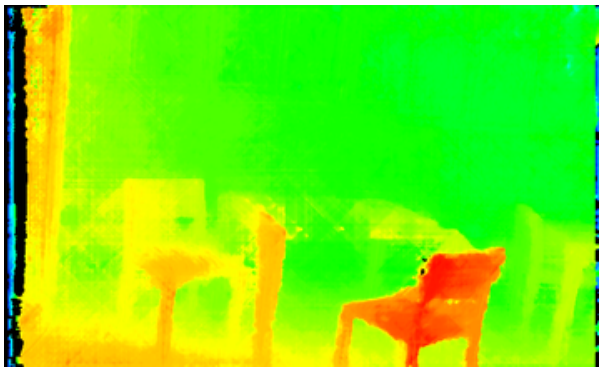


Error Map for Result with SDC

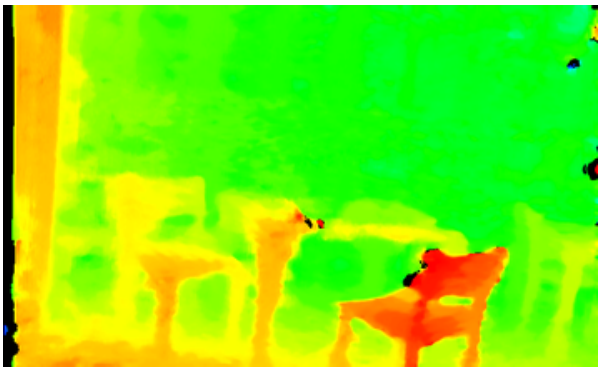
(e) Stereo matching result for ELAS [6] on ETH3D [14]. Quantitative results are given in Table 2 of the main paper.

Left Image

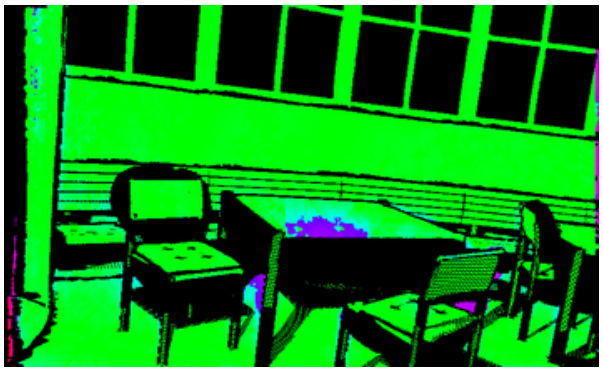
Right Image



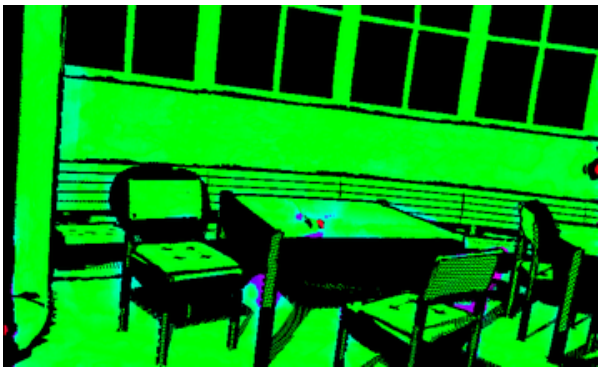
Original Result



Result with SDC



Error Map for Original Result

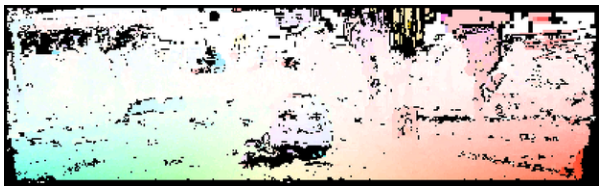


Error Map for Result with SDC

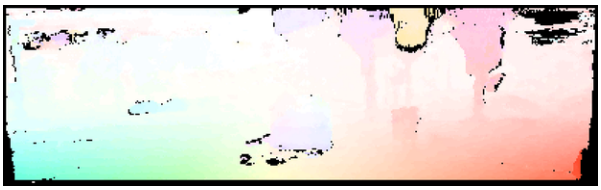
(f) Stereo matching result for SGM [7] on ETH3D [14]. Quantitative results are given in Table 2 of the main paper.

First Image

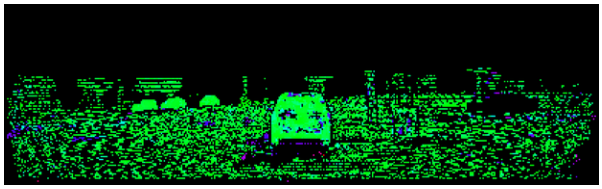
Second Image



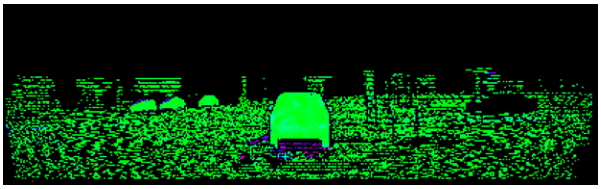
Original Result



Result with SDC



Error Map for Original Result



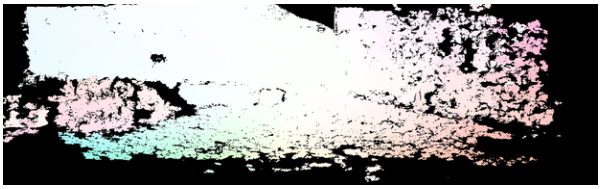
Error Map for Result with SDC

(g) Optical flow result for CPM [9] on KITTI [11]. Quantitative results are given in Table 4 of the main paper.

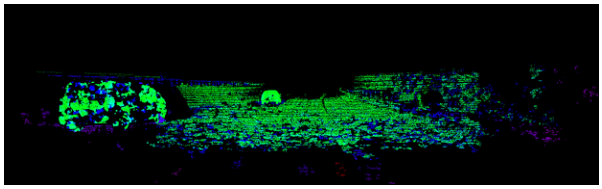
First Image

Second Image

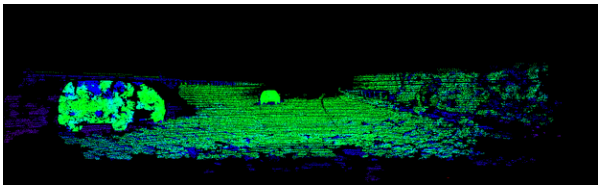
Original Result



Result with SDC

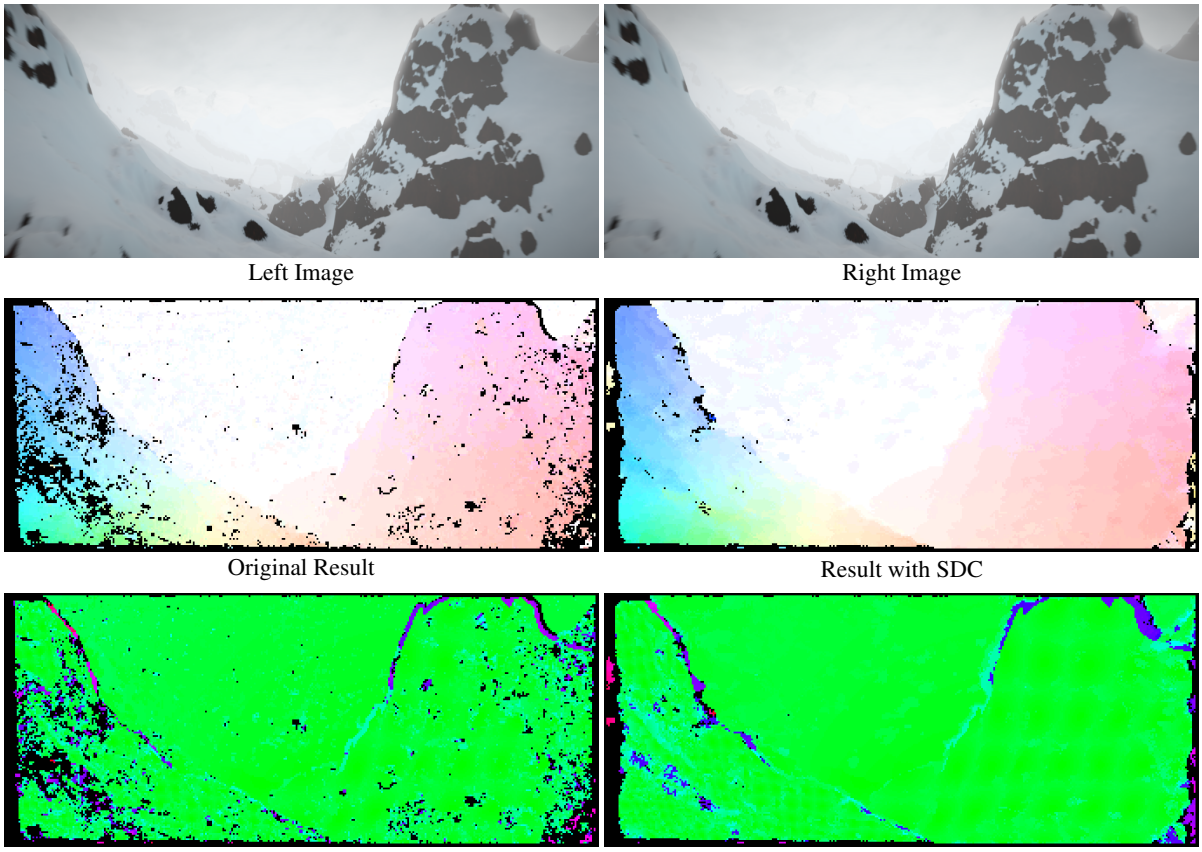


Error Map for Original Result



Error Map for Result with SDC

(h) Optical flow result for FlowFields++ [15] on KITTI [11]. Quantitative results are given in Table 3 of the main paper.



Left Image | Right Image

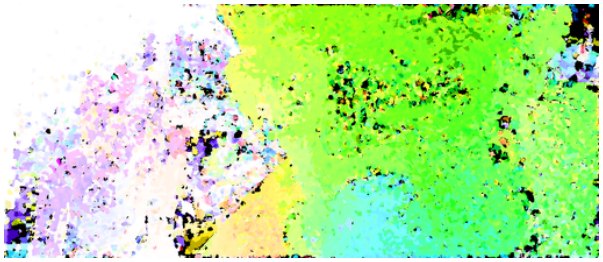
Original Result | Result with SDC

Error Map for Original Result | Error Map for Result with SDC

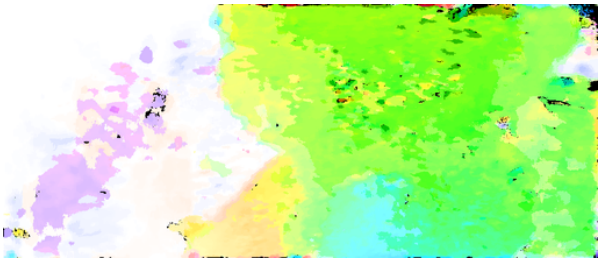
(i) Optical flow result for CPM [9] on Sintel [4]. Quantitative results are given in Table 4 of the main paper.

Left Image

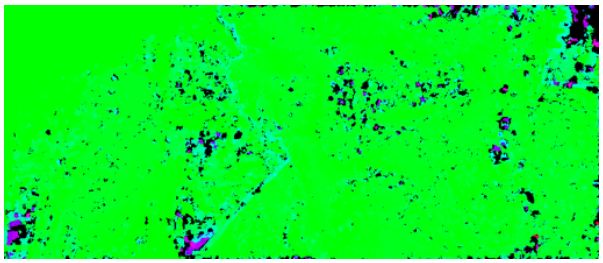
Right Image



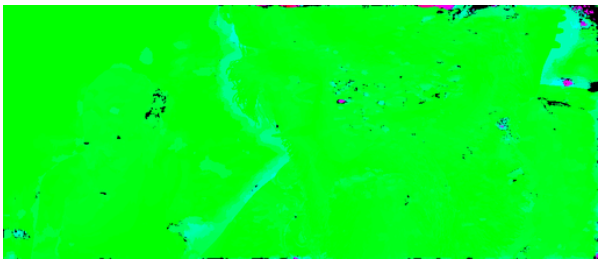
Original Result



Result with SDC

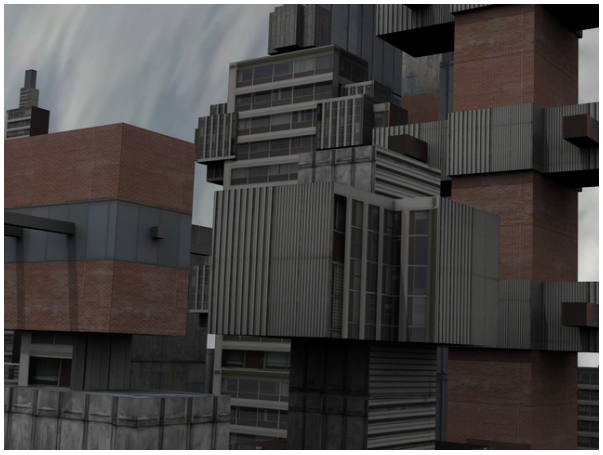


Error Map for Original Result

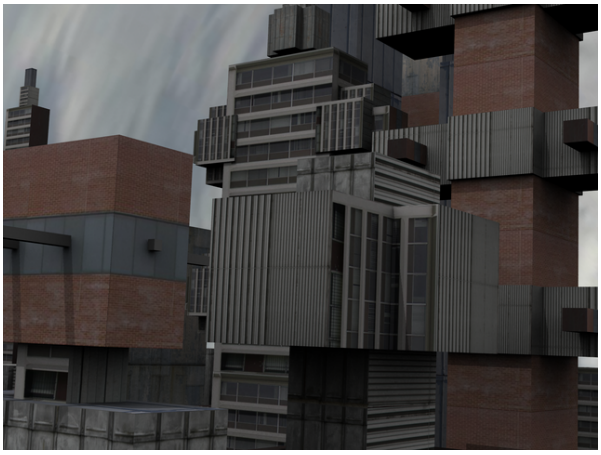


Error Map for Result with SDC

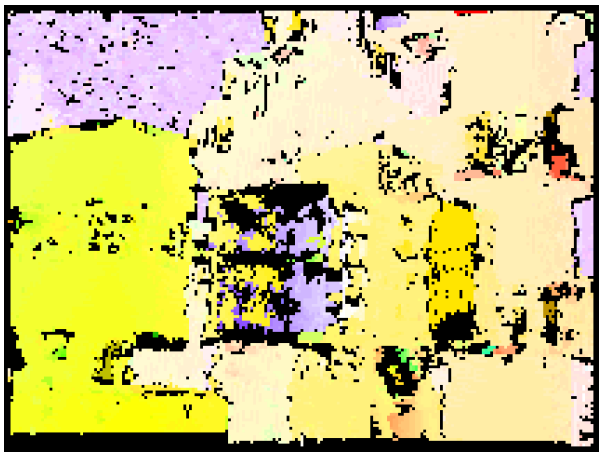
(j) Optical flow result for FlowFields++ [15] on Sintel [4]. Quantitative results are given in Table 3 of the main paper.



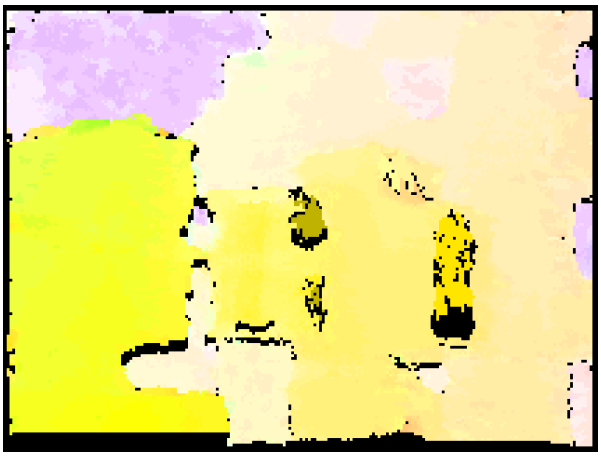
Left Image



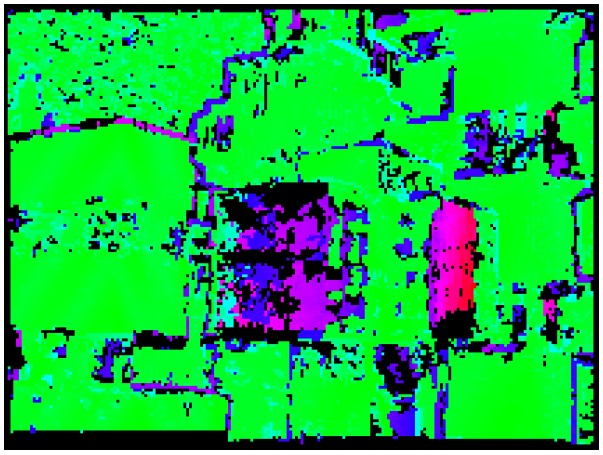
Right Image



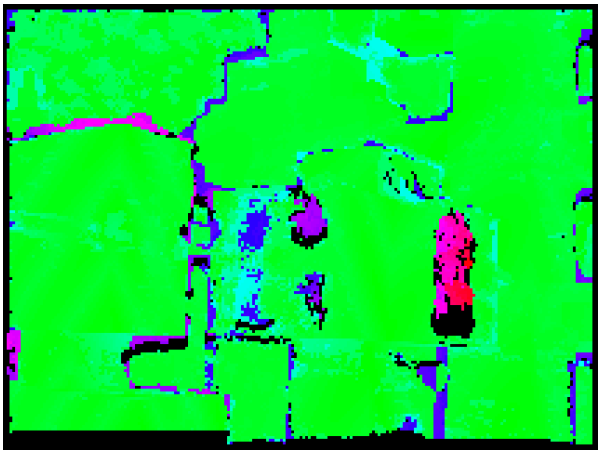
Original Result



Result with SDC



Error Map for Original Result

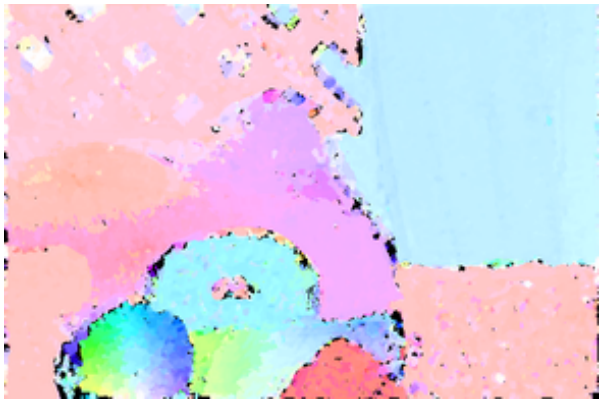


Error Map for Result with SDC

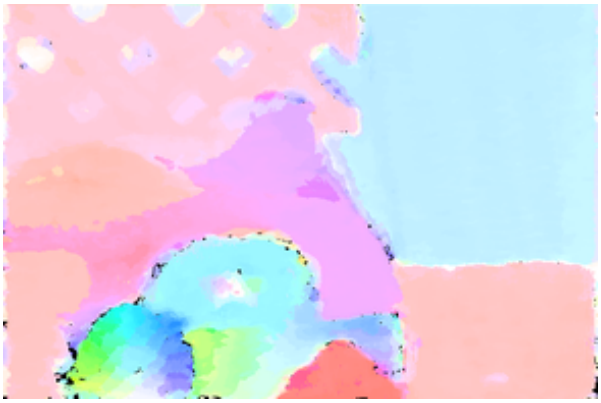
(k) Optical flow result for CPM [9] on Middlebury [2]. Quantitative results are given in Table 4 of the main paper.

Left Image

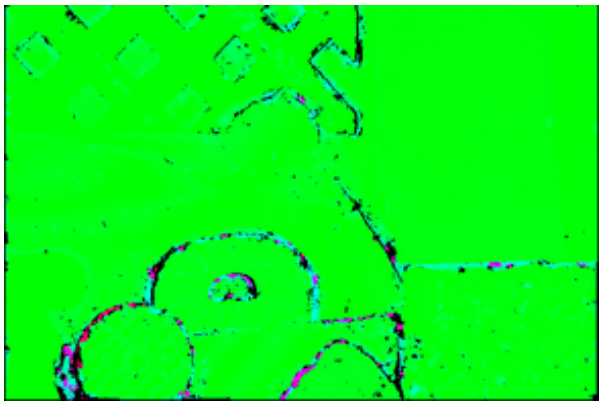
Right Image



Original Result



Result with SDC



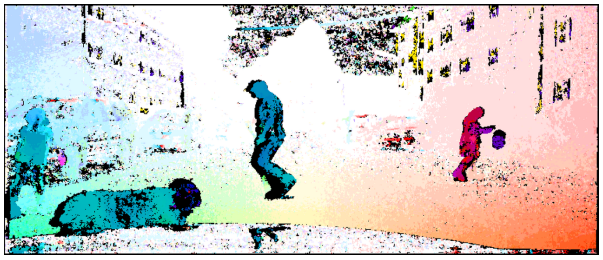
Error Map for Original Result

Error Map for Result with SDC

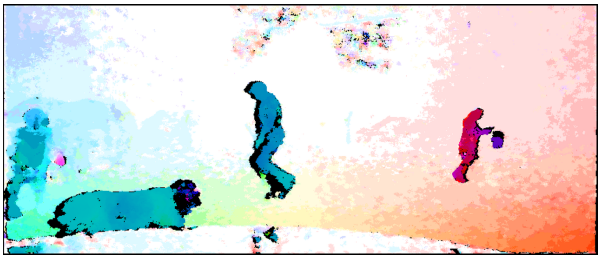
(l) Optical flow result for FlowFields++ [15] on Middlebury [2]. Quantitative results are given in Table 3 of the main paper.

Left Image

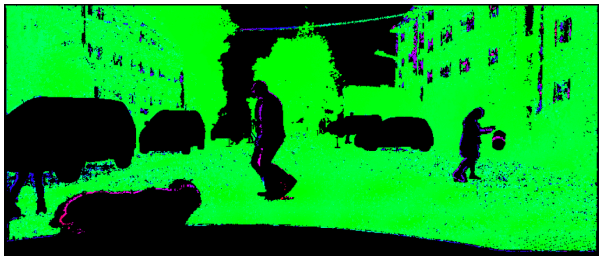
Right Image



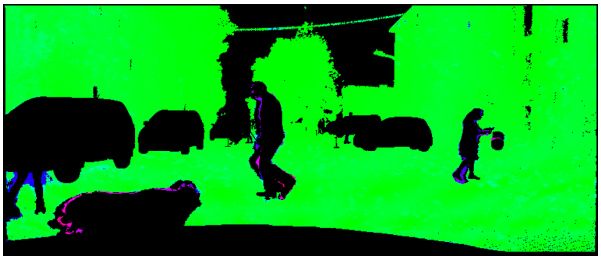
Original Result



Result with SDC



Error Map for Original Result

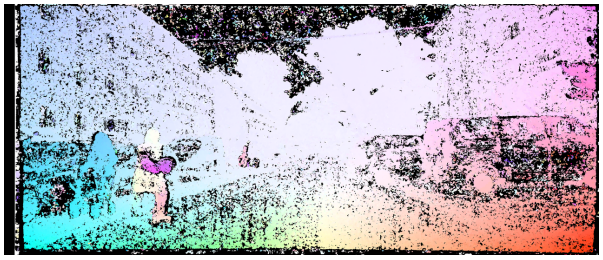


Error Map for Result with SDC

(m) Optical flow result CPM [9] on HD1K [10]. Quantitative results are given in Table 4 of the main paper.

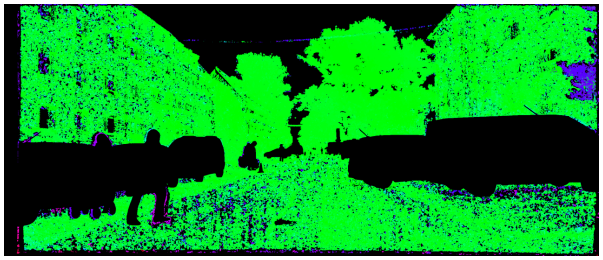
Left Image

Right Image

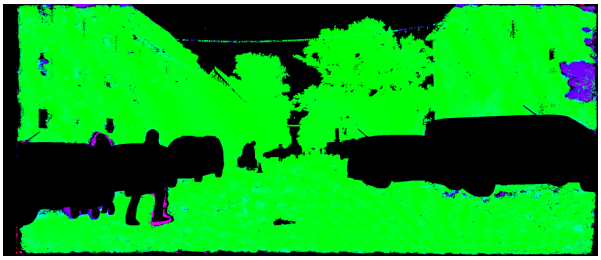


Original Result

Result with SDC



Error Map for Original Result

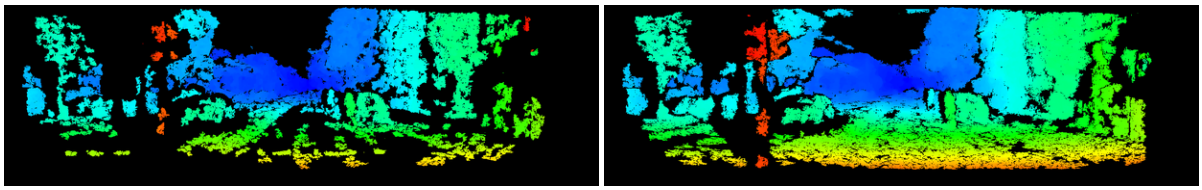


Error Map for Result with SDC

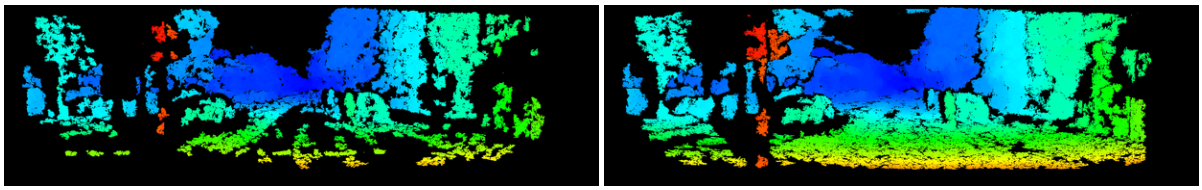
(n) Optical flow result for FlowFields++ [15] on HD1K [10]. Quantitative results are given in Table 3 of the main paper.

First Left Image | First Right Image

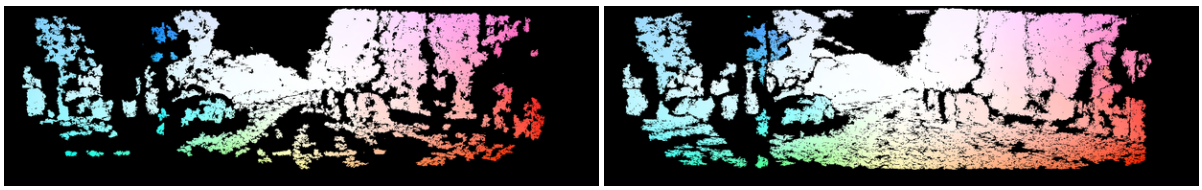
Second Left Image | Second Right Image



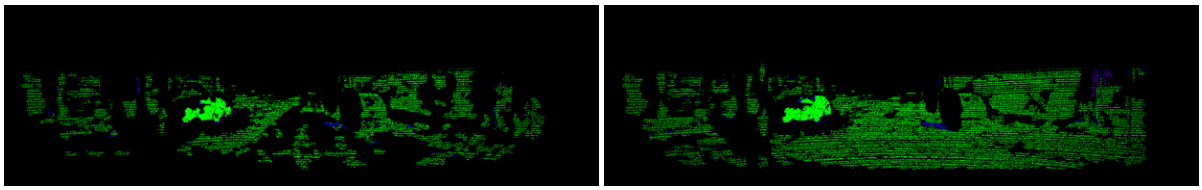
Original Result, Disparity 1 | Result with SDC, Disparity 1



Original Result, Disparity 2 | Result with SDC, Disparity 2



Original Result, Optical Flow | Result with SDC, Optical Flow



Error Map for Original Result | Error Map for Result with SDC

(o) Scene flow result for SceneFlowFields [16] on KITTI [11]. Quantitative results are given in Table 5 of the main paper.

Figure 10: Visual comparison of different matching algorithms for stereo disparity, optical flow, and scene flow on different data sets. For each combination, we show the original matching result and the results using our SDC feature descriptor. We do not change anything but the feature descriptor. With our SDC feature network, matching is more accurate (less outliers, sharper boundaries, smoother surface areas) and much denser (more matches over the complete image).