



Computer Vision; Image Transformation; Optical Flow and Depth Estimation

Maziar Raissi

Assistant Professor

Department of Applied Mathematics

University of Colorado Boulder

maziar.raissi@colorado.edu



Boulder

FlowNet: Learning Optical Flow with Convolutional Networks

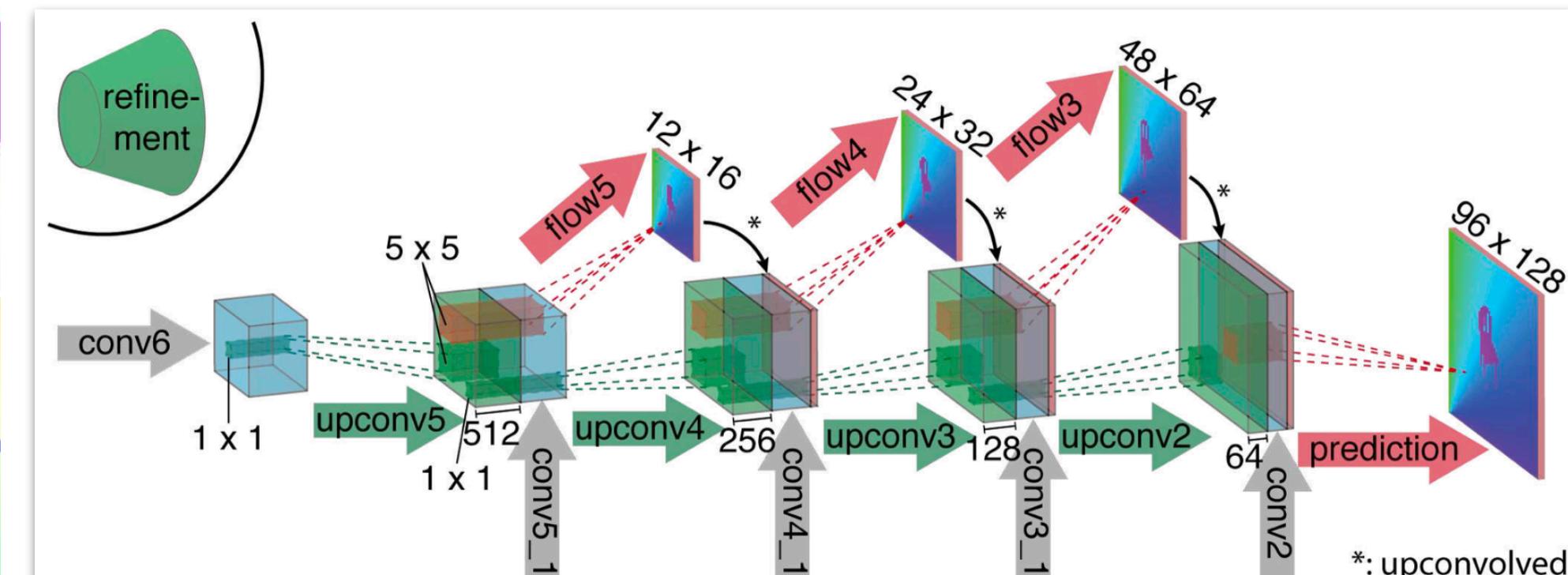
Given a dataset consisting of image pairs and ground truth flows, train a network to predict the $x-y$ flow fields directly from the images.

Flying Chairs Dataset

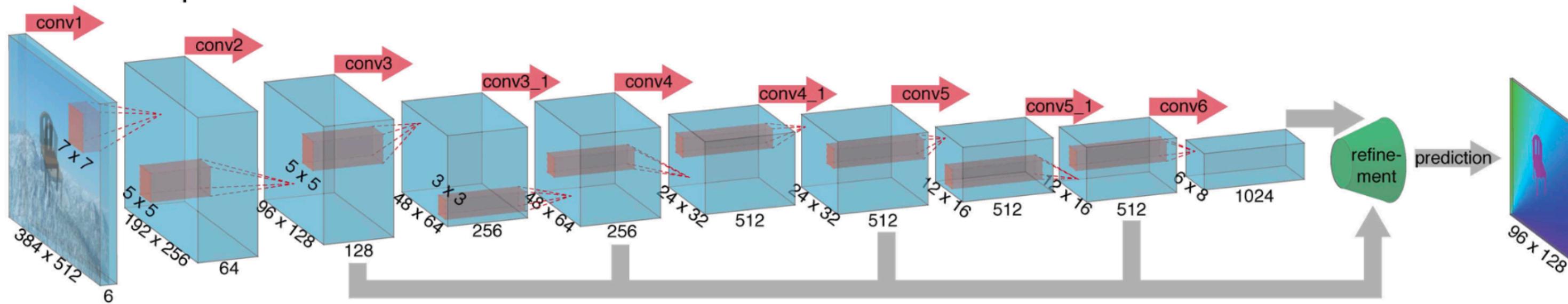


Data Augmentation

upconv: unpooling + conv

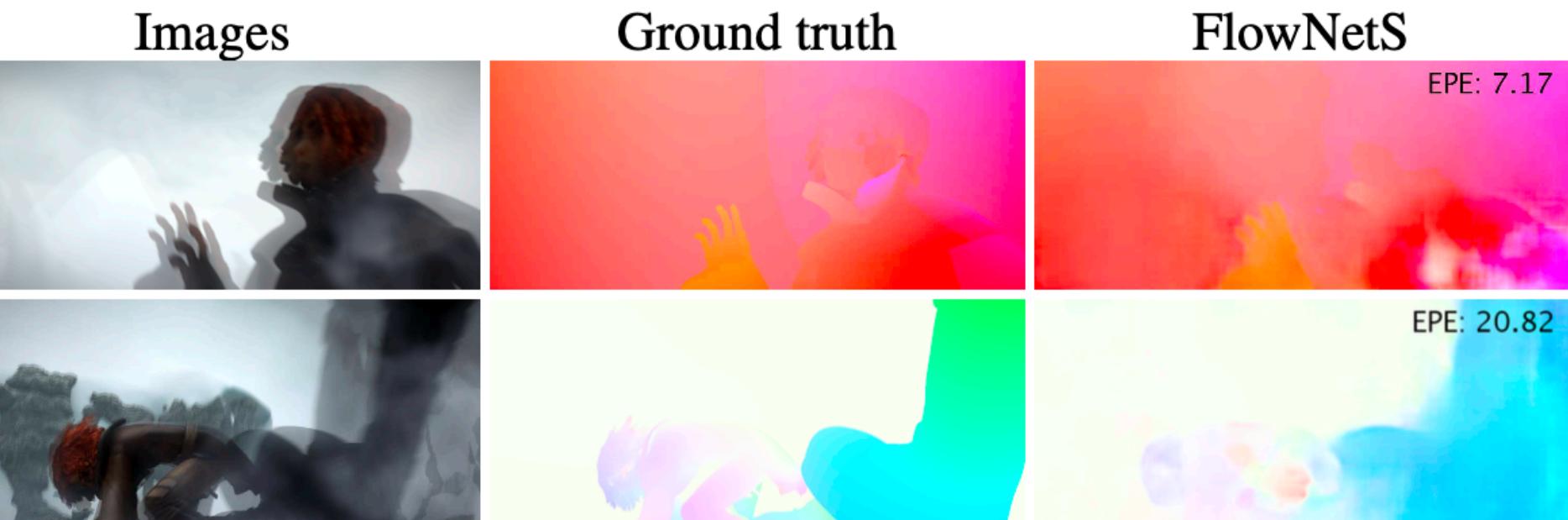


FlowNetSimple



Method	Sintel Clean		Sintel Final		KITTI		Middlebury train		Middlebury test		Chairs test	Time (sec)	
	train	test	train	test	train	test	AEE	AAE	AEE	AAE		CPU	GPU
EpicFlow [30]	2.27	4.12	3.57	6.29	3.47	3.8	0.31	3.24	0.39	3.55	2.94	16	-
DeepFlow [35]	3.19	5.38	4.40	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]	4.19	7.56	6.28	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

	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%



End Point Error (EPE) Loss: Euclidean distance between the predicted flow vector and the ground truth, averaged over all pixels



Boulder

FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks



- problems with small displacements
- noisy artifacts in estimated flow fields

Dataset Schedules

Numbers indicate endpoint errors on Sintel train clean

Architecture	Datasets	S_{short}	S_{long}	S_{fine}
FlowNetS	Chairs	4.45	-	-
	Chairs	-	4.24	4.21
	Things3D	-	5.07	4.50
	mixed	-	4.52	4.10
	Chairs → Things3D	-	4.24	3.79
FlowNetC	Chairs	3.77	-	-
	Chairs → Things3D	-	3.58	3.04

Training on Chairs first and fine-tuning on Things3D yields the best results. FlowNetC performs better than FlowNetS.

Stacking Networks

$I_1, I_2 \rightarrow$ images $w = (u, v)^T \rightarrow$ previous flow estimation

$\tilde{I}_2(x, y) = I_2(x + u, y + v) \rightarrow$ warp the second image I_2 (bilinear interpolation)

$e = \|\tilde{I}_2(x, y) - I_1\| \rightarrow$ error (\tilde{I}_2 should be close to I_1)

FlowNetCorr

$$f_1, f_2 \in \mathbb{R}^{w \times h \times c} \text{ or } f_1, f_2 : \mathbb{R}^2 \rightarrow \mathbb{R}^c$$

$c(x_1, x_2) \rightarrow$ correlation of two patches centered at x_1 in the first map and x_2 in the second map

$$c(x_1, x_2) = \sum_{o \in [-k, k] \times [-k, k]} \langle f_1(x_1 + o), f_2(x_2 + o) \rangle \rightarrow \text{for a square patch of size } K = 2k + 1$$

$$w^2 h^2 c K^2 \rightarrow \text{computational cost}$$

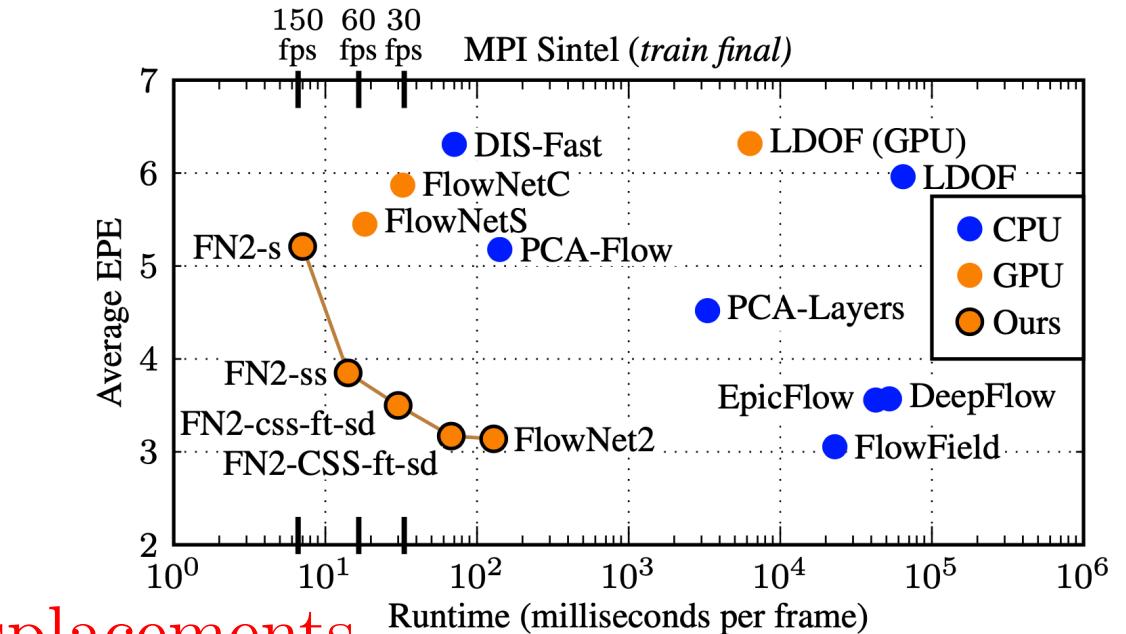
compute $c(x_1, x_2)$ only in a neighborhood of x_1 of size $D = 2d + 1$

$$w h D^2 c K^2 \rightarrow \text{computational cost}$$

$$w \times h \times D^2 \rightarrow \text{output size}$$

use stride s_1 to quantize x_1 globally

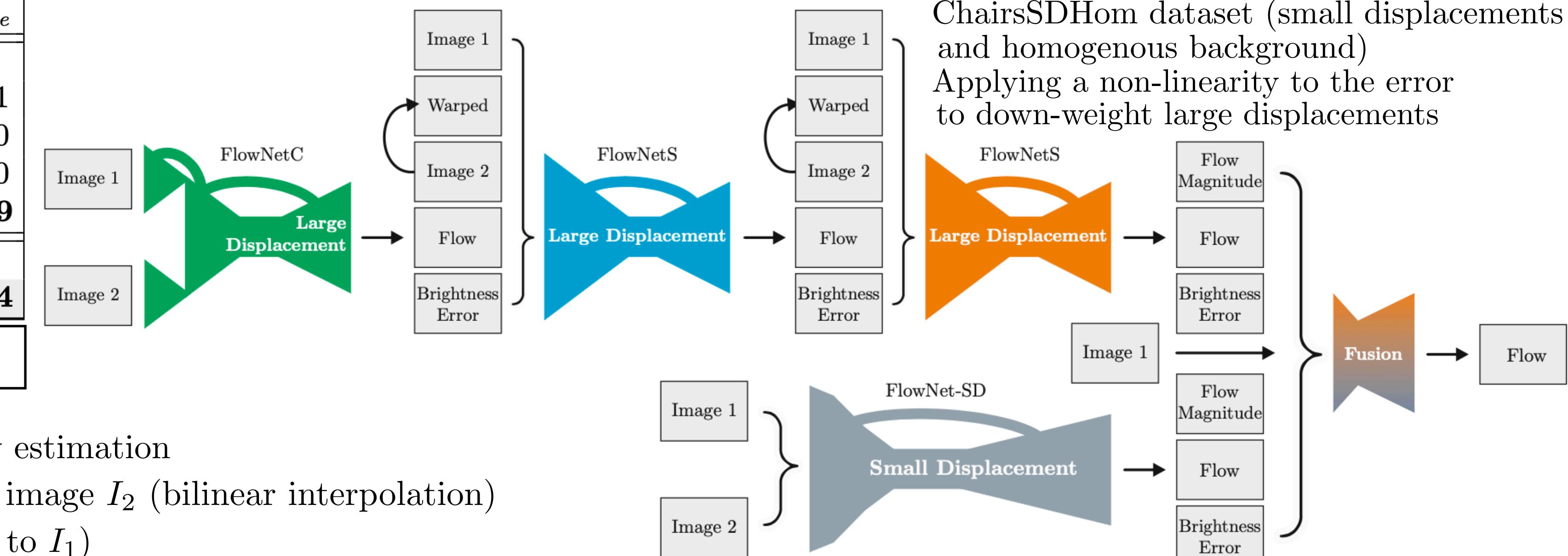
use stride s_2 to quantize x_2 locally around x_1



Small Displacements

ChairsSDHom dataset (small displacements and homogenous background)

Applying a non-linearity to the error to down-weight large displacements



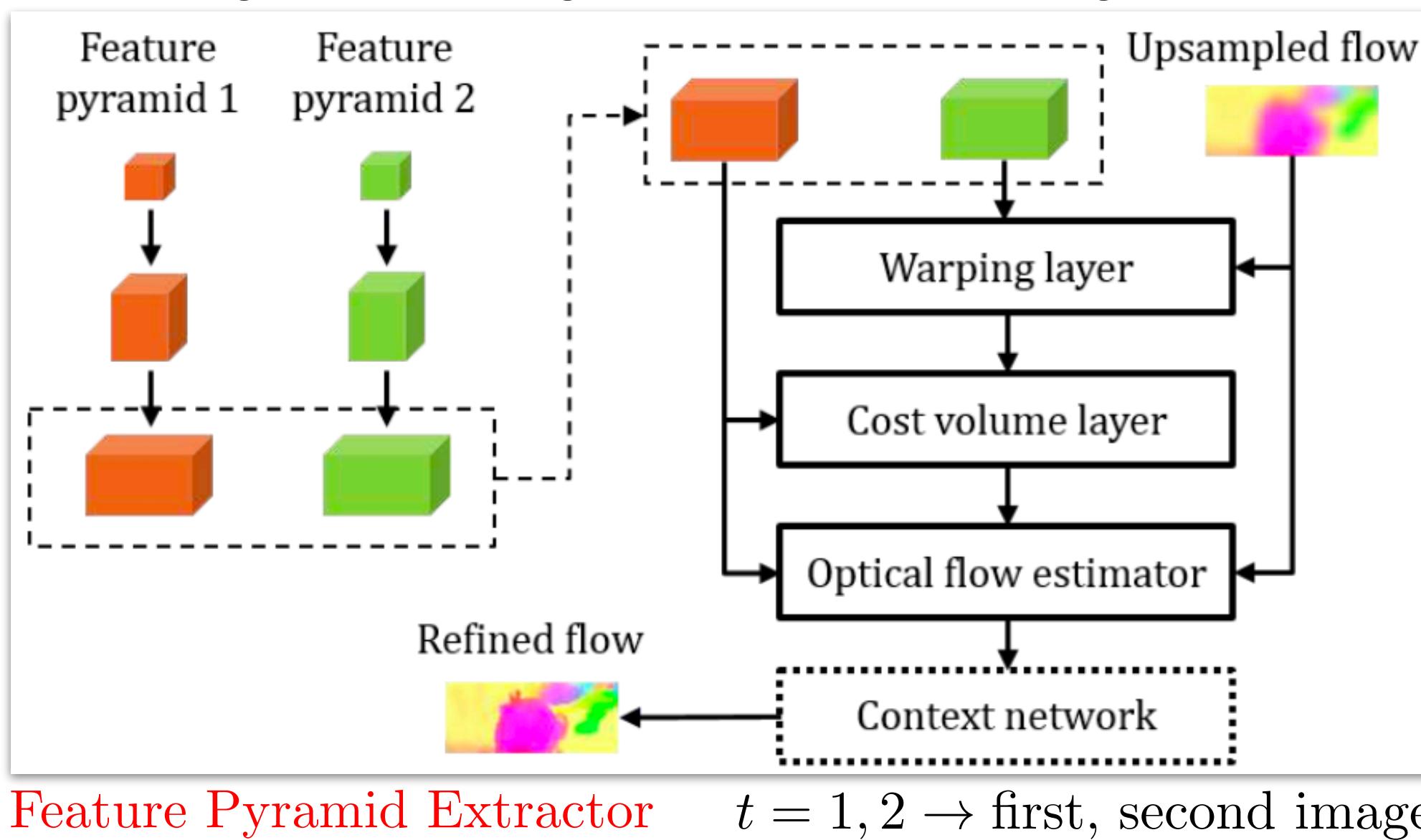


Boulder

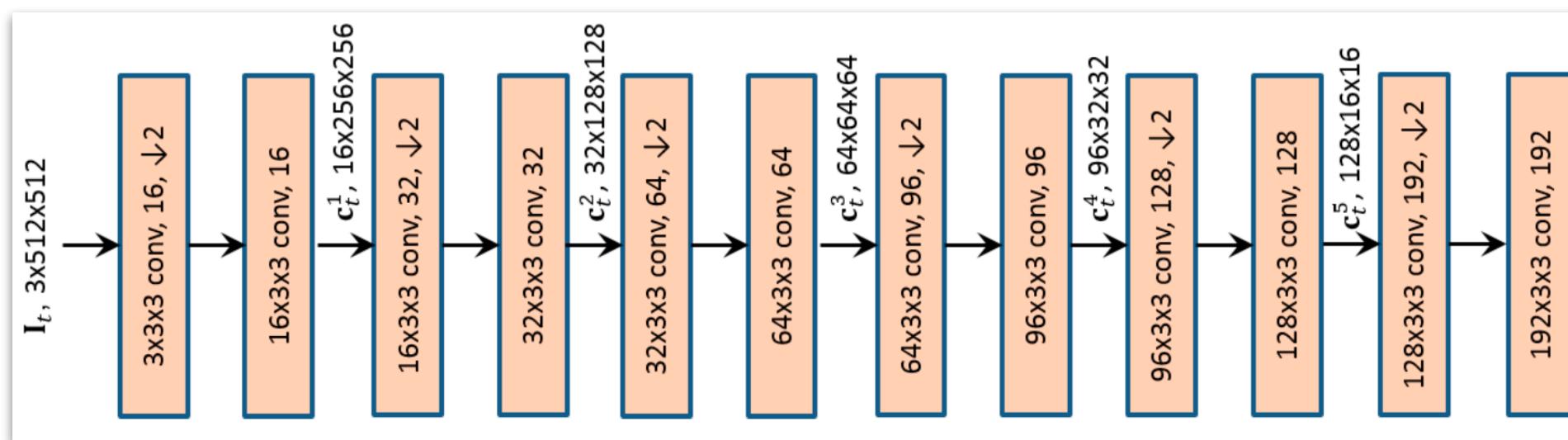
PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume

- action recognition – autonomous driving – video editing
- FlyingChairs dataset (training)
- FlyingThings3D dataset (fine-tuning)
- Sintel (fine-tuning)
- KITTI (fine-tuning)

Combining deep learning with domain knowledge!



Feature Pyramid Extractor



$t = 1, 2 \rightarrow$ first, second image

Warping Layer

Warp features of the second image toward the first image using the $\times 2$ upsampled flow from the $(l + 1)$ -th level

$$\mathbf{c}_w^l(\mathbf{x}) = \mathbf{c}_2^l(\mathbf{x} + \text{up}_2(\mathbf{w}^{l+1})(\mathbf{x})) \rightarrow \begin{matrix} \text{bi-linear interpolation} \\ \text{pixel index} \end{matrix} \cup \text{flow}$$

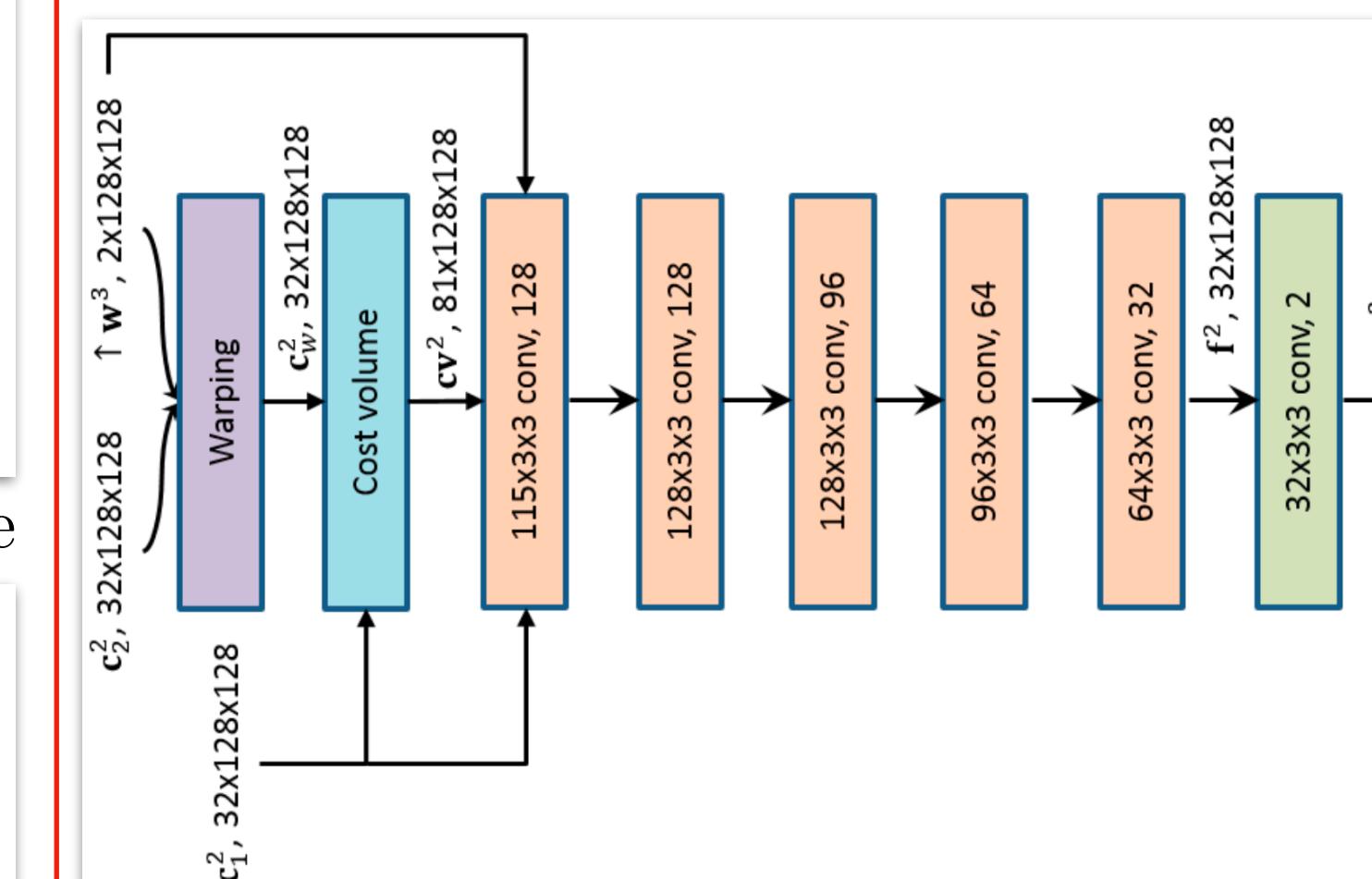
Cost Volume Layer

$\mathbf{cv}^l(\mathbf{x}_1, \mathbf{x}_2) = \frac{1}{N} (\mathbf{c}_1^l(\mathbf{x}_1))^T \mathbf{c}_w^l(\mathbf{x}_2) \rightarrow$ correlation between features of the first image and warped features of the second image

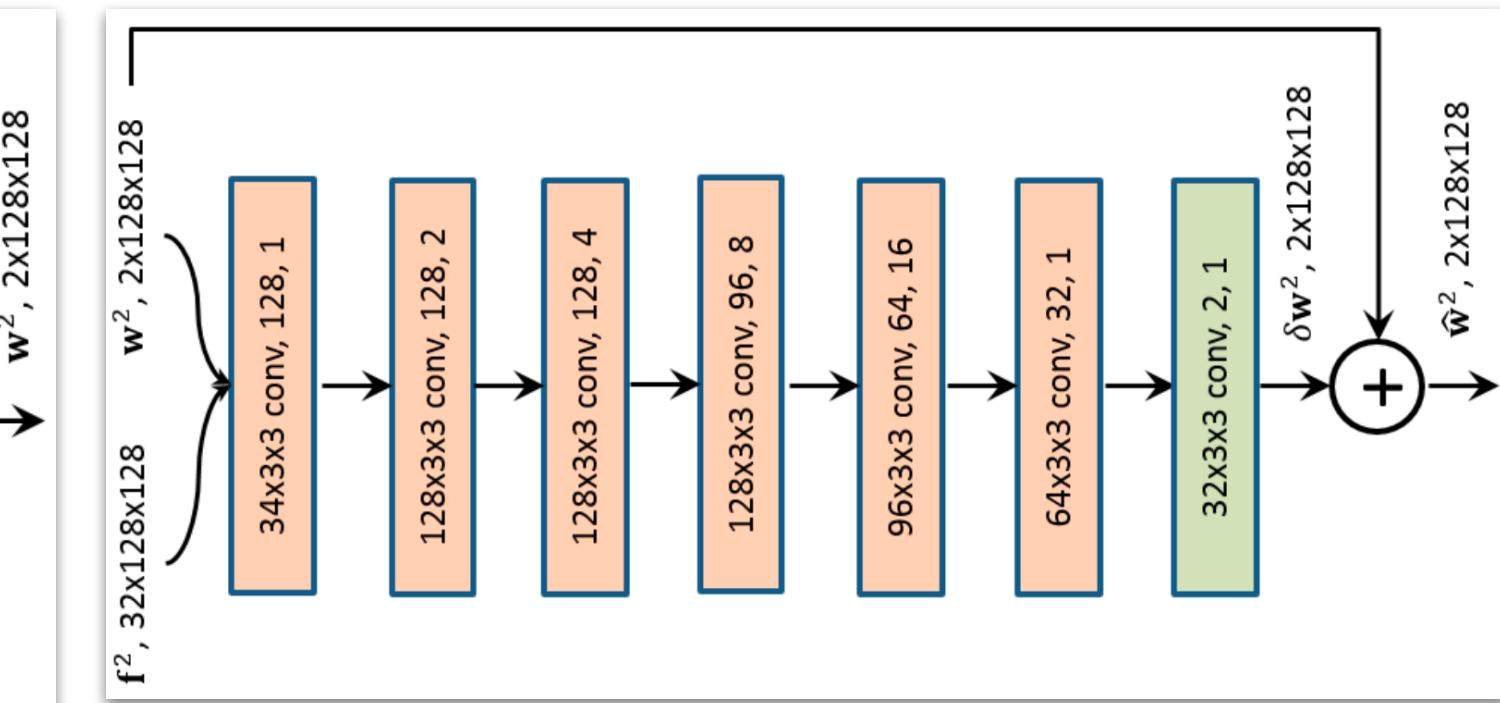
$$|\mathbf{x}_1 - \mathbf{x}_2|_\infty \leq d \rightarrow \text{limited range of } d \text{ pixels}$$

$$d^2 \times H^l \times W^l \rightarrow \text{dimension of the 3D cost volume}$$

Optical Flow Estimator



Context Network

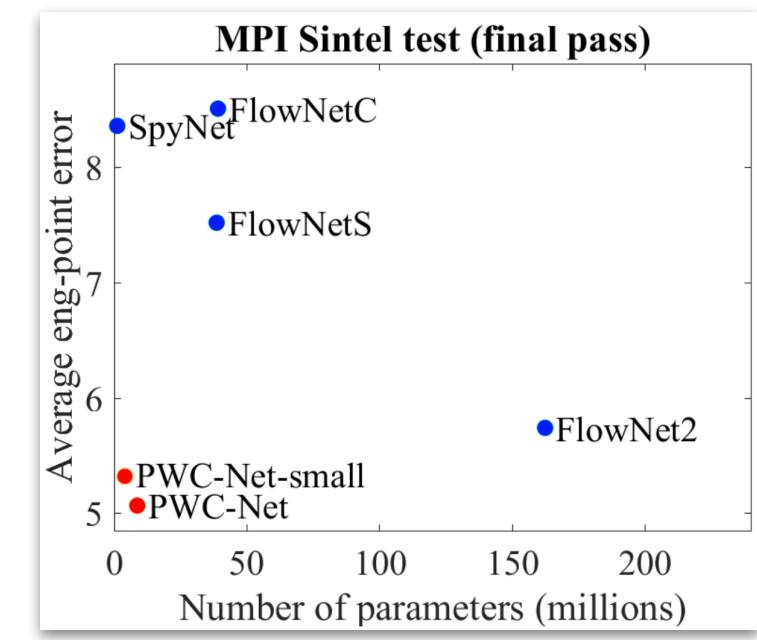


Training Loss

$$\mathcal{L}(\Theta) = \sum_{l=l_0}^L \alpha_l \sum_{\mathbf{x}} (|\mathbf{w}_{\Theta}^l(\mathbf{x}) - \mathbf{w}_{\text{GT}}^l(\mathbf{x})|_2 + \epsilon)^q + \gamma |\Theta|_2$$

Fine-tuning Loss

Sun, Deqing, et al. "Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.



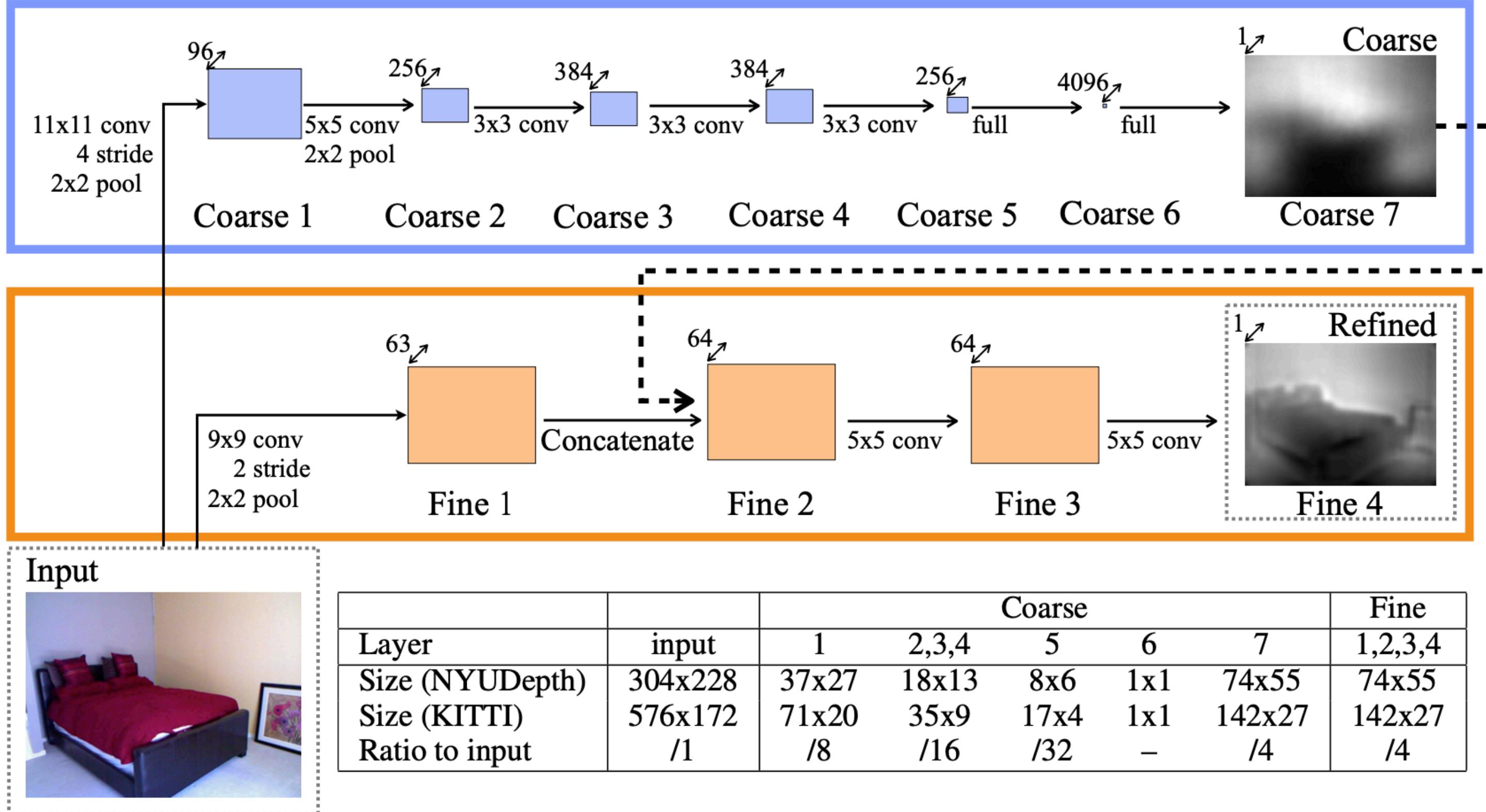


Boulder

Depth Map Prediction from a Single Image using a Multi-Scale Deep Network

3D geometry of a scene!

- NYU Depth Dataset (Microsoft Kinect Camera)
- KITTI Dataset (Depth Sensors)



Global Coarse-Scale Network

Pre-trained on ImageNet

Global understanding of the full scene

Local Fine-Scale Network

Train the coarse network first, then train the fine network

Eigen, David, Christian Puhrsich, and Rob Fergus. "Depth map prediction from a single image using a multi-scale deep network." *arXiv preprint arXiv:1406.2283* (2014).

Scale-Invariant Error

The global scale of a scene is a fundamental ambiguity in depth prediction.

$y \rightarrow$ predicted depth map

$y^* \rightarrow$ ground-truth depth map

$n \rightarrow$ number of pixels indexed with i

$$D(y, y^*) = \frac{1}{n} \sum_{i=1}^n (\log y_i - \log y_i^* + \alpha(y, y^*))^2$$

scale-invariant mean squared error (in log space)

$$\alpha(y, y^*) = \frac{1}{n} \sum_i (\log y_i^* - \log y_i)$$

the value of α that minimizes the error for a given (y, y^*)
All scalar multiples of y have the same error!

Training Loss

$$L(y, y^*) = \frac{1}{n} \sum_i d_i^2 - \frac{\lambda}{n^2} \left(\sum_i d_i \right)^2$$

$$d_i = \log y_i - \log y_i^*$$

If $\lambda = 1$ then $L(y, y^*) = D(y, y^*)$.

Evaluation Metrics

Threshold: % of y_i s.t. $\max(\frac{y_i}{y_i^*}, \frac{y_i^*}{y_i}) = \delta < \text{thr}$

Abs Relative difference: $\frac{1}{|T|} \sum_{y \in T} |y - y^*| / y^*$

Squared Relative difference: $\frac{1}{|T|} \sum_{y \in T} ||y - y^*||^2 / y^*$

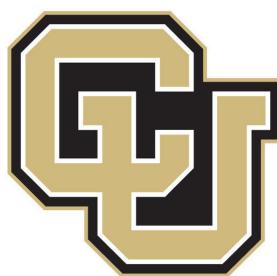
RMSE (linear): $\sqrt{\frac{1}{|T|} \sum_{y \in T} ||y - y^*||^2}$

RMSE (log): $\sqrt{\frac{1}{|T|} \sum_{y \in T} ||\log y_i - \log y_i^*||^2}$

RMSE (log, scale-invariant): The error Eqn. 1

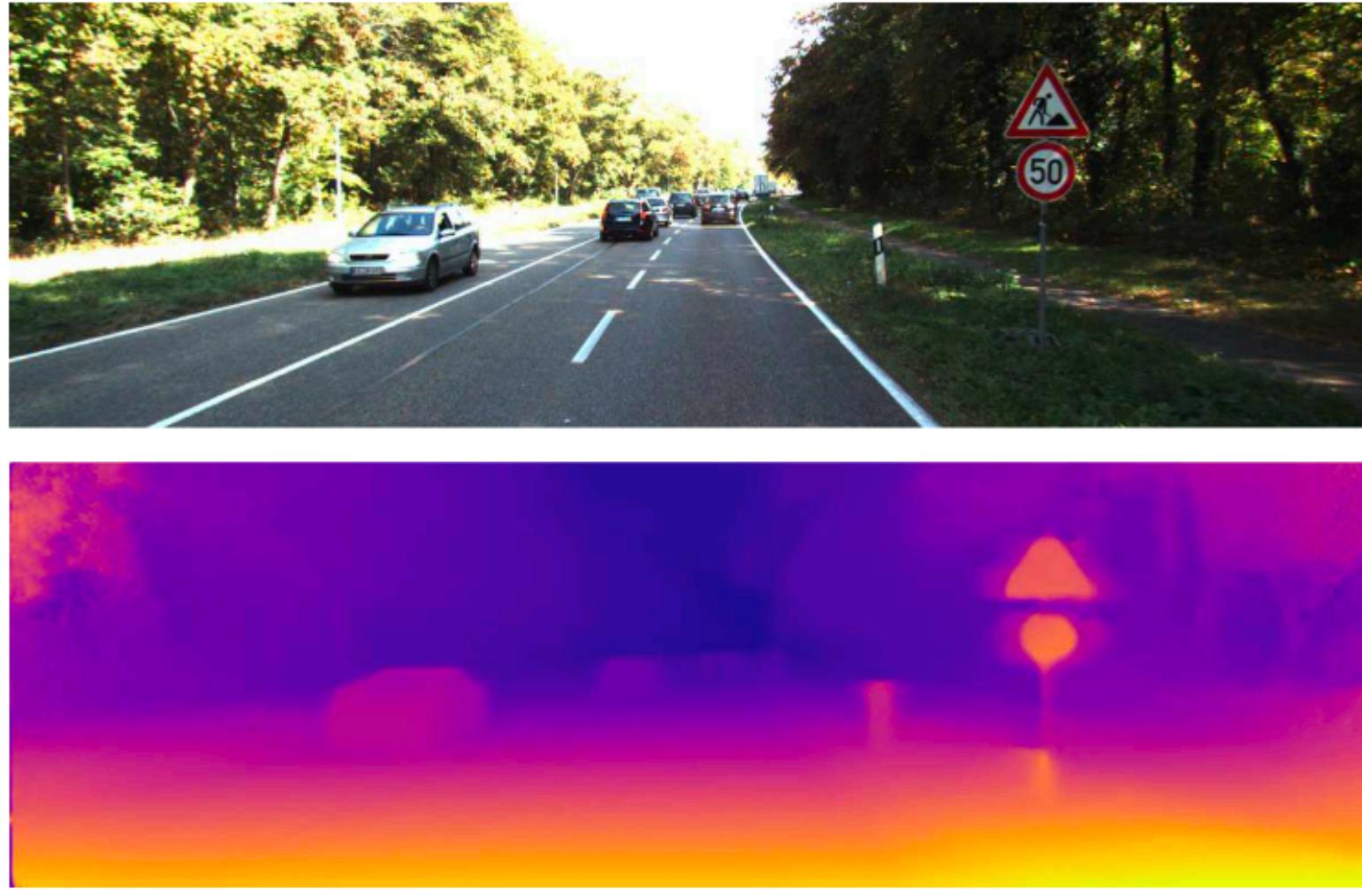
Data Augmentation

- **Scale:** Input and target images are scaled by $s \in [1, 1.5]$, and the depths are divided by s .
- **Rotation:** Input and target are rotated by $r \in [-5, 5]$ degrees.
- **Translation:** Input and target are randomly cropped to the sizes indicated in Fig. 1.
- **Color:** Input values are multiplied globally by a random RGB value $c \in [0.8, 1.2]^3$.
- **Flips:** Input and target are horizontally flipped with 0.5 probability.



Boulder

Unsupervised Monocular Depth Estimation with Left-Right Consistency



$I \rightarrow$ a single image at test time

$z = g(I) \rightarrow$ per-pixel depth prediction

$g \rightarrow$ to be learned

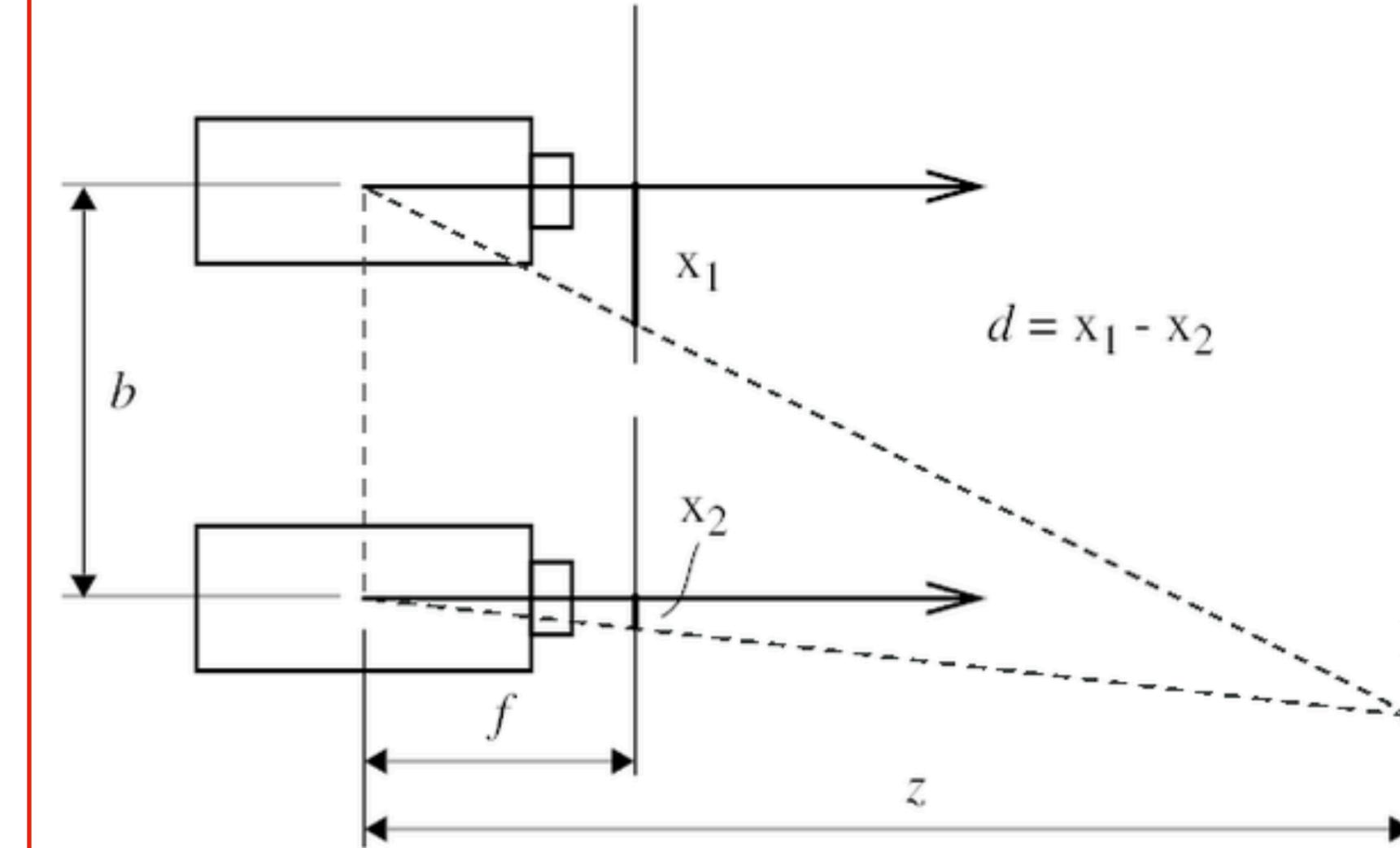
It is presently not practical to treat this as a supervised learning problem; Laser scanners are expensive and can be imprecise in natural scenes featuring movement and reflections.

Depth Estimation as Image Reconstruction

The intuition here is that, given a calibrated pair of binocular cameras, if we can learn a function that is able to reconstruct one image from the other, then we have learned something about the 3D shape of the scene that is being imaged.

Godard, Clément, Oisin Mac Aodha, and Gabriel J. Brostow. "Unsupervised monocular depth estimation with left-right consistency." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.

Training data (KITTI)
 $I^l, I^r \rightarrow$ left and right rectified stereo image pairs



$f \rightarrow$ camera focal length

$b \rightarrow$ baseline distance between the cameras

$z \rightarrow$ depth $\frac{d}{b} = \frac{f}{z} \rightarrow$ similar triangles

$d \rightarrow$ disparity $\frac{d}{b} = \frac{f}{z} \rightarrow$ similar triangles

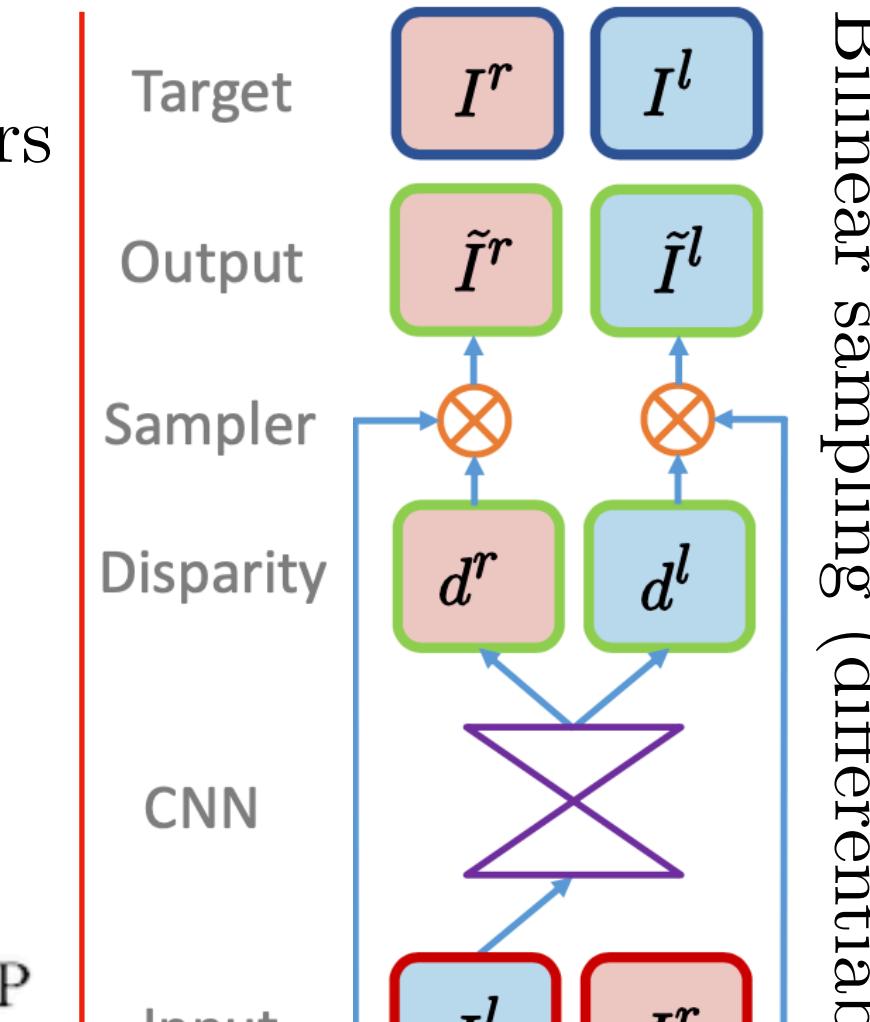
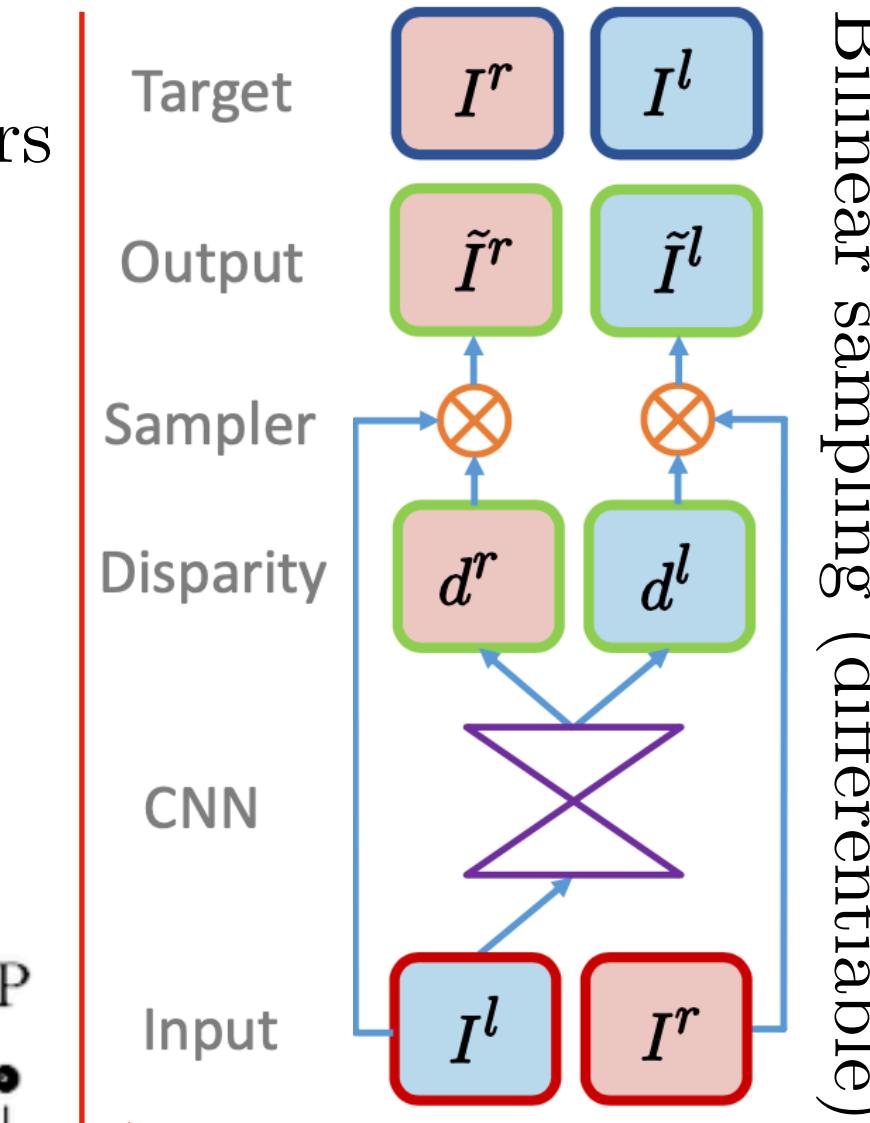
$z = \frac{bf}{d} \rightarrow$ recover depth from disparity

Training Loss

$$C = \sum_{s=1}^4 C_s \rightarrow \text{total loss}$$

$C_s \rightarrow$ loss at scale s

$$C_s = \alpha_{ap}(C_{ap}^l + C_{ap}^r) + \alpha_{ds}(C_{ds}^l + C_{ds}^r) + \alpha_{lr}(C_{lr}^l + C_{lr}^r)$$



Appearance Matching

$$C_{ap}^l = \frac{1}{N} \sum_{i,j} \alpha \frac{1 - \text{SSIM}(I_{ij}^l, \tilde{I}_{ij}^l)}{2} + (1 - \alpha) \|I_{ij}^l - \tilde{I}_{ij}^l\|$$

SSIM \rightarrow Structural Similarity

https://en.wikipedia.org/wiki/Structural_similarity#Algorithm

Disparity Smoothness

$$C_{ds}^l = \frac{1}{N} \sum_{i,j} |\partial_x d_{ij}^l| e^{-\|\partial_x I_{ij}^l\|} + |\partial_y d_{ij}^l| e^{-\|\partial_y I_{ij}^l\|}$$

edge-aware weights: depth discontinuities often occur at image gradients

Left-Right Disparity Consistency

$$C_{lr}^l = \frac{1}{N} \sum_{i,j} |d_{ij}^l - d_{ij+d_{ij}^l}^r|$$

Test Time
 Use d^l and discard d^r

Applications:
 synthetic object insertion in computer graphics, synthetic depth of field in computational photography, grasping in robotics, using depth as a cue in human body pose estimation, robot assisted surgery, automatic 2D to 3D conversion in film, and self-driving cars.



Boulder

Questions?
