A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation: Supplementary Material

Figure 1. Bird's eye view of the *Driving* scene. The camera follows a convoluted path on street level and encounters many turns, crossings, other cars and varying lighting conditions.

1. Introduction

Due to space limitations in the paper, this supplemental material contains a more detailed description of the dataset generation process (Section 2) as well as more details and more qualitative results of DispNet (Section 3).

2. Dataset creation details

We modified the pipeline of Blender's internal render engine to produce – besides stereo RGB images – three additional data passes per frame and stereo view. Fig. 2 gives a visual breakdown of this data:

- In the base pass (3DPos$_t$), each pixel stores the true 3D position of the scene point which projects into that pixel (the 3D position is given within the camera coordinate system).

- For the second pass (3DPos$_{t-1}$), we revert time to the previous frame $t-1$ and save all vertices' 3D positions at that time. We then return to the current frame t and use the vertex 3D positions at time t to project the 3D vertices of time $t-1$ into image space. Hence, we again store 3D positions for each pixel, but this time *the 3D positions from time* $t-1$ *using the projection at time* t.

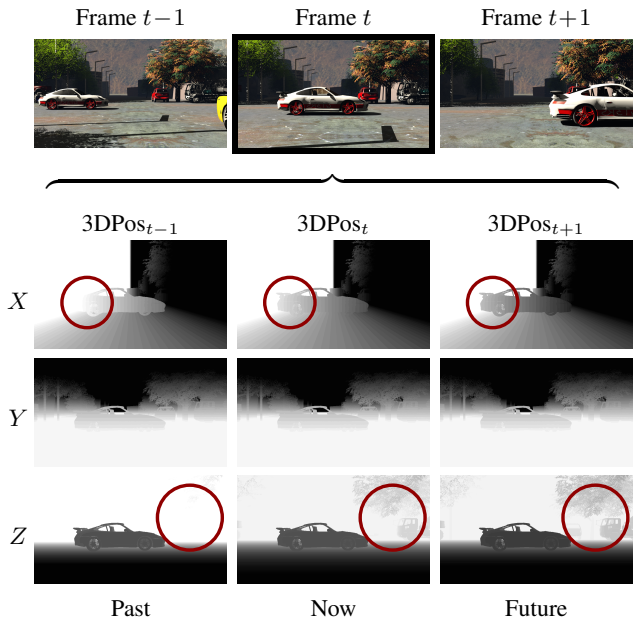

Figure 2. Our intermediate render data for frame t: The *X/Y/Z* channels encode the 3D positions (relative to the camera) of all visible points at frame t (center column) and what their respective 3D positions were/will be in the previous/next frame (left/right columns). The 3D positions of the previous and next frame are stored at the same image locations as in frame t. Hence, analyzing a location from frame t gives information about the past, current and future 3D position of the corresponding 3D point. All scene flow data can then be derived from this information. For example, the car moving to the right changes its X values (note that the perspective projection compresses the intensity gradient of the distant sky into an apparent step at $X = 0$). Nothing is moving vertically, so all Y values are constant over time. The camera is moving forward and all Z values change uniformly (note how objects on the right side become visible).

- The third pass (3DPos$_{t+1}$) is analogous to the second pass, except that this time we use the subsequent frame $t+1$ instead of the previous frame $t-1$.

These three data structures contain all information about the 3D structure and 3D motion of the scene as seen from the current viewpoint. From the 3DPos data we generate the

scene flow data. Fig. 3 describes the data conversion steps from the blender output to the resulting dataset. Note that color images and segmentation masks are directly produced by Blender and do not need any post-processing. Together with the camera intrinsics and extrinsics, various data can be generated, including calibrated RGBD images.

Fig. 4 shows example segmentation masks for a frame from one of our datasets. Materials can be shared across objects, but the combination of object indices and material indices yields a unique oversegmentation of a scene (consistent across all frames of the scene). While our experiments do not make use of these data, for other applications we also include the object and material IDs in our dataset.

With this supplemental material, we also provide a video that demonstrates the datasets we created and the final outcome of the pipeline, i.e. optical flow, disparity, disparity change and object and material index ground truth.

3. DispNetCorr

Intuitively, the simple *DispNet* disparity estimation architecture (as described in the main paper) has to learn the concept of matching parts of different images in rectified stereo images from scratch. Since the structure of the problem is well known (correspondences can only be found in accordance with the epipolar geometry [2]), we introduced an alternative architecture – the *DispNetCorr* – in which we explicitly correlate features along horizontal scanlines.

While the *DispNet* uses two stacked RGB images as a single input (i.e. one six-channel input blob), the *DispNetCorr* architecture first processes the input images separately, then correlates features between the two images and further processes the result. This behavior is similar to the correlation architecture used in [1] where Dosovitskiy et al. constructed a 2D correlation layer with limited neighborhood size and different striding in each of the images. For disparity estimation, we can use a simpler approach without striding and with larger neighborhood size, because the correlation along one dimension is computationally less demanding. One can additionally reduce the amount of comparisons by limiting the search to only one direction. For example, if we are given a left camera image and look for correspondences within the right camera image, then all disparity displacements are to the left.

Given two feature blobs $\mathbf{a}$ and $\mathbf{b}$ with multiple channels and identical sizes, we compute a correlation map of the same width and height, but with D channels, where D is the number of possible disparity values. For one pixel at location (x, y) in the first feature blob $\mathbf{a}$, the resulting correlation entry at channel $d \in [0, D-1]$ is the scalar product of the two feature vectors $\mathbf{a}_{(x,y)}$ and $\mathbf{b}_{(x-d,y)}$.

Color image | Object indices | Material indices

Figure 4. Segmentation data: object indices are unique per scene. Material indices can be shared across objects, but can be combined with the object indices to yield an oversegmentation into parts.

4. Qualitative examples

We show a qualitative evaluation of our networks for disparity estimation and compare them to other approaches in Figures 5 to 10.

References

[1] A. Dosovitskiy, P. Fischer, 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.

[2] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

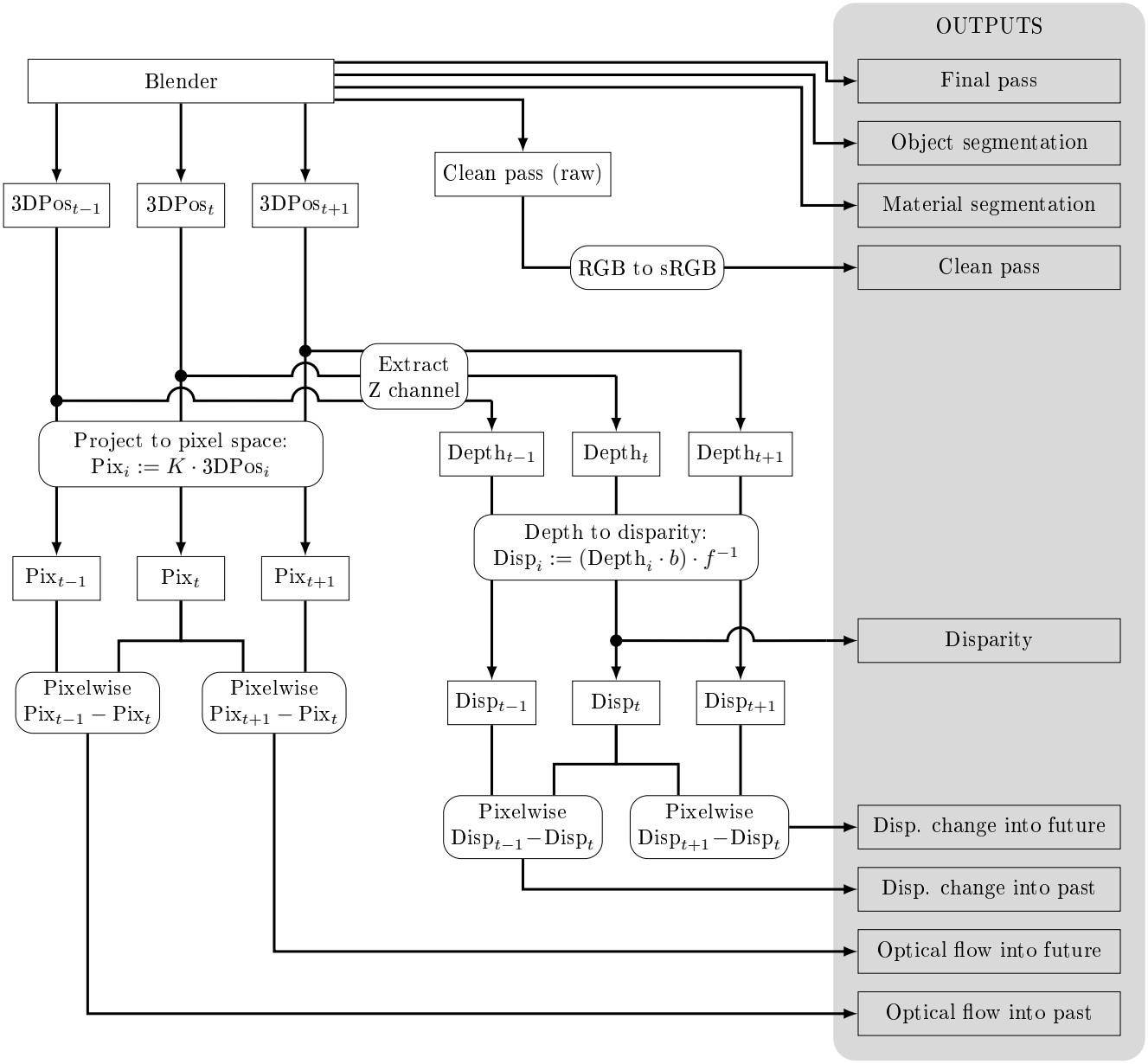

Figure 3. Data generation overview for a single view at frame time t: Blender directly outputs the *Final pass* and *Clean pass* images, as well as the object-level and material-level *segmentation masks*. *Disparity* is directly obtained from depth, which is given by the Z channel of the current 3DPos map as described in Fig. 2 (b is the stereo baseline, f denotes the focal length). Subtracting the current disparity map from the future/past disparity map results in the *disparity change* in future/past direction. The original 3DPos images are projected from camera space into pixel space using the camera intrinsics matrix K. Subtracting the current pixel position image from the future/past pixel position images yields the *optical flow* into the future/past.

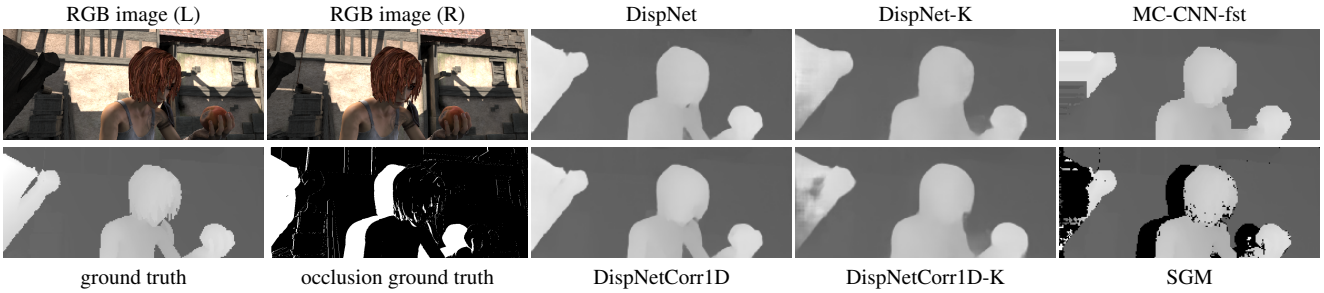

Figure 5. Disparities on a Sintel frame: DispNet and DispNetCorr1D fill the occluded regions in a much more reasonable way compared to other approaches.

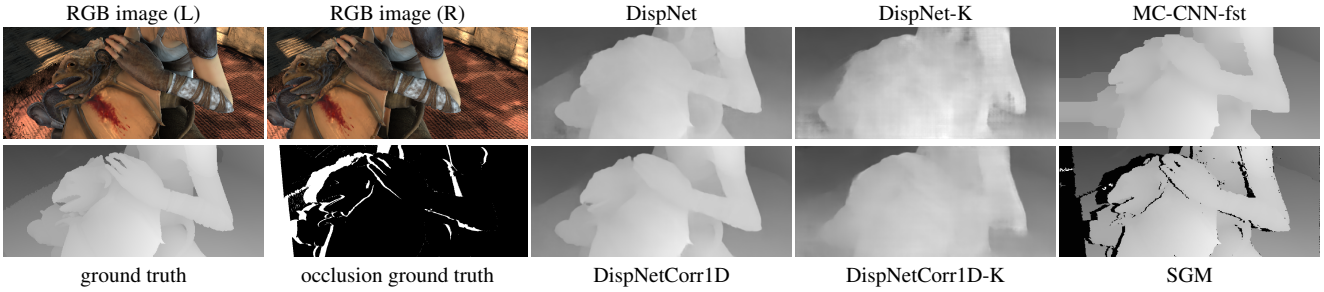

Figure 6. Disparities on a Sintel frame: DispNetCorr1D provides sharper estimates and the smooth areas on the dragon head are estimated better than with DispNet.

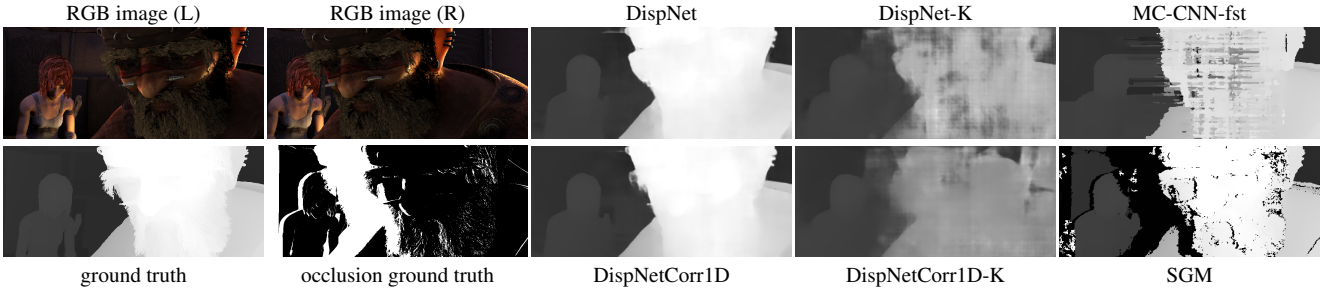

Figure 7. Disparities on a Sintel frame: The networks finetuned on the KITTI 2015 dataset cannot estimate large disparities anymore (large disparities are not present in KITTI). MC-CNN-fst has problems with large disparities, too.

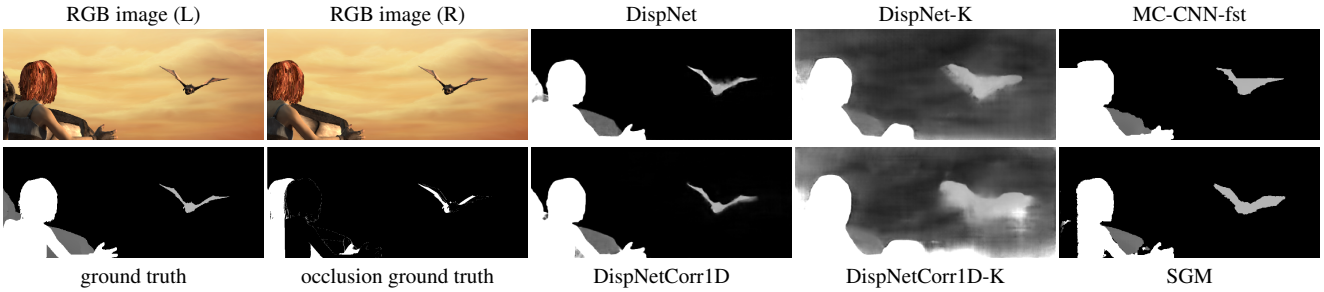

Figure 8. Disparities on a Sintel frame: DispNet and DispNetCorr1D can handle occluded regions in a nice way. After finetuning on KITTI 2015 the networks fail in the sky region (ground truth for sky and other small disparities is not available in KITTI).

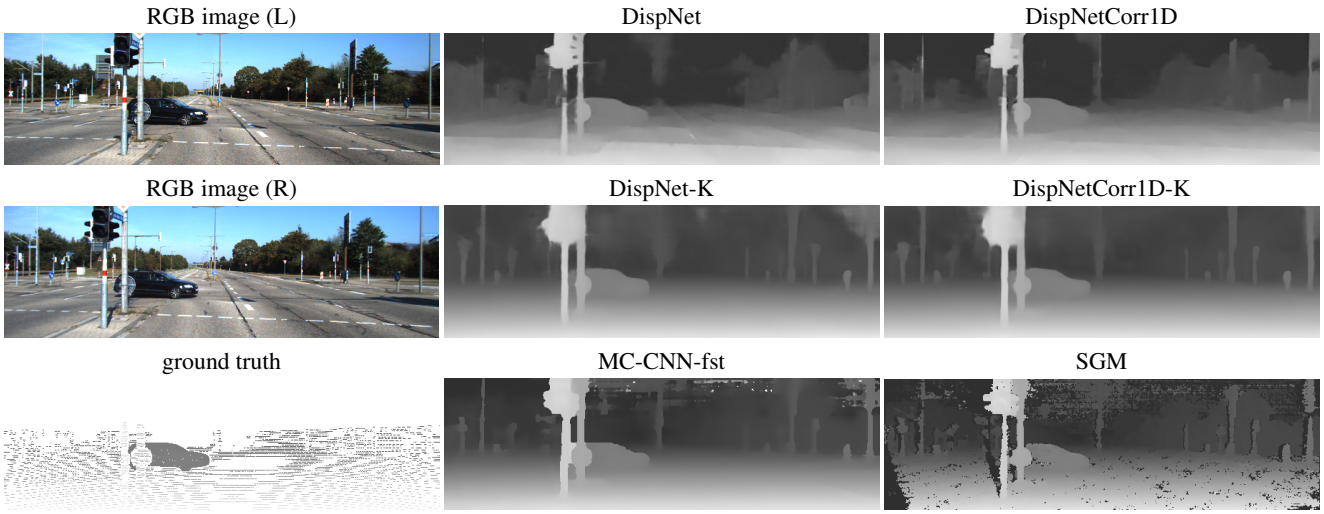

Figure 9. Disparities on a KITTI 2015 frame: The sparsity of the KITTI 2015 dataset leads to very smooth predictions when finetuning a network with such ground truth. While the non-finetuned DispNet and DispNetCorr1D estimate fine details accurately, they are less accurate in the smooth road and ground regions which are very common in KITTI.

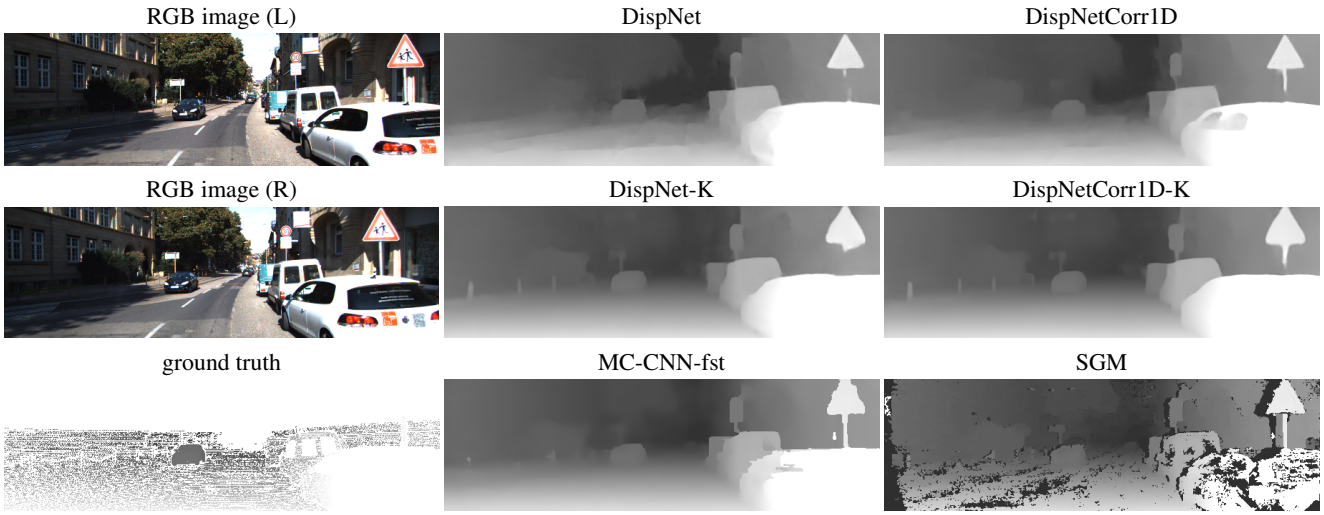

Figure 10. Disparities on a KITTI 2015 frame: Finetuning the networks on KITTI leads to much smoother estimates. However, DispNet-K and DispNetCorr1D-K can still recognize the delineator posts in the bottom left, which DispNet and DispNetCorr1D ignore completely. This shows that the finetuned networks do not simply oversmooth, but are still able to find small structures and disparity discontinuities.