ICCV 2015

FlowNet: Learning Optical Flow with Convolutional Networks

Alexey Dosovitskiy*, Philipp Fischer†*, Eddy Ilg*, Philip Häusser, Caner Hazırbaş, Vladimir Golkov†
University of Freiburg | Technical University of Munich
{fischer,dosovits,ilg}@cs.uni-freiburg.de, {haeusser,hazirbas,golkov}@cs.tum.edu

Patrick van der Smagt, Technical University of Munich, smagt@brml.org
Daniel Cremers, Technical University of Munich, cremers@tum.de
Thomas Brox, University of Freiburg, brox@cs.uni-freiburg.de

2020.09.22

Presented by Kyungmin Jo

DAVIAN

Introduction

- Task : Optical flow estimation
  - Optical flow estimation has not been among the tasks CNNs succeeded at. (in 2015)
  - Solve the optical flow estimation problem as a supervised learning task

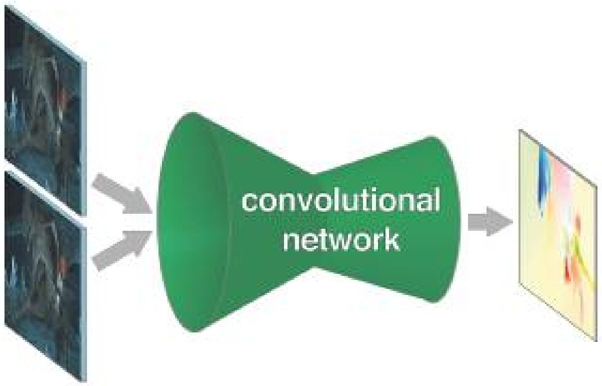


Figure 1. We present neural networks which learn to estimate optical flow, being trained end-to-end. The information is first spatially compressed in a contractive part of the network and then refined in an expanding part.

▲Example of optical flow

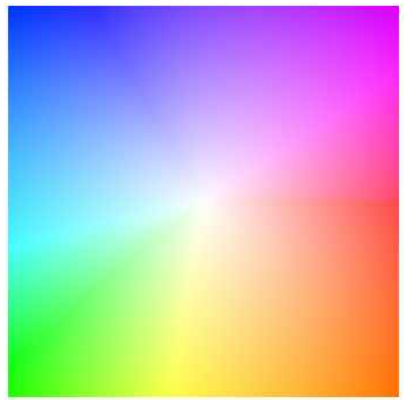


Figure 1. Flow field color coding. The central pixel does not move, and the displacement of every other pixel is the vector from the center to this pixel.

Main idea

- Take an end-to-end learning approach to predicting optical flow: given image pairs and ground truth flows
- Two types of architecture + Refinement
  - FlowNetSimple
  - FlowNetCorr

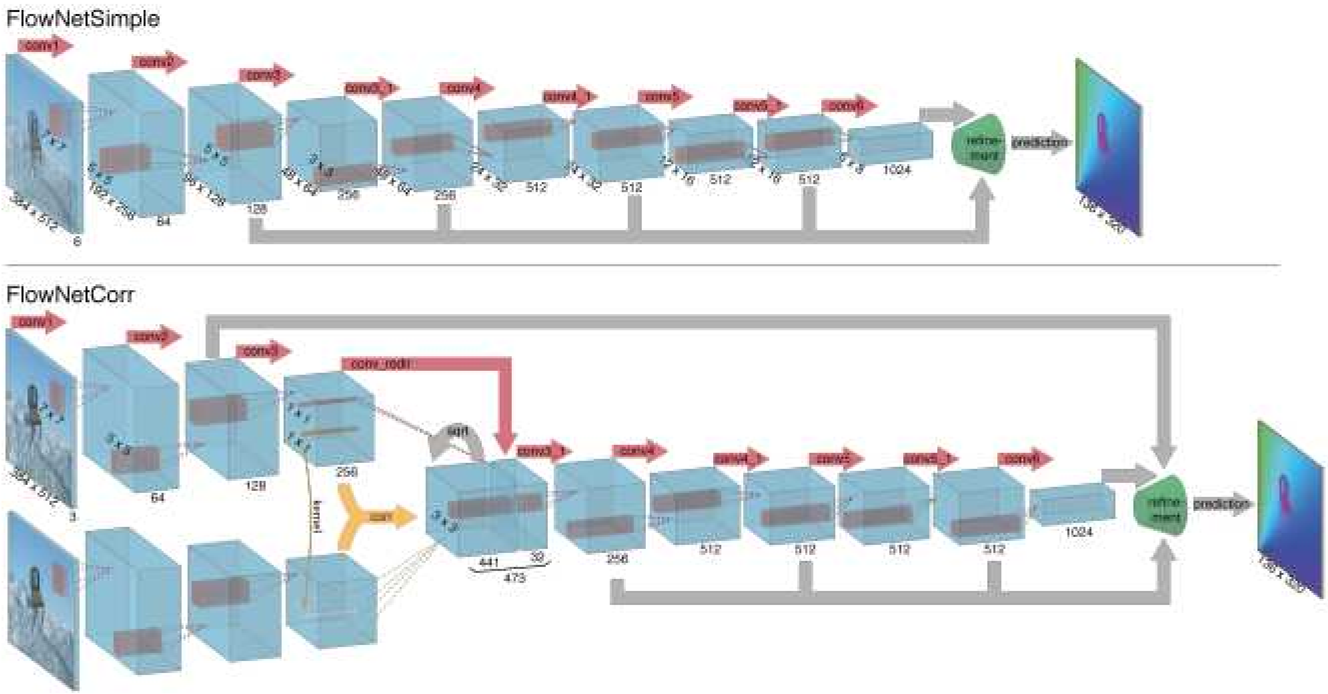
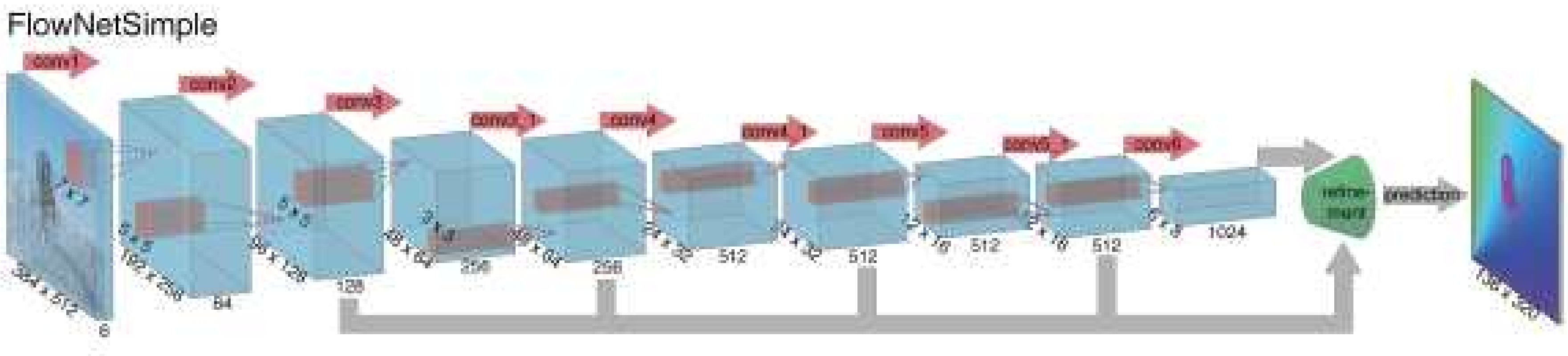


Figure 2. The two network architectures: FlowNetSimple (top) and FlowNetCorr (bottom).

FlowNetSimple

- Stack both input images together and feed them to network
- Allows the network to decide itself how to process the image pair to extract the motion information
- <u>Never be sure</u> that a local gradient optimization like stochastic gradient descent can get the network to this point



▲The architecture of FlowNetSimple

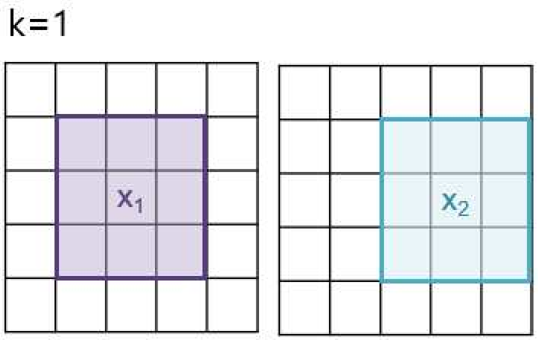
FlowNetCorr

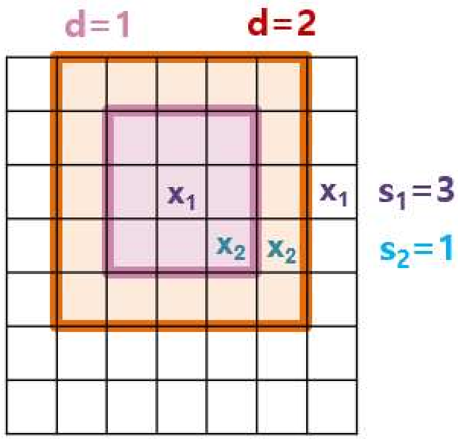
- Produce meaningful representations of the two images separately and then combine them on a higher level
- <u>Correlation layer</u> performs multiplicative patch comparisons between two feature maps ➔ <u>No trainable weights</u>

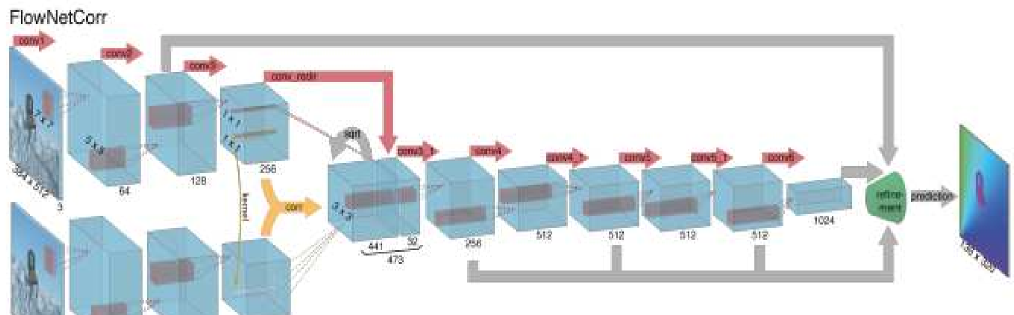
$$c(\mathbf{x}_1,\mathbf{x}_2)=\sum_{\mathbf{o}\in[-k,k]\times[-k,k]}\langle \mathbf{f}_1(\mathbf{x}_1+\mathbf{o}),\mathbf{f}_2(\mathbf{x}_2+\mathbf{o})\rangle \quad (1) \qquad K := 2k+1$$

- Limit the maximum displacement(d) for comparisons
- Use strides s_1 and s_2, to quantize x_1 globally and to quantize x_2 within the neighborhood centered around x_1

▼ Example of k, d, s







▲The architecture of FlowNetCorr

Refinement

- To provide dense per-pixel predictions
  - 1. Apply the ‘upconvolution’ to feature maps
  - 2. Concatenate it with corresponding feature maps
  - 3. Concatenate it with upsampled coarser flow prediction (if available)

➔Repeat this 4 times only (Still 4 times smaller than input)

  - 4. Apply bilinear upsampling or some variational refinement

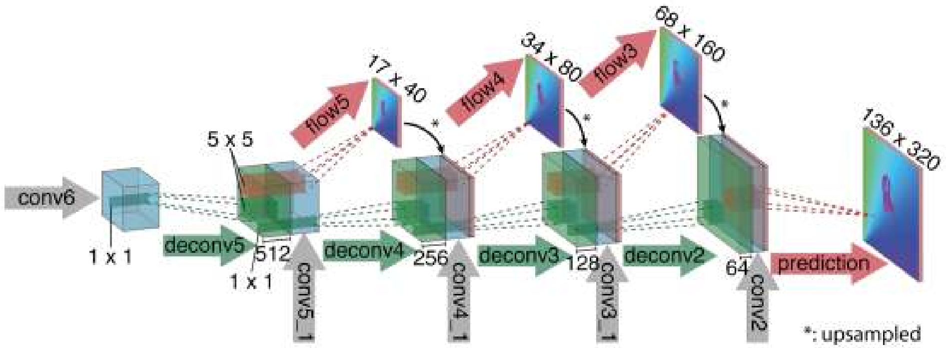


Figure 3. Refinement of the coarse feature maps to the high resolution prediction.

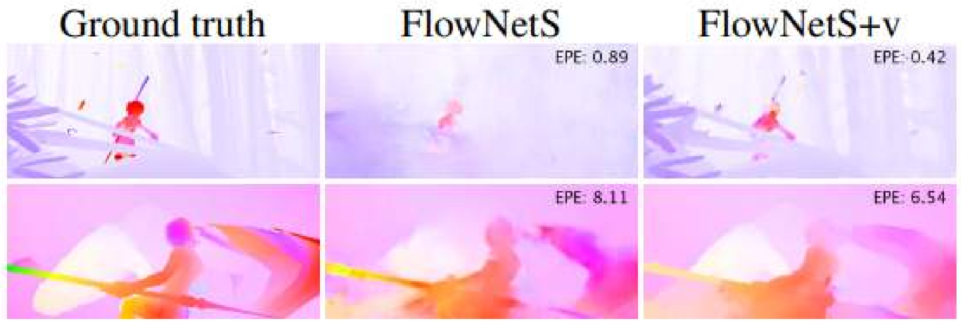


Figure 4. The effect of variational refinement. In case of small motions (first row) the predicted flow is changed dramatically. For larger motions (second row), big errors are not corrected, but the flow field is smoothed, resulting in lower EPE.

Dataset

- Middlebury : Displacements are very small, typically below 10 pixels
- KITTI : Containing only a very special motion type, motion of distant objects cannot be captured cause of recording scenes with a camera and a 3D laser scanner
- MPI Sintel : containing motion blur and atmospheric effects(Final) vs no these effects(Clean)
- Flying Chairs(created) : adding images of multiple chairs to the background and applying affine transform

	Frame pairs	Frames with ground truth	Ground truth density per frame
Middlebury	72	8	100%
KITTI	194	194	~50%
Sintel	1,041	1,041	100%
Flying Chairs	22,872	22,872	100%

Table 1. Size of already available datasets and the proposed Flying Chairs dataset.

Figure 5. Two examples from the Flying Chairs dataset. Generated image pair and color coded flow field (first three columns), augmented image pair and corresponding color coded flow field respectively (last three columns).

Results

Method	Sintel Clean		Sintel Final		KITTI		Middlebury train		Middlebury test		Chairs	Time (sec)	
	train	test	train	test	train	test	AEE	AAE	AEE	AAE	test	CPU	GPU
EpicFlow [30]	2.40	4.12	3.70	6.29	3.47	3.8	0.31	3.24	0.39	3.55	2.94	16	-
DeepFlow [35]	3.31	5.38	4.56	7.21	4.58	5.8	0.21	3.04	0.42	4.22	3.53	17	-
EPPM [3]	-	6.49	-	8.38	-	9.2	-	-	0.33	3.36	-	-	0.2
LDOF [6]	4.29	7.56	6.42	9.12	13.73	12.4	0.45	4.97	0.56	4.55	3.47	65	2.5
FlowNetS	4.50	7.42	5.45	8.43	8.26	-	1.09	13.28	-	-	2.71	-	0.08
FlowNetS+v	3.66	6.45	4.76	7.67	6.50	-	0.33	3.87	-	-	2.86	-	1.05
FlowNetS+ft	(3.66)	6.96	(4.44)	7.76	7.52	9.1	0.98	15.20	-	-	3.04	-	0.08
FlowNetS+ft+v	(2.97)	6.16	(4.07)	7.22	6.07	7.6	0.32	3.84	0.47	4.58	3.03	-	1.05
FlowNetC	4.31	7.28	5.87	8.81	9.35	-	1.15	15.64	-	-	2.19	-	0.15
FlowNetC+v	3.57	6.27	5.25	8.01	7.45	-	0.34	3.92	-	-	2.61	-	1.12
FlowNetC+ft	(3.78)	6.85	(5.28)	8.51	8.79	-	0.93	12.33	-	-	2.27	-	0.15
FlowNetC+ft+v	(3.20)	6.08	(4.83)	7.88	7.31	-	0.33	3.81	0.50	4.52	2.67	-	1.12

Table 2. Average endpoint errors (in pixels) of our networks compared to several well-performing methods on different datasets. Th numbers in parentheses are the results of the networks on data they were trained on, and hence are not directly comparable to other result

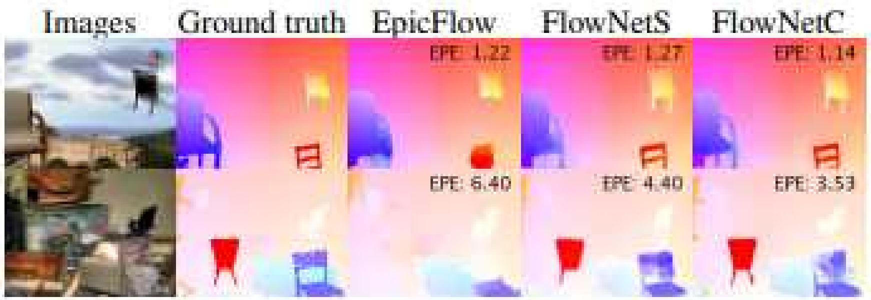


Figure 6. Examples of optical flow prediction on the Flying Chairs dataset. The images include fine details and small objects with large displacements which EpicFlow often fails to find. The networks are much more successful.

Figure 7. Examples of optical flow prediction on the Sintel dataset. In each row left to right: overlaid image pair, ground truth flow and 3 predictions: EpicFlow, FlowNetS and FlowNetC. Endpoint error is shown for every frame. Note that even though the EPE of FlowNets is usually worse than that of EpicFlow, the networks often better preserve fine details.

The end