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

Nikolaus Mayer*[1], Eddy Ilg*[1], Philip Häusser*[2], Philipp Fischer*[1]†
[1]University of Freiburg [2]Technical University of Munich
[1]{mayern,ilg,fischer}@cs.uni-freiburg.de [2]haeusser@cs.tum.edu

Daniel Cremers
Technical University of Munich
cremers@tum.de

Alexey Dosovitskiy, Thomas Brox
University of Freiburg
{dosovits,brox}@cs.uni-freiburg.de

Abstract

Recent work has shown that optical flow estimation can be formulated as a supervised learning task and can be successfully solved with convolutional networks. Training of the so-called FlowNet was enabled by a large synthetically generated dataset. The present paper extends the concept of optical flow estimation via convolutional networks to disparity and scene flow estimation. To this end, we propose three synthetic stereo video datasets with sufficient realism, variation, and size to successfully train large networks. Our datasets are the first large-scale datasets to enable training and evaluation of scene flow methods. Besides the datasets, we present a convolutional network for real-time disparity estimation that provides state-of-the-art results. By combining a flow and disparity estimation network and training it jointly, we demonstrate the first scene flow estimation with a convolutional network.

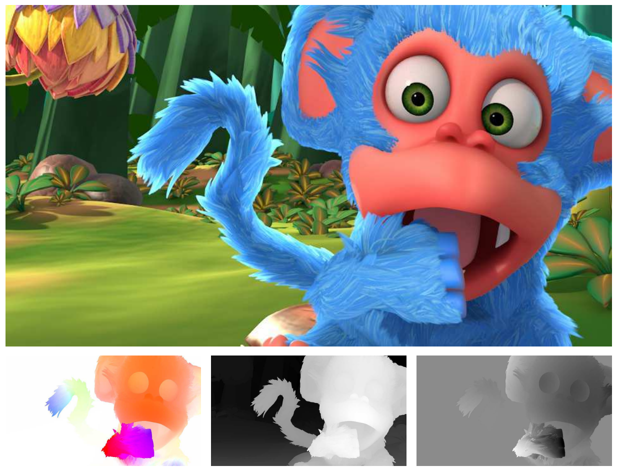

Figure 1. Our datasets provide over 35000 stereo frames with dense ground truth for *optical flow*, *disparity* and *disparity change*, as well as other data such as object segmentation.

1. Introduction

Estimating scene flow means providing the depth and 3D motion vectors of all visible points in a stereo video. It is the "royal league" task when it comes to reconstruction and motion estimation and provides an important basis for numerous higher-level challenges such as advanced driver assistance and autonomous systems. Research over the last decades has focused on its subtasks, namely disparity estimation and optical flow estimation, with considerable success. The full scene flow problem has not been explored to the same extent. While partial scene flow can be simply assembled from the subtask results, it is expected that the joint estimation of all components would be advantageous with regard to both efficiency and accuracy. One reason for scene flow being less explored than its subtasks seems to be a shortage of fully annotated ground truth data.

The availability of such data has become even more important in the era of convolutional networks. Dosovitskiy et al. [4] showed that optical flow estimation can be posed as a supervised learning problem and can be solved with a large network. For training their network, they created a simple synthetic 2D dataset of flying chairs, which proved to be sufficient to predict accurate optical flow in general videos. These results suggest that also disparities and scene flow can be estimated via a convolutional network, ideally jointly, efficiently, and in real-time. What is missing to implement this idea is a large dataset with sufficient realism and variability to train such a network and to evaluate its performance.

*These authors contributed equally
†Supported by the Deutsche Telekom Stiftung

Dataset	MPI Sintel [2]	KITTI Benchmark Suite [16]		SUN3D[26]	NYU2[21]	Ours		
		2012	2015			FlyingThings3D	Monkaa	Driving
#Training frames	1064	194	200†	2.5M	1449	21818	8591	4392
#Test frames	564	195	200†	—	—	4248	—	—
#Training scenes	25	194	200	415	464	2247	8	1
Resolution	1024×436	1226×370	1242×375	640×480	640×480	960×540	960×540	960×540
Disparity/Depth	✓	sparse	sparse	✓	✓	✓	✓	✓
Disparity change	✗	✗	✗	✗	✗	✓	✓	✓
Optical flow	✓	(sparse)	(sparse)	✗	✗	✓	✓	✓
Segmentation	✓	✗	✗	(✓)	✓	✓	✓	✓
Motion boundaries	✓	✗	✗	✗	✗	✓	✓	✓
Naturalism	(✓)	✓	✓	✓	✓	✗	✗	(✓)

Table 1. Comparison of available datasets: our new collection offers more annotated data and greater data variety than any existing choice. All our data has fully contiguous, dense, accurate ground truth. †Note that in KITTI 2015, a scene is a sequence of 21 stereo pairs, but groundtruth is only provided for a single frame.

In this paper, we present a collection of three such datasets, made using a customized version of the open source 3D creation suite Blender[3]. Our effort is similar in spirit to the Sintel benchmark [2]. In contrast to Sintel, our dataset is large enough to facilitate training of convolutional networks, and it provides ground truth for scene flow. In particular, it includes stereo color images and ground truth for bidirectional disparity, bidirectional optical flow and disparity change, motion boundaries, and object segmentation. Moreover, the full camera calibration and 3D point positions are available, i.e. our dataset also covers RGBD data. The datasets are freely available online[4].

We cannot exploit the full potential of this dataset in a single paper, but we already demonstrate various usage examples in conjunction with convolutional network training. We train a network for disparity estimation, which yields competitive performance also on previous benchmarks, especially among those methods that run in real-time. Finally, we also present a network for scene flow estimation and provide the first quantitative numbers on full scene flow on a sufficiently sized test set.

2. Related work

Datasets. The first significant efforts to create standard datasets were the Middlebury datasets for stereo disparity estimation [20] and optical flow estimation [1]. While the stereo dataset consists of real scenes, the optical flow dataset is a mixture of real scenes and rendered scenes. Both datasets are very small in today's terms. Especially the small test sets have led to heavy manual overfitting. An advantage of the stereo dataset is the availability of relevant real scenes, especially in the latest high-resolution version from 2014 [19].

MPI Sintel [2] is an entirely synthetic dataset derived from a short open source animated 3D movie. It provides dense ground truth for optical flow. Since very recently, a beta testing version with disparities is available for training. With 1064 training frames, the Sintel dataset is the largest dataset currently available. It contains sufficiently realistic scenes including natural image degradations such as fog and motion blur. The authors put much effort into the correctness of the ground truth for all frames and pixels. This makes the dataset a very reliable test set for comparison of methods. However, for training convolutional networks, the dataset is still too small.

The KITTI dataset was produced in 2012 [8] and extended in 2015 [16]. It contains stereo videos of road scenes from a calibrated pair of cameras mounted on a car. Ground truth for optical flow and disparity is obtained from a 3D laser scanner combined with the egomotion data of the car. While the dataset contains real data, the acquisition method restricts the ground truth to static parts of the scene. Moreover, the laser only provides sparse data up to a certain distance and height. For the most recent version, 3D models of cars were fitted to the point clouds to obtain a denser labeling and to also include moving objects. However, the ground truth in these areas is still an approximation.

Dosovitskiy et al. [4] trained convolutional networks for optical flow estimation on a synthetic dataset of moving 2D chair images superimposed on natural background images. This dataset is large but limited to single-view optical flow and does *not* contain any 3D motions.

Both the latest Sintel dataset and the KITTI dataset can be used to estimate scene flow with some restrictions. In occluded areas (visible in one frame but not in the other), ground truth for scene flow is not available. On KITTI, the most interesting component of scene flow, namely the 3D motion of foreground points, is missing or approximated via fitted CAD models of cars. A comprehensive overview of the most important comparable datasets and their features is given in Table 1.

Convolutional networks. Convolutional networks [15] have proven very successful for a variety of recognition

[3]https://www.blender.org/
[4]http://lmb.informatik.uni-freiburg.de/resources/datasets/

tasks, such as image classification [14]. Recent applications of convolutional networks include also depth estimation from single images [6], stereo matching [27], and optical flow estimation [4].

The FlowNet of Dosovitskiy et al. [4] is most related to our work. It uses an encoder-decoder architecture with additional crosslinks between contracting and expanding network parts, where the encoder computes abstract features from receptive fields of increasing size, and the decoder reestablishes the original resolution via an expanding upconvolutional architecture [5]. We adapt this approach for disparity estimation.

The disparity estimation method in Žbontar et al. [27] uses a Siamese network for computing matching distances between image patches. To finally estimate the disparity, the authors then perform cross-based cost aggregation [28] and semi-global matching (SGM) [10]. In contrast to our work, Žbontar et al. have no end-to-end training of a convolutional network on the disparity estimation task, with corresponding consequences for computational efficiency and elegance.

Scene flow. While there are hundreds of papers on disparity estimation and optical flow estimation, there are only a few on scene flow. None of them uses a learning approach.

Scene flow estimation was popularized for the first time by the work of Vedula et al. [22] who analyzed different possible problem settings. Later works were dominated by variational methods. Huguet and Devernay [11] formulated scene flow estimation in a joint variational approach. Wedel et al. [25] followed the variational framework but decoupled the disparity estimation for larger efficiency and accuracy. Vogel et al. [24] combined the task of scene flow estimation with superpixel segmentation using a piecewise rigid model for regularization. Quiroga et al. [17] extended the regularizer further to a smooth field of rigid motion. Like Wedel et al. [25] they decoupled the disparity estimation and replaced it by the depth values of RGBD videos.

The fastest method in KITTI's scene flow top 7 is from Cech et al. [3] with a runtime of 2.4 seconds. The method employs a seed growing algorithm for simultaneous disparity and optical flow estimation.

3. Definition of scene flow

Optical flow is a projection of the world's 3D motion onto the image plane. Commonly, *scene flow* is considered as the underlying 3D motion field that can be computed from stereo videos or RGBD videos. Assume two successive time frames t and $t+1$ of a stereo pair, yielding four images (I_L^t, I_R^t, I_L^{t+1}, I_R^{t+1}). Scene flow provides for each visible point in one of these four images the point's 3D position and its 3D motion vector [23].

These 3D quantities can be computed only in the case of known camera intrinsics and extrinsics. A camera-independent definition of scene flow is obtained by the separate components *optical flow*, the *disparity*, and the *disparity change* [11], cf. Fig. 2. This representation is complete in the sense that the visible 3D points and their 3D motion vectors can be computed from the components if the camera parameters are known.

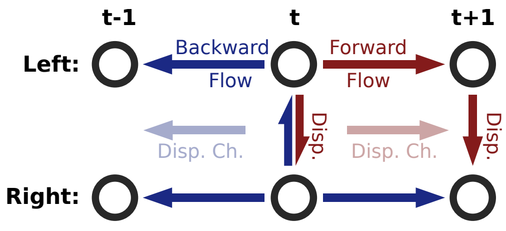

Figure 2. Given stereo images at times $t-1$, t and $t+1$, the arrows indicate disparity and flow relations between them. The red components are commonly used to estimate scene flow. In our datasets we provide all relations including the blue arrows.

Given the disparities at t and $t+1$, the disparity change is almost redundant. Thus, in the KITTI 2015 scene flow benchmark [16], only optical flow and disparities are evaluated. In this case, scene flow can be reconstructed only for surface points that are visible in both the left and the right frame. Especially in the context of convolutional networks, it is particularly interesting to estimate also depth and motion in partially occluded areas. Moreover, reconstruction of the 3D motions from flow and disparities is more sensitive to noise, because a small error in the optical flow can lead to a large error in the 3D motion vector.

4. Three rendered datasets

We created a synthetic dataset suite that consists of three subsets and provides the complete ground truth scene flow (incl. disparity change) in forward and backward direction. To this end, we used the open source 3D creation suite Blender to animate a large number of objects with complex motions and to render the results into tens of thousands of frames. We modified the pipeline of Blender's internal render engine to produce – besides stereo RGB images – three additional data passes per frame and view. These provide 3D positions of all visible surface points, as well as their future and past 3D positions. The pixelwise difference between two such data passes for a given camera view results in an "image" of 3D motion vectors – the complete scene flow ground truth as seen by this camera. Note that the information is complete even in occluded regions since the render engine always has full knowledge about all (visible and invisible) scene points.

All non-opaque materials – notably, most car windows – were rendered as fully transparent to avoid consistency problems in the 3D data. To prevent layer blending artifacts, we rendered all non-RGB data without antialiasing.

Given the intrinsic camera parameters (focal length, principal point) and the render settings (image size, virtual sensor size and format), we project the 3D motion vector of each pixel into a 2D pixel motion vector coplanar to the imaging plane: the optical flow. Depth is directly retrieved from a pixel's 3D position and converted to disparity using the known configuration of the virtual stereo rig. We compute the *disparity change* from the depth component of the 3D motion vector. Examples are shown in Fig. 1,3,8.

In addition, we rendered object segmentation masks in which each pixel's value corresponds to the unique index of its object. Objects can consist of multiple subparts, of which each can have a separate *material* (with own appearance properties such as textures). We make use of this and render additional segmentation masks, where each pixel encodes its material's index. The recently available beta version of Sintel also includes this data.

Similar to the Sintel dataset, we also provide object and material segmentations, as well as *motion boundaries* which highlight pixels between at least two moving objects, if the following holds: the difference in motion between the objects is at least 1.5 pixels, and the boundary segment covers an area of at least 10 pixels. The thresholds were chosen to match the results of Sintel's segmentation.

For all frames and views, we provide the full camera intrinsics and extrinsics matrices. Those can be used for structure from motion or other tasks that require camera tracking. We rendered all image data using a virtual focal length of 35mm on a 32mm wide simulated sensor. For the *Driving* dataset we added a wide-angle version using a focal length of 15mm which is visually closer to the existing KITTI datasets.

Like the Sintel dataset, our datasets also include two distinct versions of every image: the *clean pass* shows colors, textures and scene lighting but no image degradations, while the *final pass* additionally includes postprocessing effects such as simulated depth-of-field blur, motion blur, sunlight glare, and gamma curve manipulation.

To handle the massive amount of data (2.5TB), we compressed all RGB image data to the lossy but high-quality WebP[5] format (we provide both WebP and lossless PNG versions). Non-RGB data was compressed losslessly.

4.1. FlyingThings3D

The main part of the new data collection consists of everyday objects flying along randomized 3D trajectories. We generated about 25000 stereo frames with ground truth data. Instead of focusing on a particular task (like KITTI) or enforcing strict naturalism (like Sintel), we rely on randomness and a large pool of rendering assets to generate orders of magnitude more data than any existing option, without running a risk of repetition or saturation. Data generation is fast, fully automatic, and yields dense accurate ground truth for the complete scene flow task. The motivation for creating this dataset is to facilitate training of large convolutional networks, which should benefit from the large variety.

[5]https://developers.google.com/speed/webp/

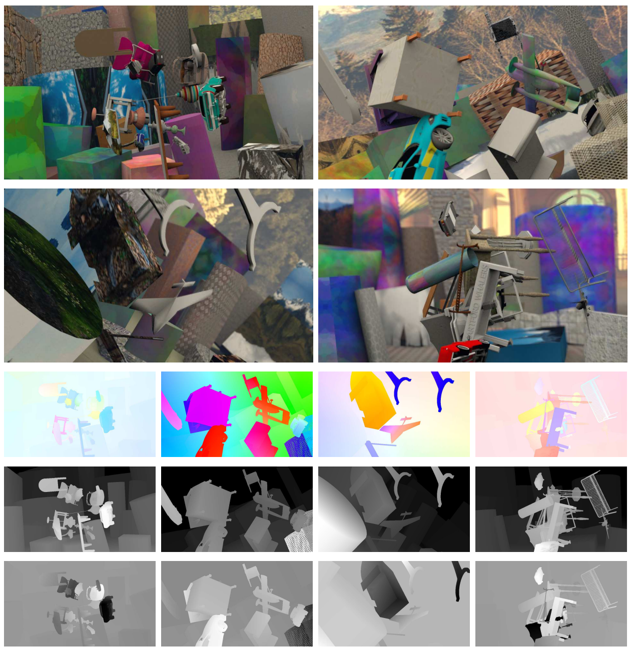
Figure 3. Example scenes from our *FlyingThings3D* dataset. **3rd row:** Optical flow images, **4th row:** Disparity images, **5th row:** Disparity change images. Best viewed on a color screen in high resolution (data images normalized for display).

The base of each scene is a large textured ground plane. We generated 200 static background objects with shapes that were randomly chosen from cuboids and deformed cylinders. Each object was randomly scaled, rotated, textured and then placed on the ground plane.

To populate the scene, we downloaded 35927 detailed 3D models from Stanford's ShapeNet[6] [18] database. From these we assembled a training set of 32872 models and a testing set of size 3055 (model categories are disjoint).

We sampled between 5 and 20 random objects from this object collection and randomly textured every material of every object. The camera and all ShapeNet objects were translated and rotated along linear 3D trajectories modeled such that the camera can see the objects, but with randomized displacements.

The texture collection was a combination of procedural images created using ImageMagick[7], landscape and cityscape photographs from Flickr[8], and texture-style pho-

[6]http://shapenet.cs.stanford.edu/

[7]http://www.imagemagick.org/script/index.php

[8]https://www.flickr.com/ Non-commercial public license. We used the code framework by Hays and Efros [9]

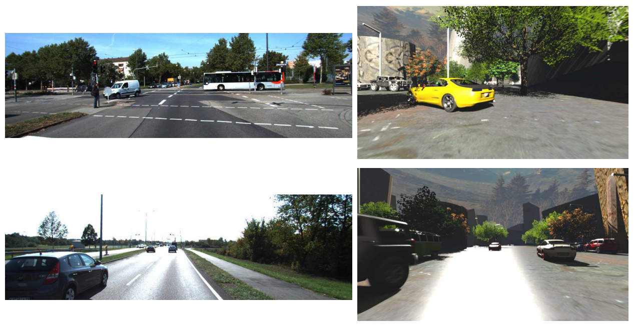

Figure 4. Example frames from the 2015 version of the KITTI benchmark suite [16] and our new *Driving* dataset. Both show many static and moving cars from various realistic viewpoints, thin objects, complex shadows, textured ground, and challenging specular reflections.

tographs from Image*After[9]. Like the 3D models, also the textures were split into disjoint training and testing parts.

For the final pass images, the scenes vary in presence and intensity of motion blur and defocus blur.

4.2. Monkaa

The second part of our dataset is made from the open source Blender assets of the animated short film Monkaa[10]. In this regard, it resembles the MPI Sintel dataset. Monkaa contains nonrigid and softly articulated motion as well as visually challenging fur. Beyond that, there are few visual similarities to Sintel; the Monkaa movie does not strive for the same amount of naturalism.

We selected a number of suitable movie scenes and additionally created entirely new scenes using parts and pieces from Monkaa. To increase the amount of data, we rendered our selfmade scenes in multiple versions, each with random incremental changes to the camera's rotation and motion path.

4.3. Driving

The *Driving* scene is a mostly naturalistic, dynamic street scene from the viewpoint of a driving car, made to resemble the KITTI datasets. It uses car models from the same pool as the *FlyingThings3D* dataset and additionally employs highly detailed tree models from 3D Warehouse[11] and simple street lights. In Fig. 4 we show selected frames from *Driving* and lookalike frames from KITTI 2015.

Our stereo baseline is set to 1 Blender unit, which together with a typical car model width of roughly 2 units is comparable to KITTI's setting (54cm baseline, 186cm car width [8]).

[9]http://www.imageafter.com/textures.php
[10]https://cloud.blender.org/bi/monkaa/
[11]https://3dwarehouse.sketchup.com/

5. Networks

To prove the applicability of our new synthetic datasets to scene flow estimation, we use them to train convolutional networks. In general, we follow the architecture of the FlowNet [4]: each network consists of a contractive part and an expanding part with long-range links between them. The contracting part contains convolutional layers with occasional strides of 2, resulting in a total downsampling factor of 64. This allows the network to estimate large displacements. The expanding part of the network then gradually and nonlinearly upsamples the feature maps, taking into account also the features from the contractive part. This is done by a series of up-convolutional and convolutional layers. Note that there is no data bottleneck in the network, as information can also pass through the long-range connections between contracting and expanding layers. For an illustration of the overall architecture we refer to the figures in Dosovitskiy et al. [4].

For disparity estimation we propose the basic architecture *DispNet* described in Table 2. We found that additional convolutions in the expanding part yield smoother disparity maps than the FlowNet architecture (see Fig. 6).

We also tested an architecture that makes use of an explicit correlation layer [4], which we call *DispNetCorr*. In this network, the two images are processed separately up to layer conv2 and the resulting features are then correlated horizontally. We consider a maximum displacement of 40 pixels, which corresponds to 160 pixels in the input image. Compared to the 2D correlation in Dosovitskiy et al. [4], 1D correlation is computationally much cheaper and allows us to cover larger displacements with finer sampling than in the FlowNet, which used a stride of 2 for the correlation.

We also train a FlowNet and a joint SceneFlowNet for scene flow estimation by combining and fine-tuning pretrained networks for disparity and flow. This is illustrated in Figure 5. A FlowNet predicts flow between the left and right image and two DispNets predict the disparities at t and $t+1$. The networks in this case do not contain correlation layers and convolutions between up-convolutions. We then fine-tune the large combined network to estimate flow, disparity, and additionally disparity change.

Training. All networks are trained end-to-end, given the images as input and the ground truth (optical flow, disparity, or scene flow) as output. We employ a custom version of Caffe [12] and make use of the Adam optimizer [13]. We set $\beta_1 = 0.9$ and $\beta_2 = 0.999$ as in Kingma et al. [13]. As learning rate we used $\lambda = 0.0001$ and divided it by 2 every 200k iterations starting from iteration 400k.

Due to the depth of the networks and the direct connections between contracting and expanding layers (see Table 2), lower layers get mixed gradients if all six losses are active. We found that using a loss weight schedule can be beneficial: we start training with a loss weight of 1 assigned

Method	KITTI 2012		KITTI 2015		Driving	FlyingThings3D	Monkaa	Sintel	Time
	train	test	train	test (D1)	clean	clean test	clean	clean train	
DispNet	2.38	—	2.19	—	15.62	2.02	5.99	5.38	0.06s
DispNetCorr1D	1.75	—	1.59	—	16.12	1.68	5.78	5.66	0.06s
DispNet-K	1.77	—	(0.77)	—	19.67	7.14	14.09	21.29	0.06s
DispNetCorr1D-K	1.48	$1.0^\dagger$	(0.68)	4.34%	20.40	7.46	14.93	21.88	0.06s
SGM	10.06	—	7.21	10.86%	40.19	8.70	20.16	19.62	1.1s
MC-CNN-fst	—	—	—	4.62%	19.58	4.09	6.71	11.94	0.8s
MC-CNN-acrt	—	0.9	—	3.89%	—	—	—	—	67s

Table 3. Disparity errors. All measures are endpoint errors, except for the *D1-all* measure (see the text for explanation) for KITTI 2015 test. †This result is from a network fine-tuned on KITTI 2012 train.

Name	Kernel	Str.	Ch I/O	InpRes	OutRes	Input
conv1	7×7	2	6/64	768×384	384×192	images
conv2	5×5	2	64/128	384×192	192×96	conv1
conv3a	5×5	2	128/256	192×96	96×48	conv2
conv3b	3×3	1	256/256	96×48	96×48	conv3a
conv4a	3×3	2	256/512	96×48	48×24	conv3b
conv4b	3×3	1	512/512	48×24	48×24	conv4a
conv5a	3×3	2	512/512	48×24	24×12	conv4b
conv5b	3×3	1	512/512	24×12	24×12	conv5a
conv6a	3×3	2	512/1024	24×12	12×6	conv5b
conv6b	3×3	1	1024/1024	12×6	12×6	conv6a
pr6+loss6	3×3	1	1024/1	12×6	12×6	conv6b
upconv5	4×4	2	1024/512	12×6	24×12	conv6b
iconv5	3×3	1	1025/512	24×12	24×12	upconv5+pr6+conv5b
pr5+loss5	3×3	1	512/1	24×12	24×12	iconv5
upconv4	4×4	2	512/256	24×12	48×24	iconv5
iconv4	3×3	1	769/256	48×24	48×24	upconv4+pr5+conv4b
pr4+loss4	3×3	1	256/1	48×24	48×24	iconv4
upconv3	4×4	2	256/128	48×24	96×48	iconv4
iconv3	3×3	1	385/128	96×48	96×48	upconv3+pr4+conv3b
pr3+loss3	3×3	1	128/1	96×48	96×48	iconv3
upconv2	4×4	2	128/64	96×48	192×96	iconv3
iconv2	3×3	1	193/64	192×96	192×96	upconv2+pr3+conv2
pr2+loss2	3×3	1	64/1	192×96	192×96	iconv2
upconv1	4×4	2	64/32	192×96	384×192	iconv2
iconv1	3×3	1	97/32	384×192	384×192	upconv1+pr2+conv1
pr1+loss1	3×3	1	32/1	384×192	384×192	iconv1

Table 2. Specification of DispNet architecture. The contracting part consists of convolutions *conv1* to *conv6b*. In the expanding part, upconvolutions (*upconvN*), convolutions (*iconvN*, *prN*) and loss layers are alternating. Features from lower layers are concatenated with higher layer features. The predicted disparity image is output by *pr1*.

to the lowest resolution loss *loss6* and a weight of 0 for all other losses (that is, all other losses are switched off). During training, we progressively increase the weights of losses with higher resolution and deactivate the low resolution losses. This enables the network to first learn a coarse representation and then proceed with finer resolutions without losses constraining intermediate features.

Data augmentation. Despite the large training set, we chose to perform data augmentation to introduce more diversity into the training data at almost no extra cost[12]. We perform spatial (rotation, translation, cropping, scaling) and chromatic transformations (color, contrast, brightness), and we use the same transformation for all 2 or 4 input images.

[12]The computational bottleneck is in reading the training samples from disk, whereas data augmentation is performed on the fly.

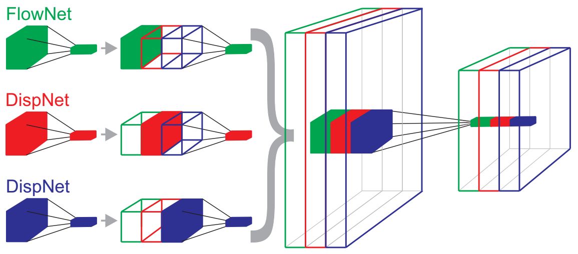

Figure 5. Interleaving the weights of a FlowNet (green) and two DispNets (red and blue) to a SceneFlowNet. For every layer, the filter masks are created by taking the weights of one network (left) and setting the weights of the other networks to zero, respectively (middle). The outputs from each network are then concatenated to yield one big network with three times the number of inputs and outputs (right).

For disparity, any rotation or vertical shift would break the epipolar constraint, and horizontal shifts between stereo views could lead to negative disparities.

6. Experiments

Evaluation of existing methods. We evaluated several existing disparity methods on our new dataset. Namely, for disparity we evaluate the state-of-the-art method of Žbontar and LeCun [27] and the popular Semi-Global Matching [10] approach with a block matching implementation from OpenCV[13]. Results are shown together with those of our DispNets in Table 3. We use the endpoint error (EPE) as error measure in most cases, with the only exception of KITTI 2015 test set where only the *D1-all* error measure is reported by the KITTI evaluation server (percentage of pixels with estimation error >3px and $>5\%$ of the true disparity).

DispNet. We train DispNets on the FlyingThings3D dataset and then optionally fine-tune on KITTI. The fine-

[13]http://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html#stereosgbm

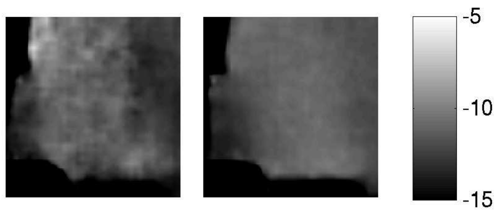

Figure 6. Close-up of a predicted disparity map without (left) and with (right) convolutions between up-convolutions. Note how the prediction on the right is much smoother.

tuned networks are denoted by a '-K' suffix in the table. At submission time, DispNetCorr fine-tuned on KITTI 2015 was second best in the KITTI 2015 top results table, slightly behind MC-CNN-acrt [27] but being roughly 1000 times faster. On KITTI resolution it runs at 15 frames per second on an Nvidia GTX Titan X GPU. For foreground pixels (belonging to car models) our error is roughly half that of [27]. The network achieves an error that is $\sim 30\%$ lower than the best real-time method reported in the table, Multi-Block-Matching [7]. Also on the other datasets DispNet performs well and outperforms both SGM and MC-CNN.

While fine-tuning on KITTI improves the results on this dataset, it increases errors on other datasets. We explain this significant performance drop by the fact that KITTI 2015 only contains relatively small disparities, up to roughly 150 pixels, while the other datasets contain some disparities of 500 pixels and more. When fine-tuned on KITTI, the network seems to lose its ability to predict large displacements, hence making huge errors on these.

We introduced several modifications to the network architecture compared to the FlowNet [4]. First, we added convolutional layers between up-convolutional layers in the expanding part of the network. As expected, this allows the network to better regularize the disparity map and predict smoother results, as illustrated in Fig. 6. The result is a $\sim 15\%$ relative EPE decrease on KITTI 2015.

Second, we trained a version of our network with a 1D correlation layer. In contrast to Dosovitskiy et al. [4], we find that networks with correlation in many cases improve the performance (see Table 3). A likely plausible explanation is that the 1D nature of the disparity estimation problem allows us to compute correlations at a finer grid than the FlowNet.

SceneFlowNet. As mentioned in Sec. 5, to construct a SceneFlowNet, we first train a FlowNet and a DispNet, then combine them as described in Fig. 5 and train the combination. Table 4 shows the results of the initial networks and the SceneFlowNet. We observe that solving the joint task yields better results than solving the individual tasks. The final results on our datasets are given in Table 5 and a qualitative example from FlyingThings3D is shown in Fig. 8.

	Flow	Disparity	Disp. Ch
FlowNet	13.78		
DispNet		2.41	
FlowNet +500k	12.18		
DispNet +500k		2.37	
SceneFlowNet +500k	10.99	2.21	0.79

Table 4. Performance of solving the single tasks compared to solving the joint scene flow task, trained and tested on FlyingThings3D. FlowNet was initially trained for 1.2M and DispNet for 1.4M iterations, +500k denotes training for 500k more iterations. The SceneFlowNet is initialized with the FlowNet and DispNet. Solving the joint task yields better results in each individual task.

SceneFlowNet	Driving	FlyingThings3D	Monkaa
Flow	23.53	10.99	6.54
Disparity	15.35	2.21	6.59
Disp. change	16.34	0.80	0.78

Table 5. Endpoint errors for the evaluation of our SceneFlowNet on the presented datasets. The Driving dataset contains the largest disparities, flows and disparity changes, resulting in large errors. The FlyingThings3D dataset contains large flows, while Monkaa contains smaller flows and larger disparities.

Although the FlyingThings3D dataset is more sophisticated than the FlyingChairs dataset, training on this dataset did not yield a FlowNet that performs better than training on FlyingChairs. Notwithstanding the fact that FlyingThings3D, in contrast to FlyingChairs, offers the possibility to train networks for disparity and scene flow estimation, we are investigating how 3D datasets can also improve the performance of FlowNet.

7. Conclusion

We have introduced a synthetic dataset containing over 35000 stereo image pairs with ground truth disparity, optical flow, and scene flow. While our motivation was to create a sufficiently large dataset that is suitable to train convolutional networks to estimate these quantities, the dataset can also serve for evaluation of other methods. This is particularly interesting for scene flow, where there has been a lack of datasets with ground truth.

We have demonstrated that the dataset can indeed be used to successfully train large convolutional networks: the network we trained for disparity estimation is on par with the state of the art and runs 1000 times faster. A first approach of training the network for scene flow estimation using a standard network architecture also shows promising results. We are convinced that our dataset will help boost deep learning research for such challenging vision tasks as stereo, flow and scene flow estimation.

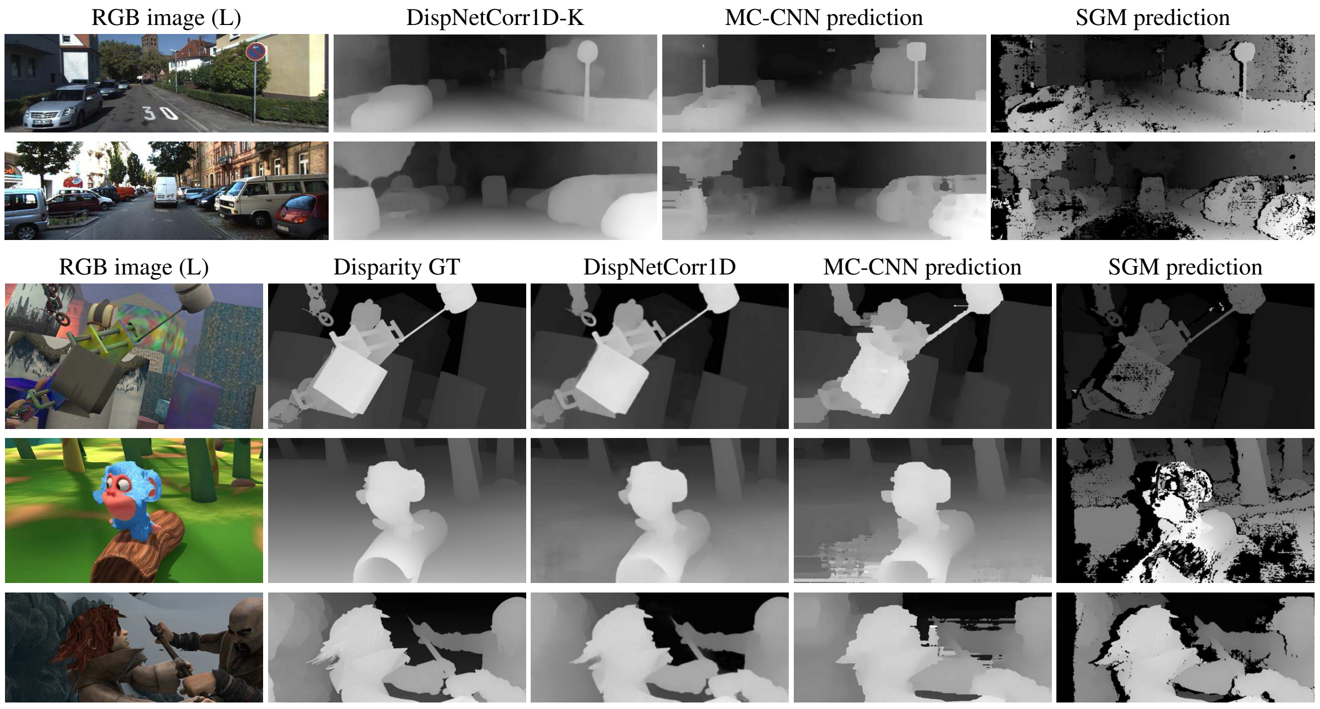

Figure 7. Disparity results. Rows from top to bottom: KITTI 2012, KITTI 2015, FlyingThings3D (clean), Monkaa (clean), Sintel (clean). Note how the DispNet prediction is basically noise-free.

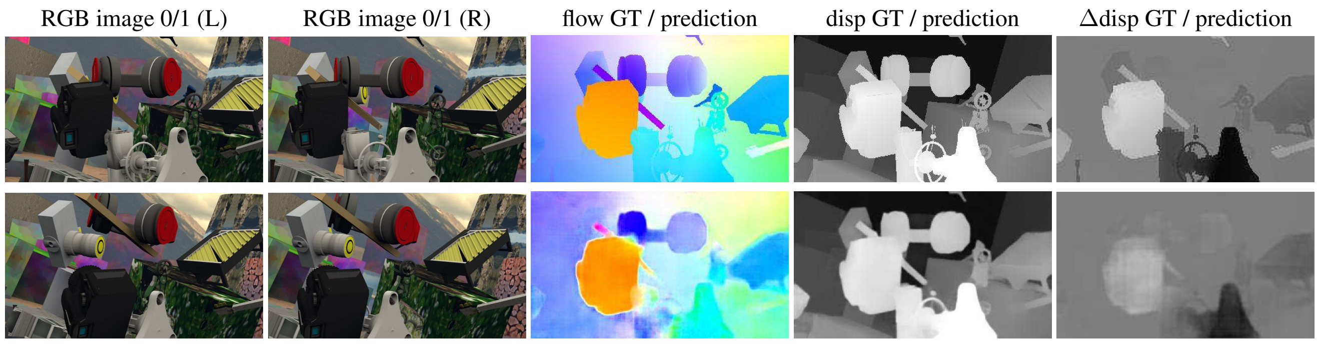

Figure 8. Results of our SceneFlowNet created from pretrained FlowNet and DispNets. The disparity change was added and the network was fine-tuned on FlyingThings3D for 500k iterations.

8. Acknowledgements

The work was partially funded by the ERC Starting Grant VideoLearn, the ERC Consolidator Grant 3D Reloaded, and by the DFG Grants BR 3815/7-1 and CR 250/13-1.

References

[1] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. Technical Report MSR-TR-2009-179, December 2009.

[2] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, Part IV, LNCS 7577, pages 611–625, Oct. 2012.

[3] J. Cech, J. Sanchez-Riera, and R. P. Horaud. Scene flow estimation by growing correspondence seeds. In *CVPR*, 2011.

[4] 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.

[5] A. Dosovitskiy, J. T. Springenberg, and T. Brox. Learning to generate chairs with convolutional neural networks. In *CVPR*, 2015.

[6] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction

from a single image using a multi-scale deep network. *NIPS*, 2014.

[7] N. Einecke and J. Eggert. A multi-block-matching approach for stereo. In *Intelligent Vehicles Symposium*, pages 585–592, 2015.

[8] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The KITTI dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[9] J. Hays and A. A. Efros. im2gps: estimating geographic information from a single image. In *CVPR*, 2008.

[10] H. Hirschmüller. Stereo processing by semiglobal matching and mutual information. *PAMI*, 30(2):328–341, 2008.

[11] F. Huguet and F. Deverney. A variational method for scene flow estimation from stereo sequences. In *ICCV*, 2007.

[12] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[13] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1106–1114, 2012.

[15] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[16] M. Menze and A. Geiger. Object scene flow for autonomous vehicles. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[17] J. Quiroga, F. Devernay, and J. Crowley. Scene flow by tracking in intensity and depth data. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, pages 50–57. IEEE, 2012.

[18] M. Savva, A. X. Chang, and P. Hanrahan. Semantically-Enriched 3D Models for Common-sense Knowledge. *CVPR 2015 Workshop on Functionality, Physics, Intentionality and Causality*, 2015.

[19] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nešić, X. Wang, and P. Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *Pattern Recognition*, pages 31–42. Springer, 2014.

[20] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 47(1-3):7–42, 2002.

[21] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[22] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade. Three-dimensional scene flow. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):475–480, 2005.

[23] S. Vedula, S. Baker, P. Rander, R. T. Collins, and T. Kanade. Three-dimensional scene flow. In *ICCV*, pages 722–729, 1999.

[24] C. Vogel, K. Schindler, and S. Roth. 3d scene flow estimation with a piecewise rigid scene model. *International Journal of Computer Vision*, 115(1):1–28, 2015.

[25] A. Wedel, T. Brox, T. Vaudrey, C. Rabe, U. Franke, and D. Cremers. Stereoscopic scene flow computation for 3d motion understanding. *International Journal of Computer Vision*, 95(1):29–51, 2010.

[26] J. Xiao, A. Owens, and A. Torralba. Sun3d: A database of big spaces reconstructed using sfm and object labels. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 1625–1632, Dec 2013.

[27] J. Žbontar and Y. LeCun. Stereo matching by training a convolutional neural network to compare image patches. *arXiv preprint arXiv:1510.05970*, 2015.

[28] K. Zhang, J. Lu, and G. Lafruit. Cross-based local stereo matching using orthogonal integral images. *IEEE Trans. Circuits Syst. Video Techn.*, 19(7):1073–1079, 2009.