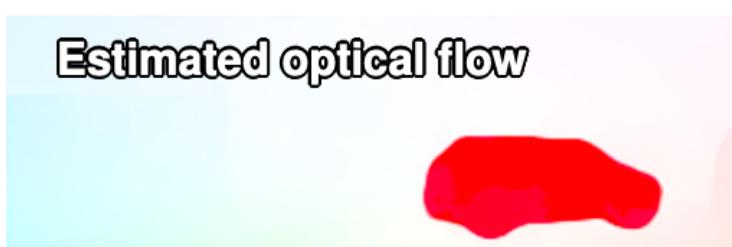
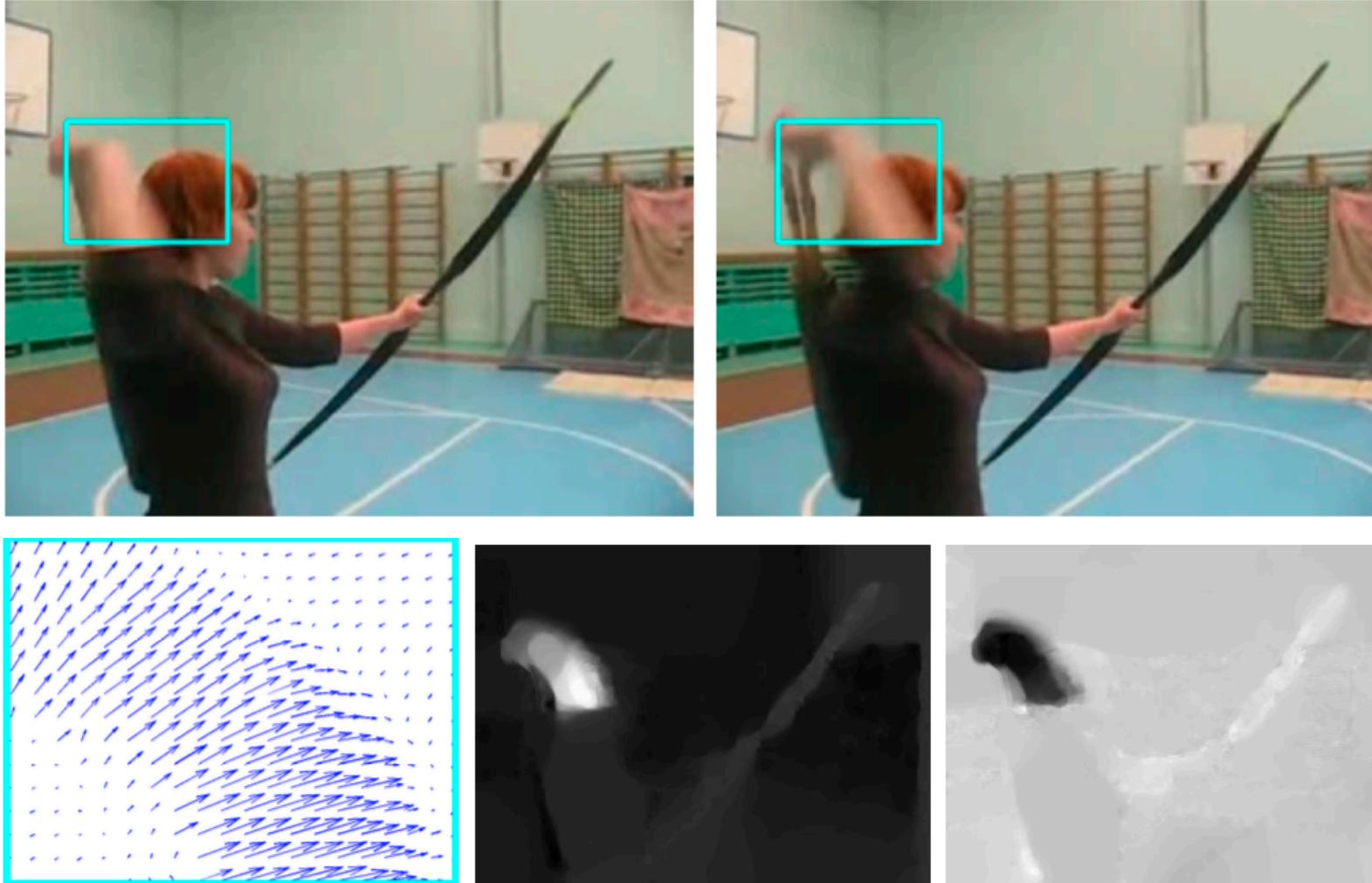


# Computer Vision; Image Transformation; Optical Flow and Depth Estimation



[YouTube Playlist](#)



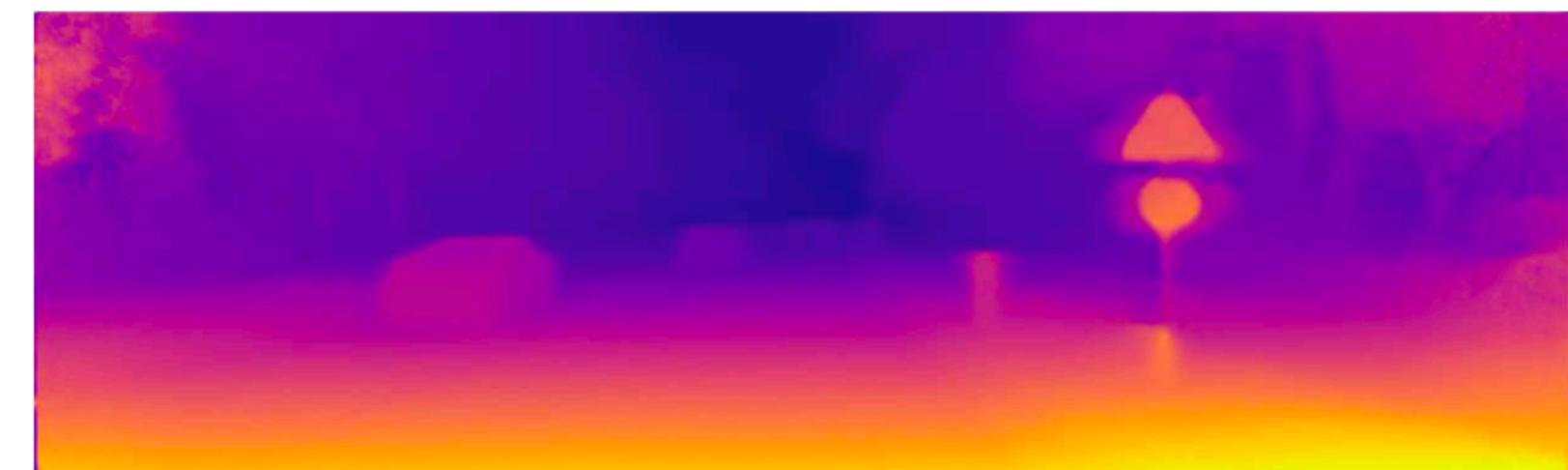
**Maziar Raissi**

**Assistant Professor**

Department of Applied Mathematics

University of Colorado Boulder

[maziar.raissi@colorado.edu](mailto:maziar.raissi@colorado.edu)





Boulder

# FlowNet: Learning Optical Flow with Convolutional Networks



[YouTube Playlist](#)

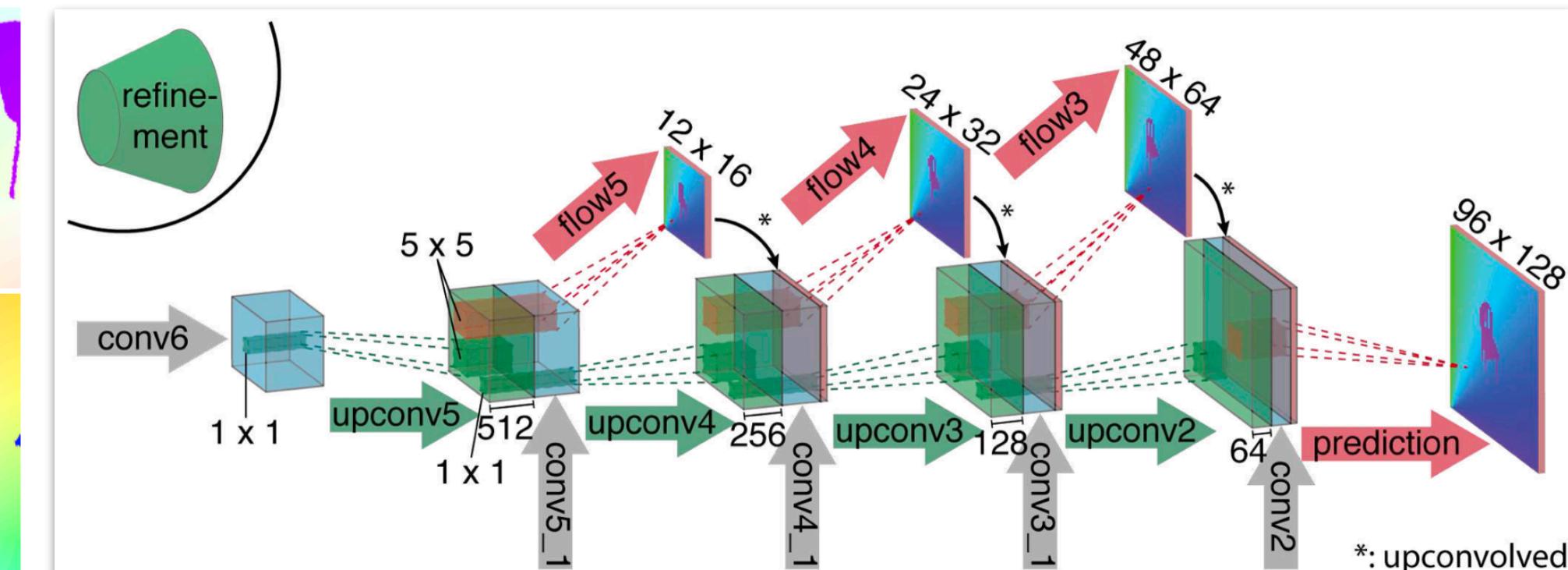
Given a dataset consisting of image pairs and ground truth flows, train a network to predict the  $x-y$  flow fields directly from the images.

Flying Chairs Dataset

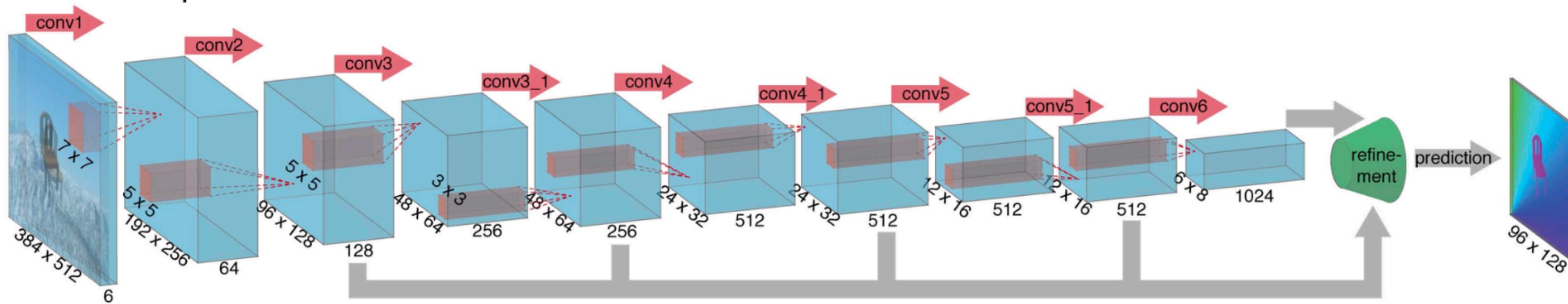


Data Augmentation

upconv: unpooling + conv

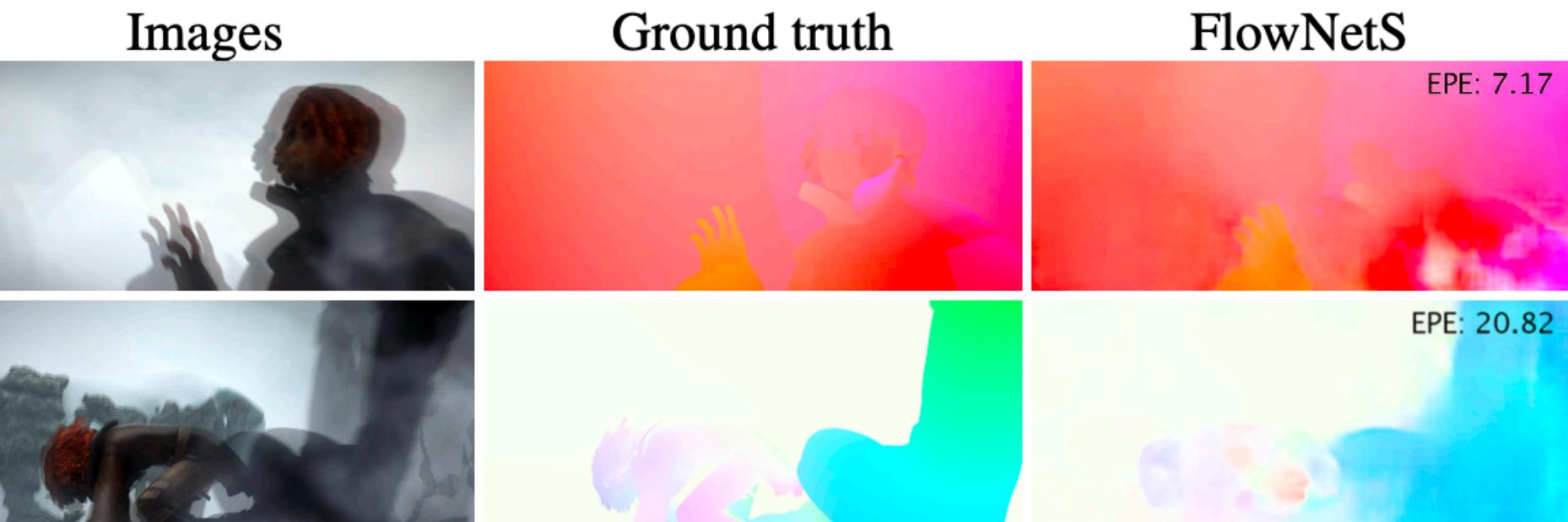


FlowNetSimple



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



[YouTube Playlist](#)



- 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	<b>3.79</b>
FlowNetC	Chairs	3.77	-	-
	Chairs → Things3D	-	3.58	<b>3.04</b>

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  
 $f_1, f_2 \in \mathbb{R}^{w \times h \times c}$  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

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

$w^2 h^2 c K^2 \rightarrow$  computational cost

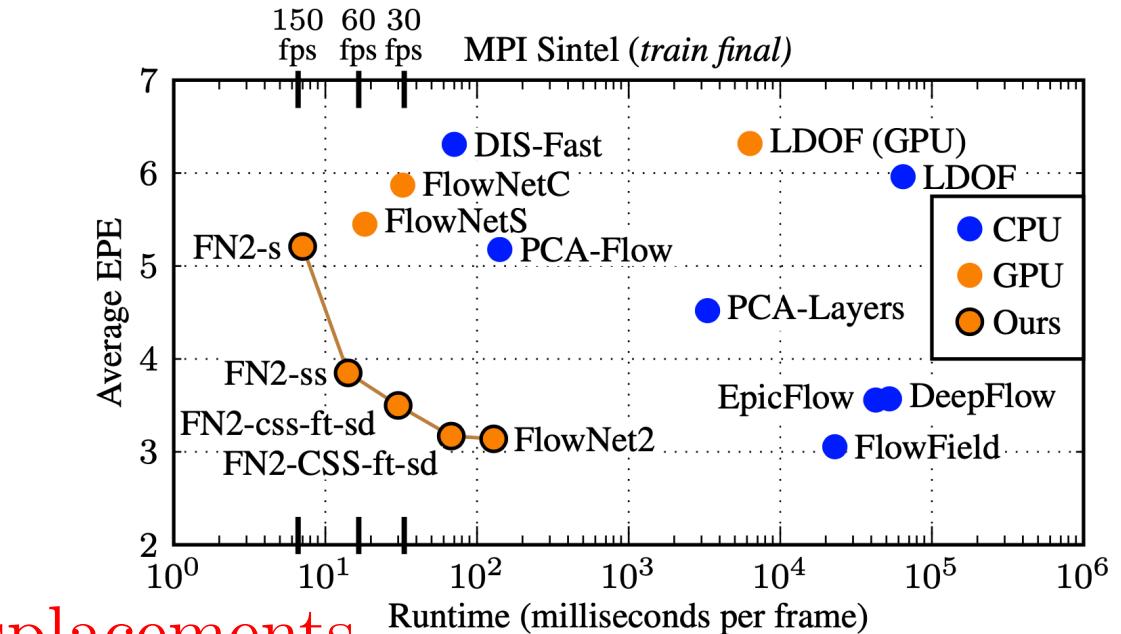
compute  $c(x_1, x_2)$  only in a neighborhood of  $x_1$  of size  $D = 2d + 1$

$wh D^2 c K^2 \rightarrow$  computational cost

$w \times h \times D^2 \rightarrow$  output size

use stride  $s_1$  to quantize  $x_1$  globally

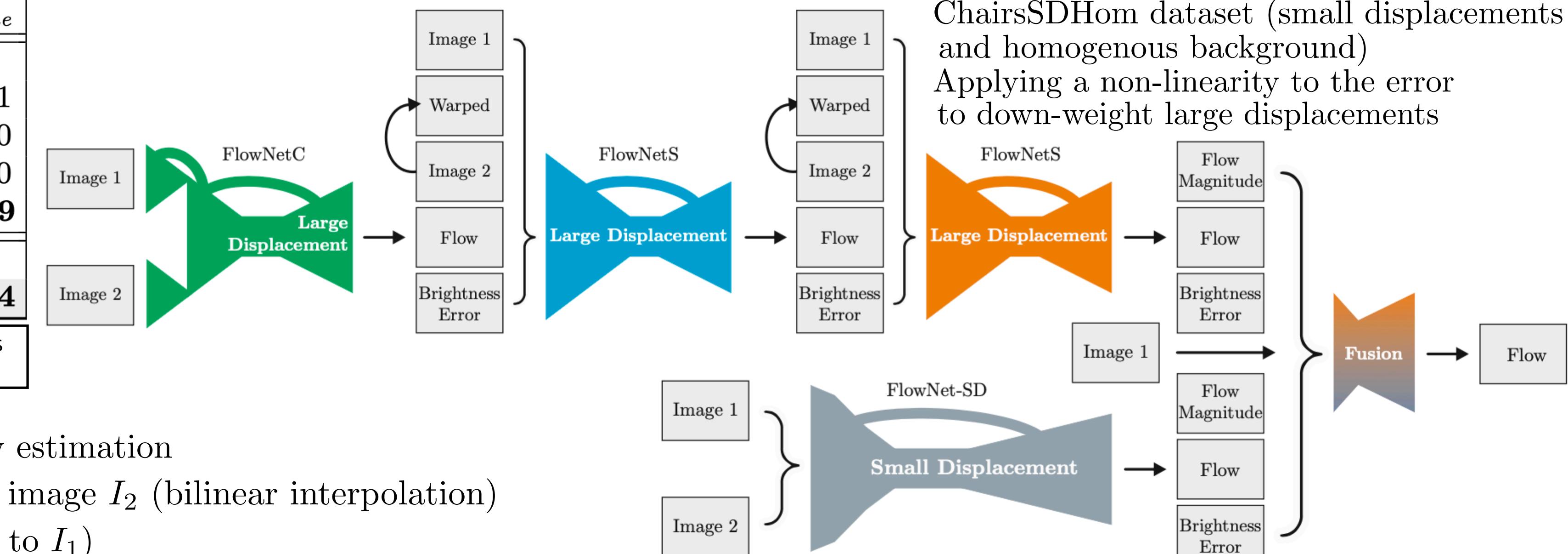
use stride  $s_2$  to quantize  $x_2$  locally around  $x_1$



## Small Displacements

ChairsSDHom dataset (small displacements and homogenous background)

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





# Boulder

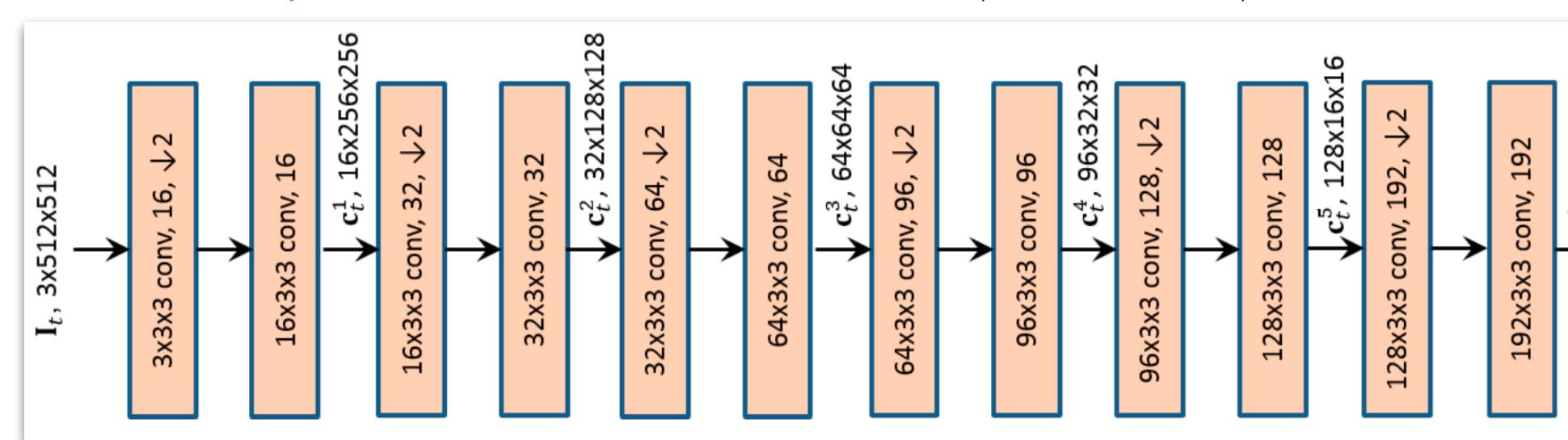
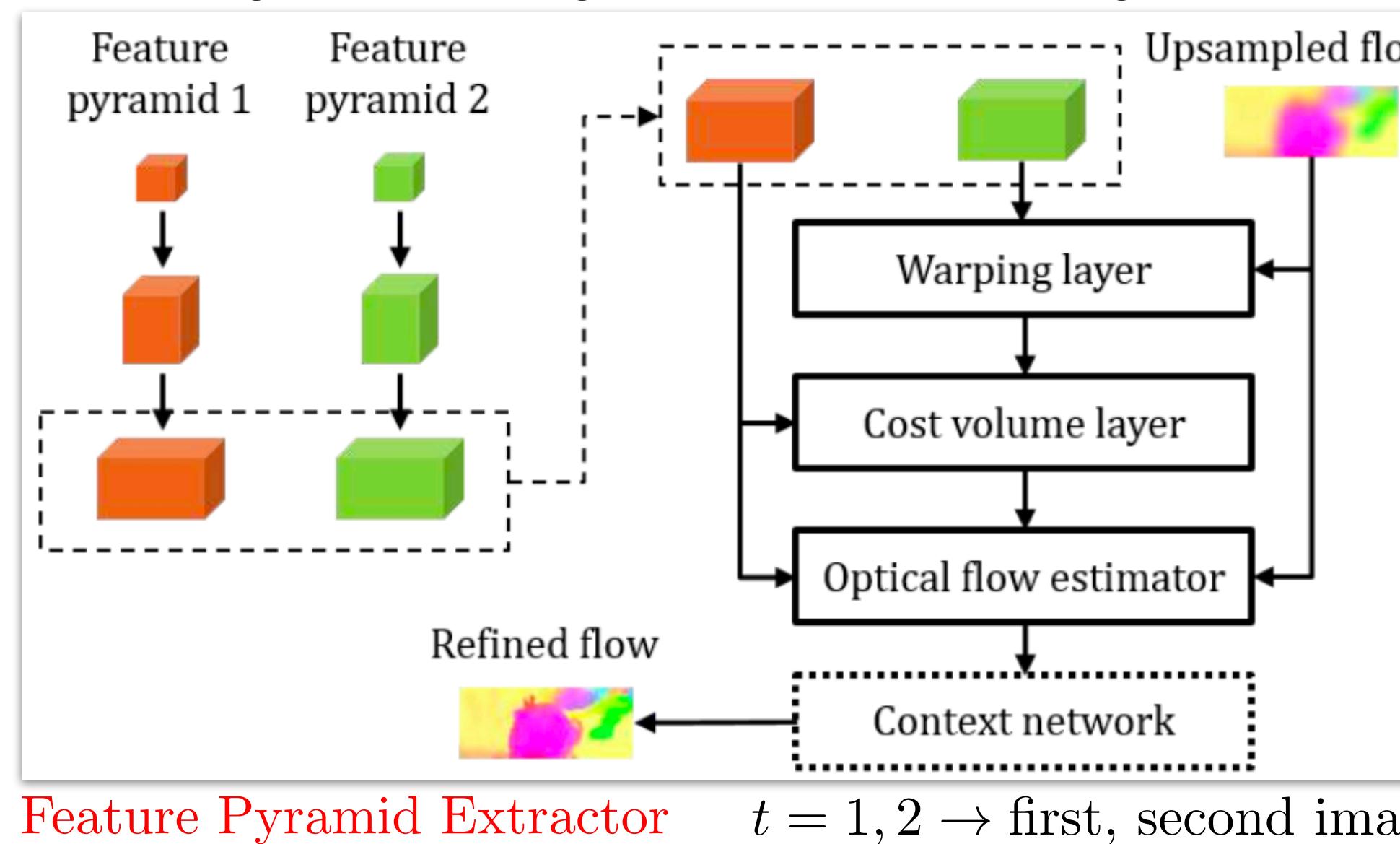
# PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume



# YouTube Video

- 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



## | Warping Layer

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

$$\mathbf{c}_w^l(\mathbf{x}) = \mathbf{c}_2^l\left(\mathbf{x} + \text{up}_2(\mathbf{w}^{l+1})(\mathbf{x})\right) \rightarrow \begin{array}{l} \text{bi-linear interpolation} \\ \text{pixel index} \curvearrowleft \qquad \qquad \curvearrowright \text{flow} \end{array}$$

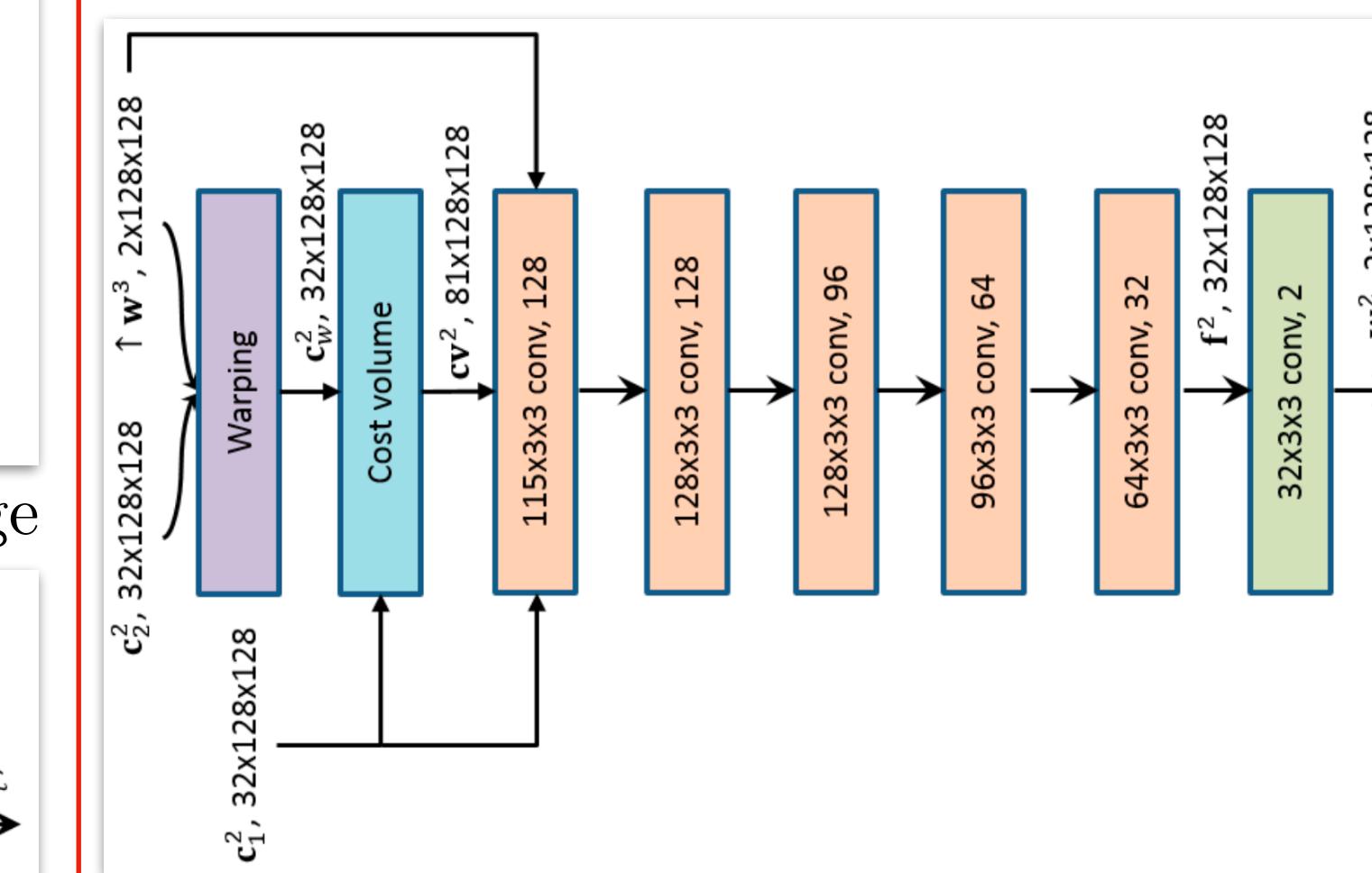
## Cost Volume Laye

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

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

$d^2 \times H^l \times W^l$  → dimension of the 3D cost volume

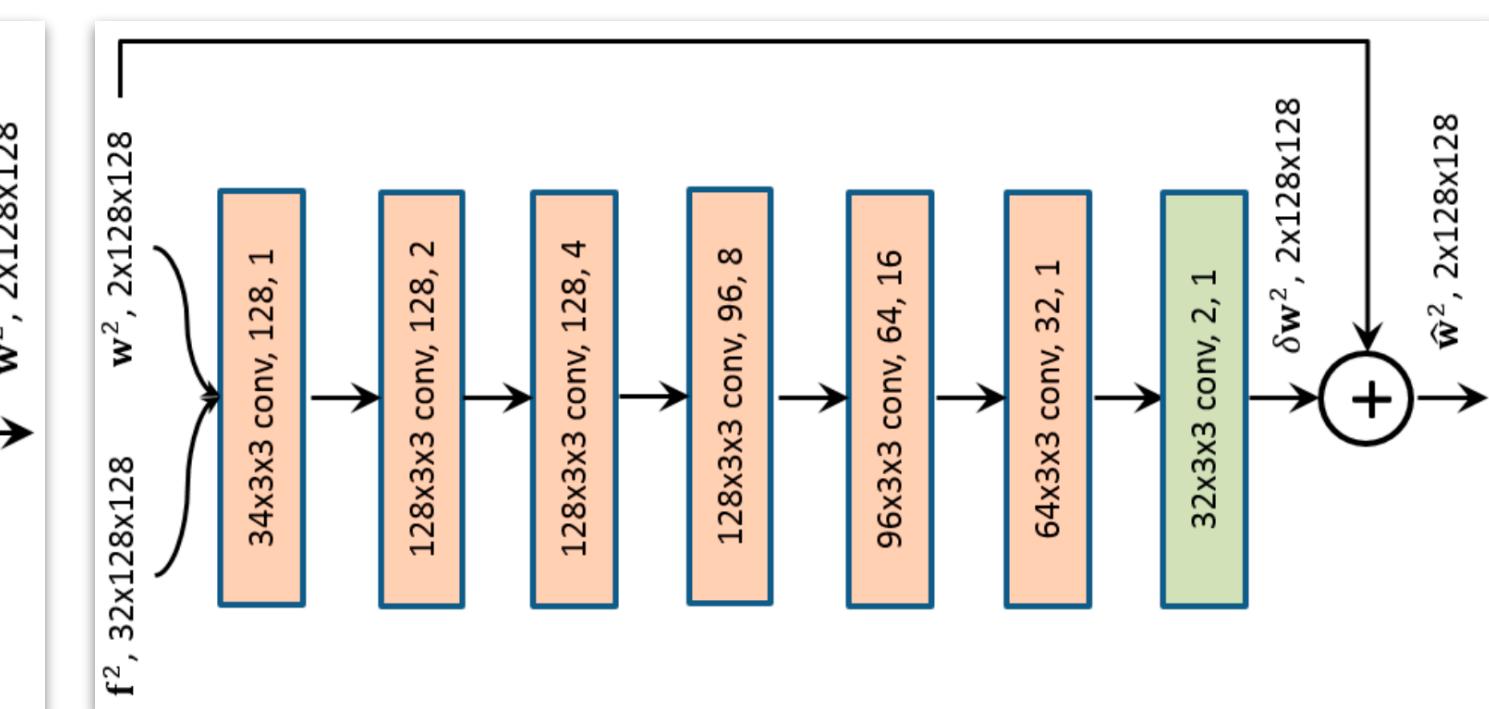
## Optical Flow Estimation



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

Fir

# Context Network



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

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

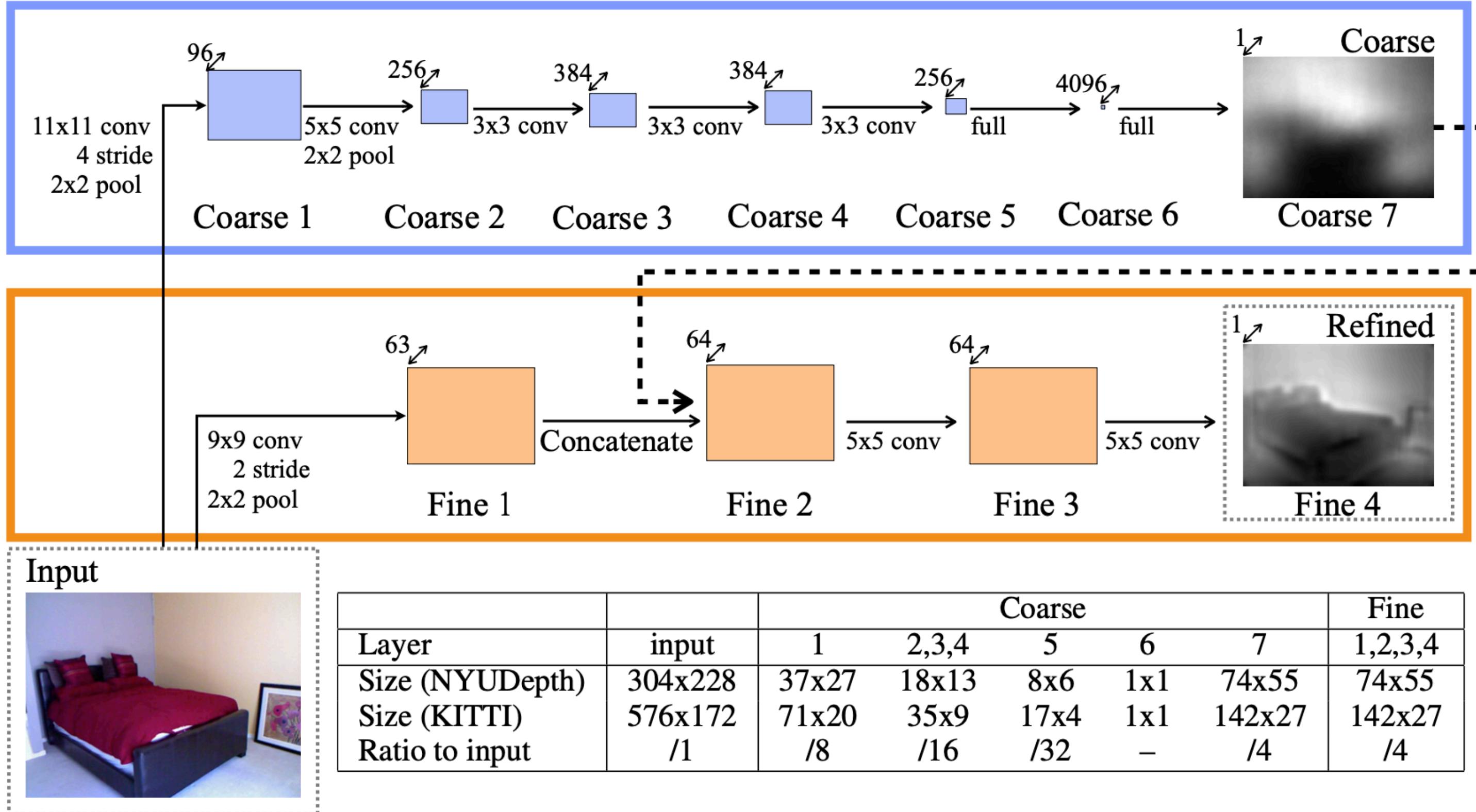
# Depth Map Prediction from a Single Image using a Multi-Scale Deep Network



[YouTube Video](#)

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

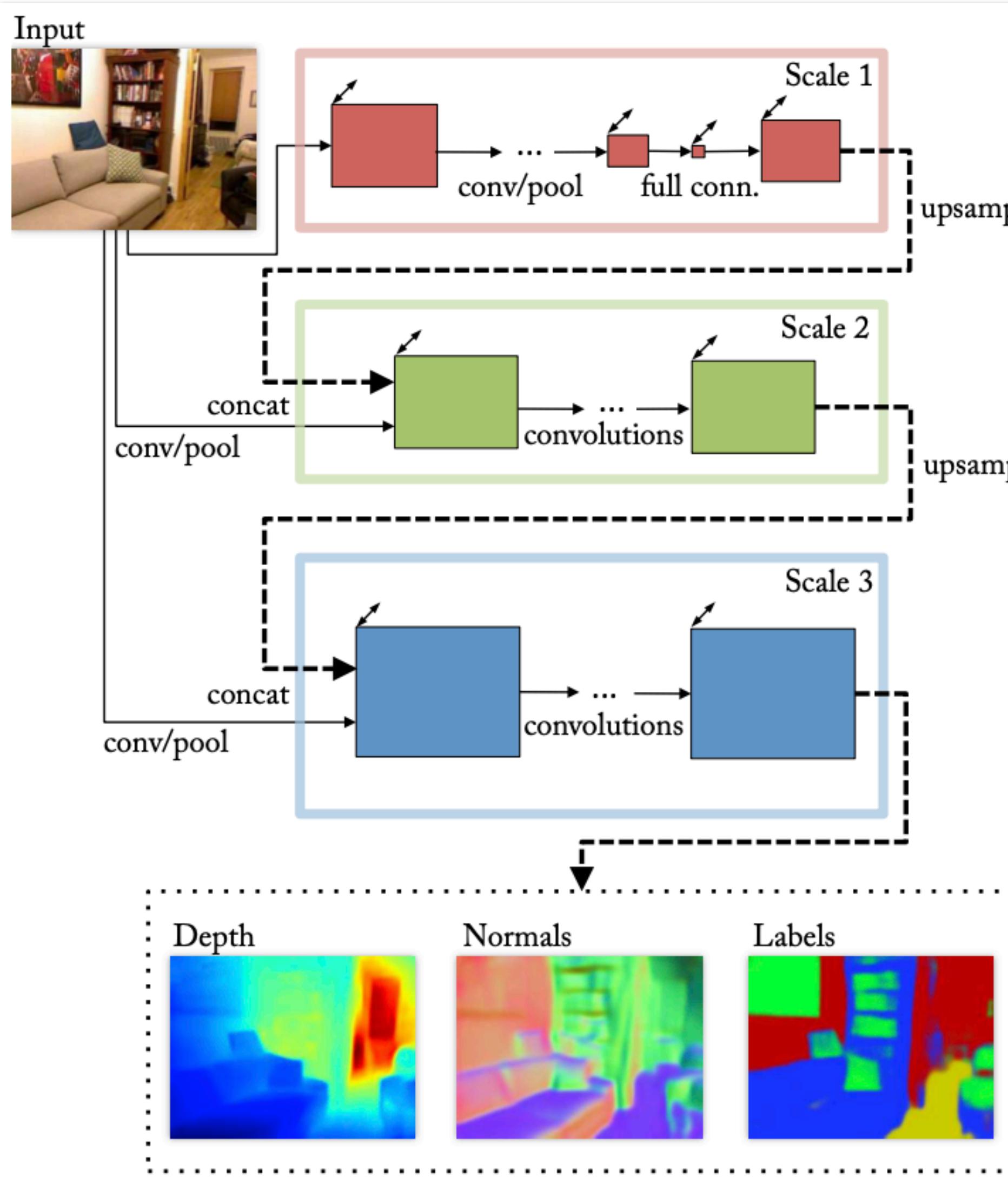
RMSE (log):  $\sqrt{\frac{1}{|T|} \sum_{y \in T} ||\log y_i - \log y_i^*||^2}$

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.

# Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture


[YouTube Video](#)


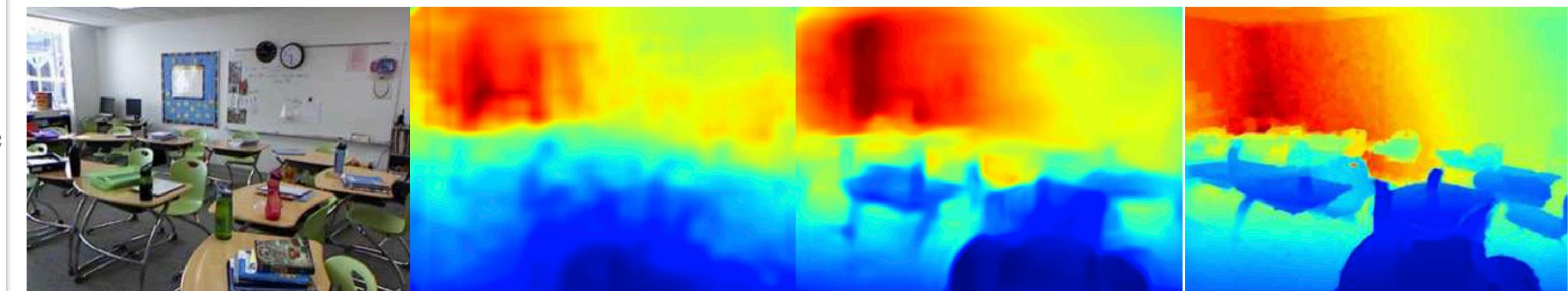
## Depth

$D \& D^* \rightarrow$  predicted & ground-truth log depth maps

$$d = D - D^*$$

$$L_{depth}(D, D^*) = \frac{1}{n} \sum_i d_i^2 - \frac{1}{2n^2} \left( \sum_i d_i \right)^2 + \frac{1}{n} \sum_i [(\nabla_x d_i)^2 + (\nabla_y d_i)^2]$$

Encourage predictions to have not only close-by values, but also similar local structure



## Surface Normals

Predict the  $x$ ,  $y$  and  $z$  components of the normal at each pixel

$$L_{normals}(N, N^*) = -\frac{1}{n} \sum_i N_i \cdot N_i^* = -\frac{1}{n} N \cdot N^*$$

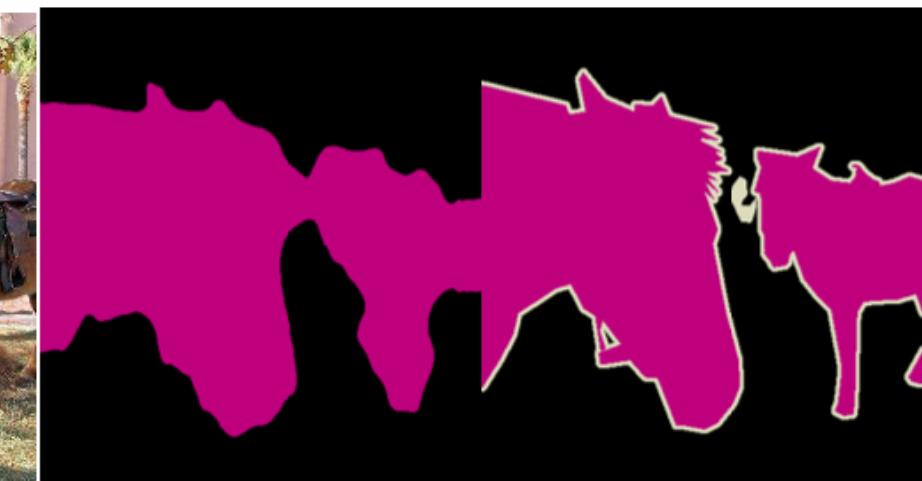
For ground truth targets, estimate normals from depth by fitting least-squares planes to neighboring sets of points in the point cloud.

## Semantic Labels

$$L_{semantic}(C, C^*) = -\frac{1}{n} \sum_i C_i^* \log(C_i)$$

$$C_i = e^{z_i} / \sum_c e^{z_{i,c}}$$

softmax



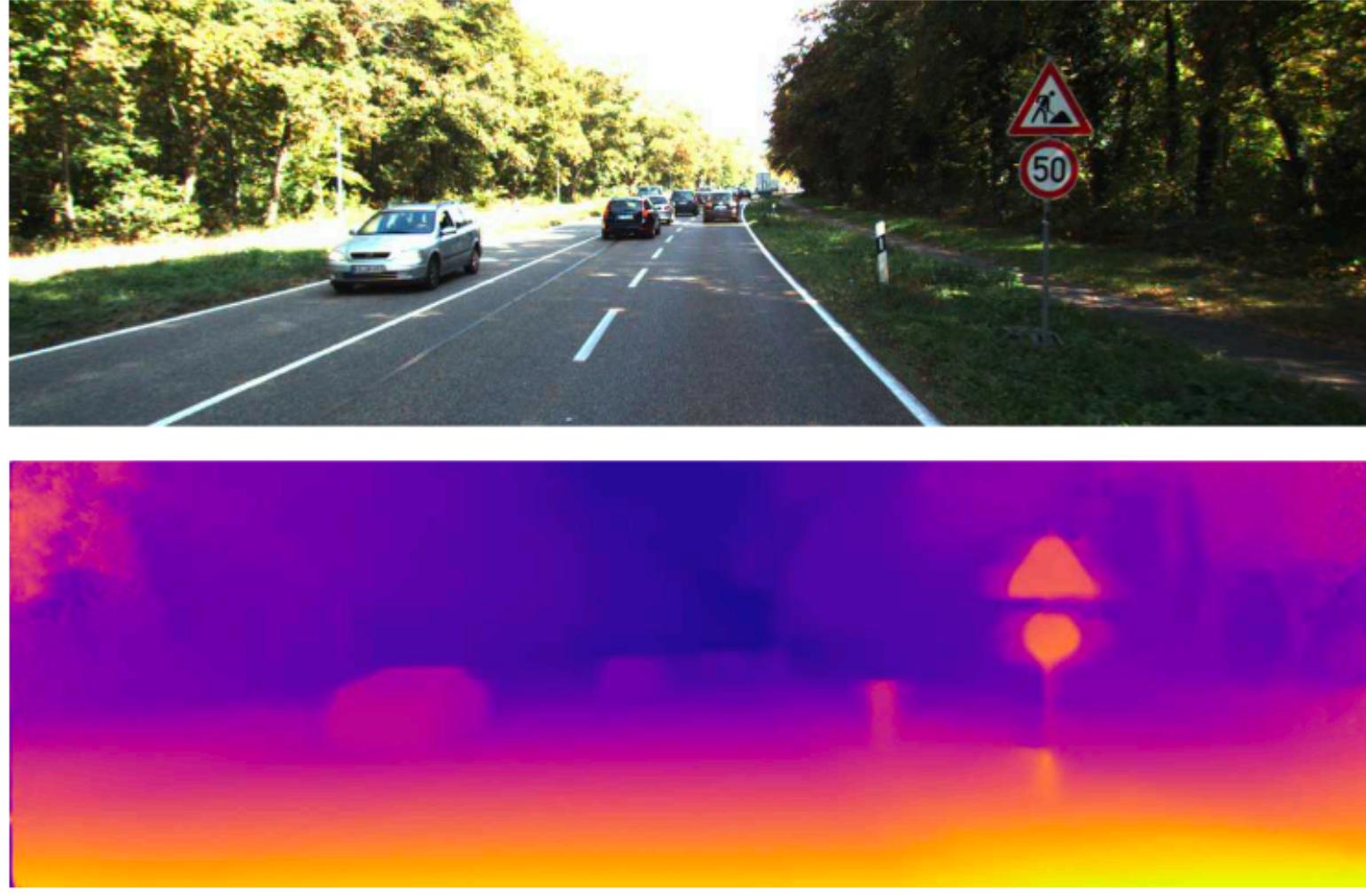


Boulder

# Unsupervised Monocular Depth Estimation with Left-Right Consistency



[YouTube Video](#)



$I \rightarrow$  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.

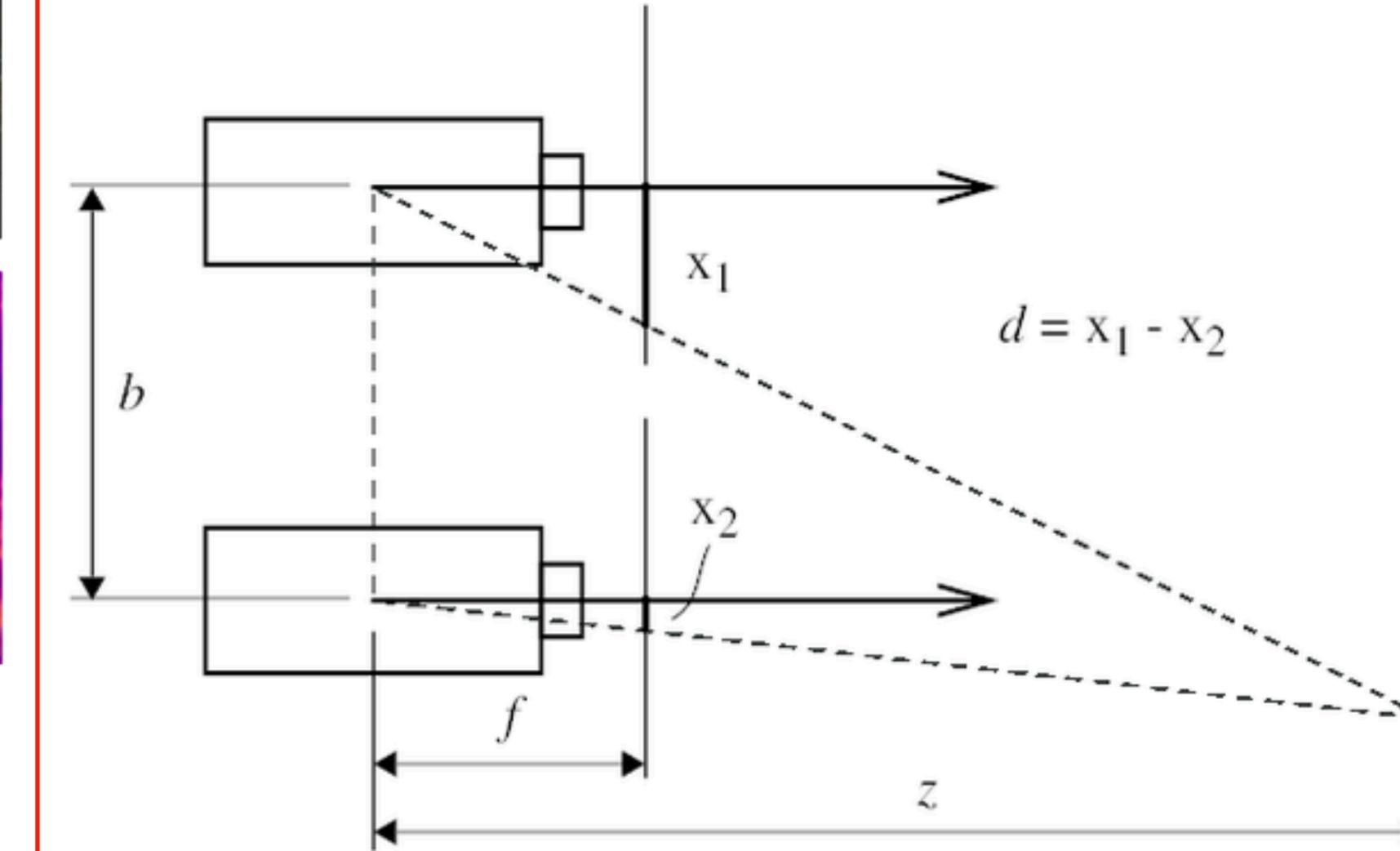
## Depth Estimation as Image Reconstruction

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  $\frac{d}{f} = \frac{1}{z} \rightarrow$  similar triangles

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

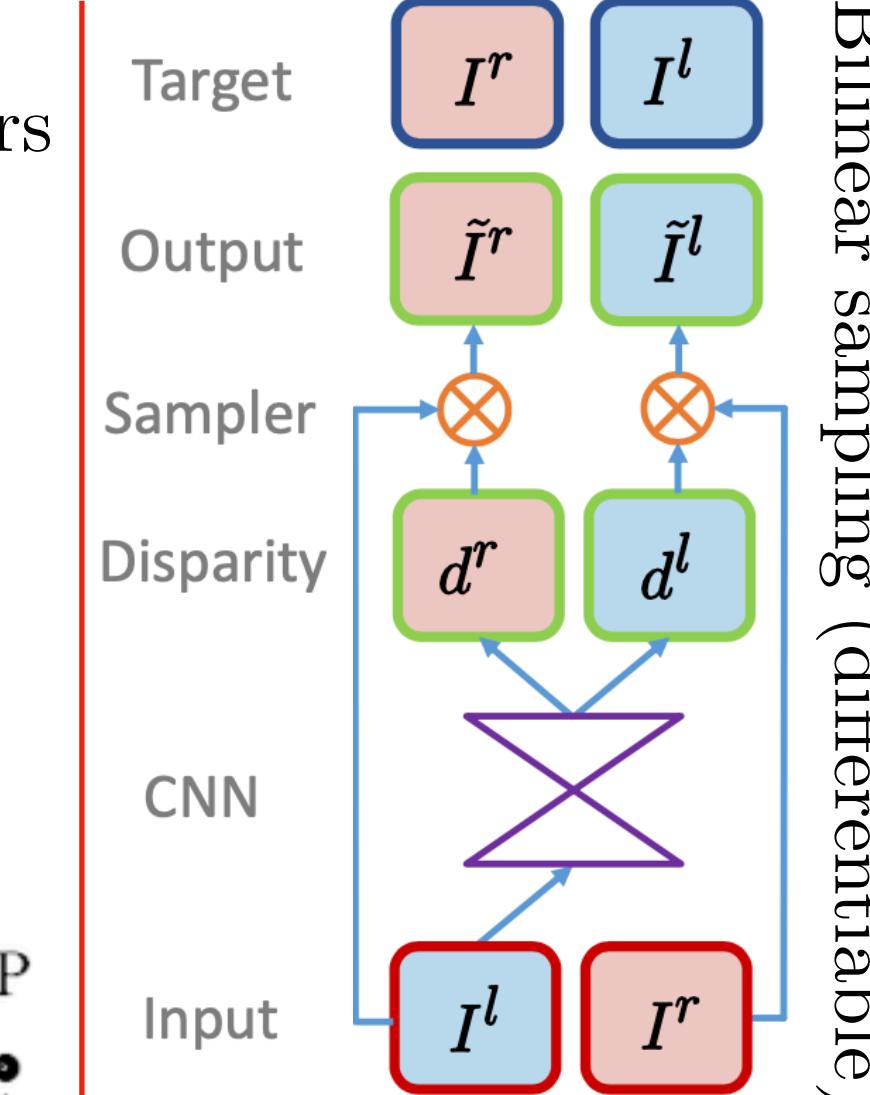
$z = \frac{bf}{d} \rightarrow$  recover depth from disparity

## Training Loss

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

$C_s \rightarrow$  loss at scale  $s$

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



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.

## 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](https://en.wikipedia.org/wiki/Structural_similarity#Algorithm)

## Disparity Smoothness

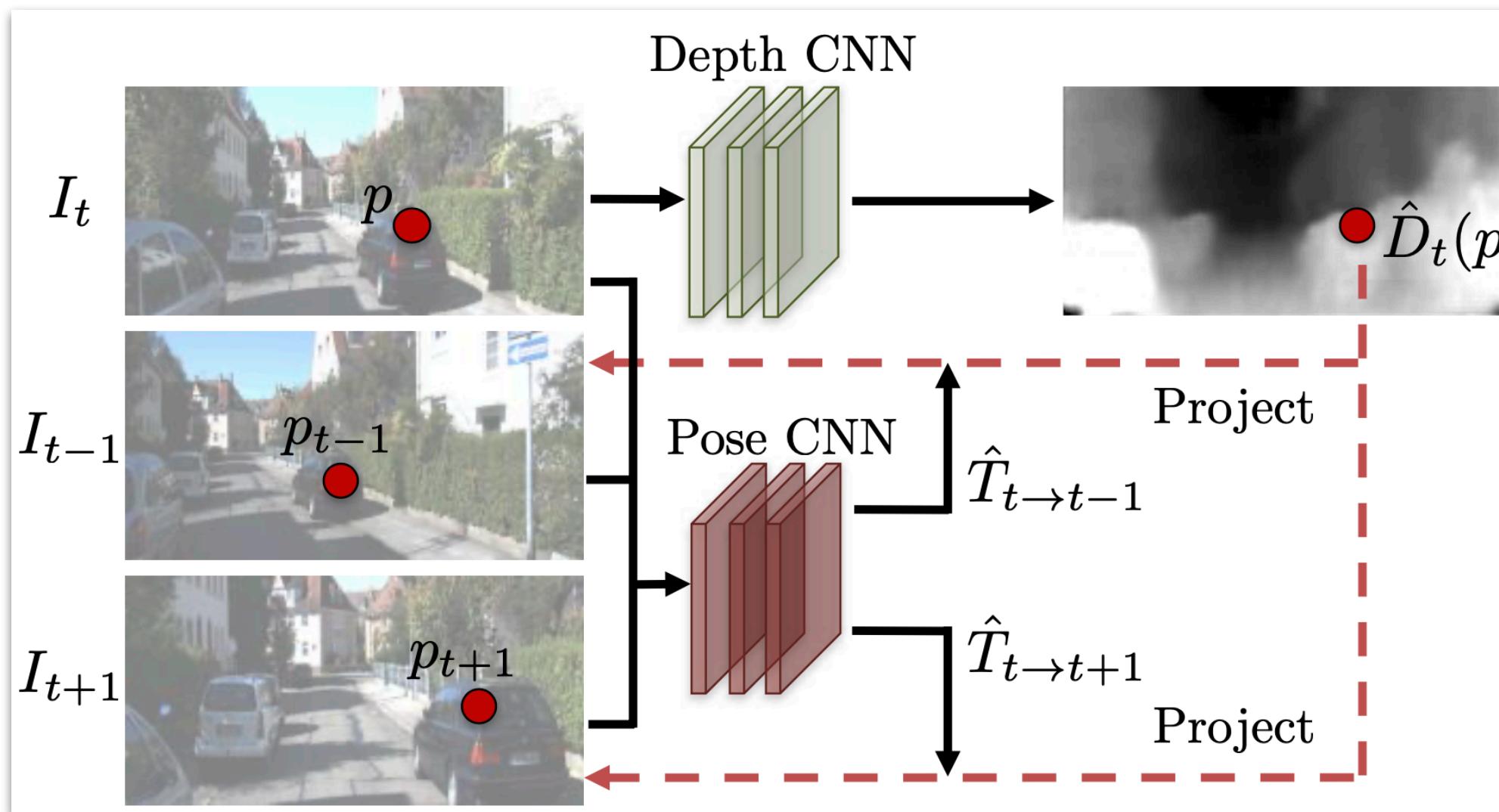
$$C_{ds}^l = \frac{1}{N} \sum_{i,j} |\partial_x d_{ij}^l| e^{-\|\partial_x I_{ij}^l\|} + |\partial_y d_{ij}^l| e^{-\|\partial_y I_{ij}^l\|}$$

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

## Left-Right Disparity Consistency

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

# Unsupervised Learning of Depth and Ego-Motion from Video


[YouTube Video](#)


**View synthesis as supervision**

$\langle I_1, \dots, I_N \rangle \rightarrow$  a training mage sequence

$I_t \rightarrow$  target view ( $t = 1, \dots, N$ )

$I_s \rightarrow$  source view ( $s = 1, \dots, N, s \neq t$ )

$N = 3$

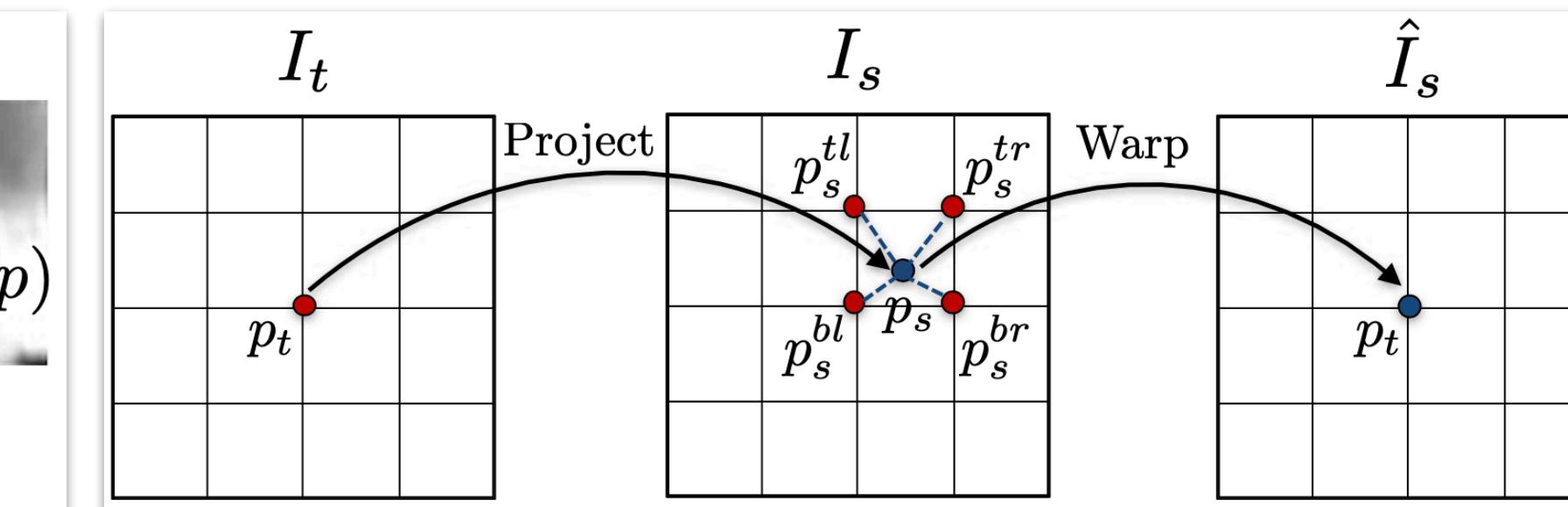
$$\mathcal{L}_{vs} = \sum_s \sum_p |I_t(p) - \hat{I}_s(p)| \rightarrow \text{view synthesis objective}$$

$p \rightarrow$  indexes over pixel coordinates

$\hat{I}_s \rightarrow$  source view  $I_s$  warped to the target coordinate frame based on a depth image-based rendering module

**Differentiable depth image-based rendering**

$$p_s \sim K \hat{T}_{t \rightarrow s} \hat{D}_t(p_t) K^{-1} p_t \rightarrow p_t \text{'s projected coordinates onto the source view } p_s$$



$p_t \rightarrow$  homogeneous coordinates of a pixel in the target view

$K \rightarrow$  camera intrinsics matrix

$\hat{D}_t \rightarrow$  predicted depth

$\hat{T}_{t \rightarrow s} \rightarrow$  predicted camera pose (transformation matrix)  
differentiable bilinear sampling mechanism

**Modeling the model limitation**

Implicit assumptions:

- 1) the scene is static without moving objects
- 2) there is no occlusion/disocclusion between the target view and the source views
- 3) the surface is Lambertian so that the photo-consistency error is meaningful.

$$\mathcal{L}_{vs} = \sum_{\langle I_1, \dots, I_N \rangle \in \mathcal{S}} \sum_p \hat{E}_s(p) |I_t(p) - \hat{I}_s(p)|$$

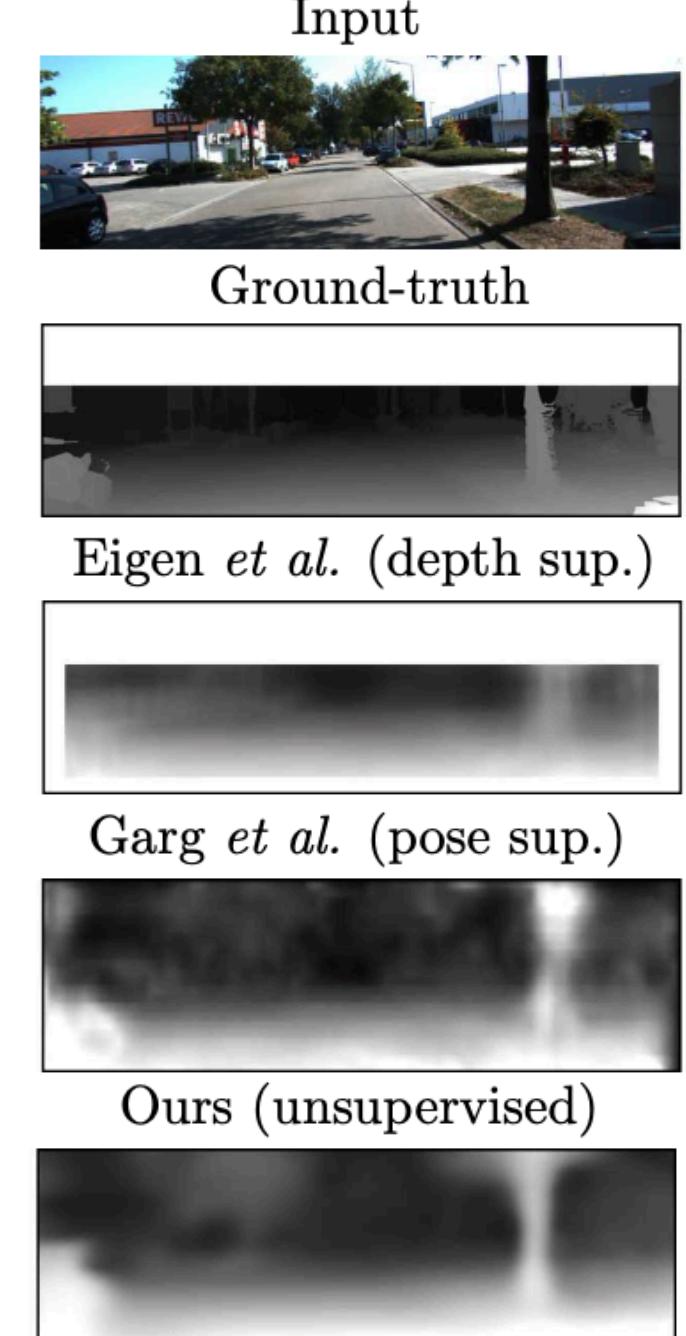
$\hat{E}_s \rightarrow$  explainability prediction network: pre-pixel softmax for each target-source pair

$$\mathcal{L}_{final} = \sum_l \mathcal{L}_{vs}^l + \lambda_s \mathcal{L}_{smooth}^l + \lambda_e \sum_s \mathcal{L}_{reg}(\hat{E}_s^l)$$

$l$  indexes over different image scales

encourages nonzero predictions by minimizing the cross-entropy loss with constant label 1 at each pixel location

$L_1$  norm of the second-order gradients for the predicted depth maps



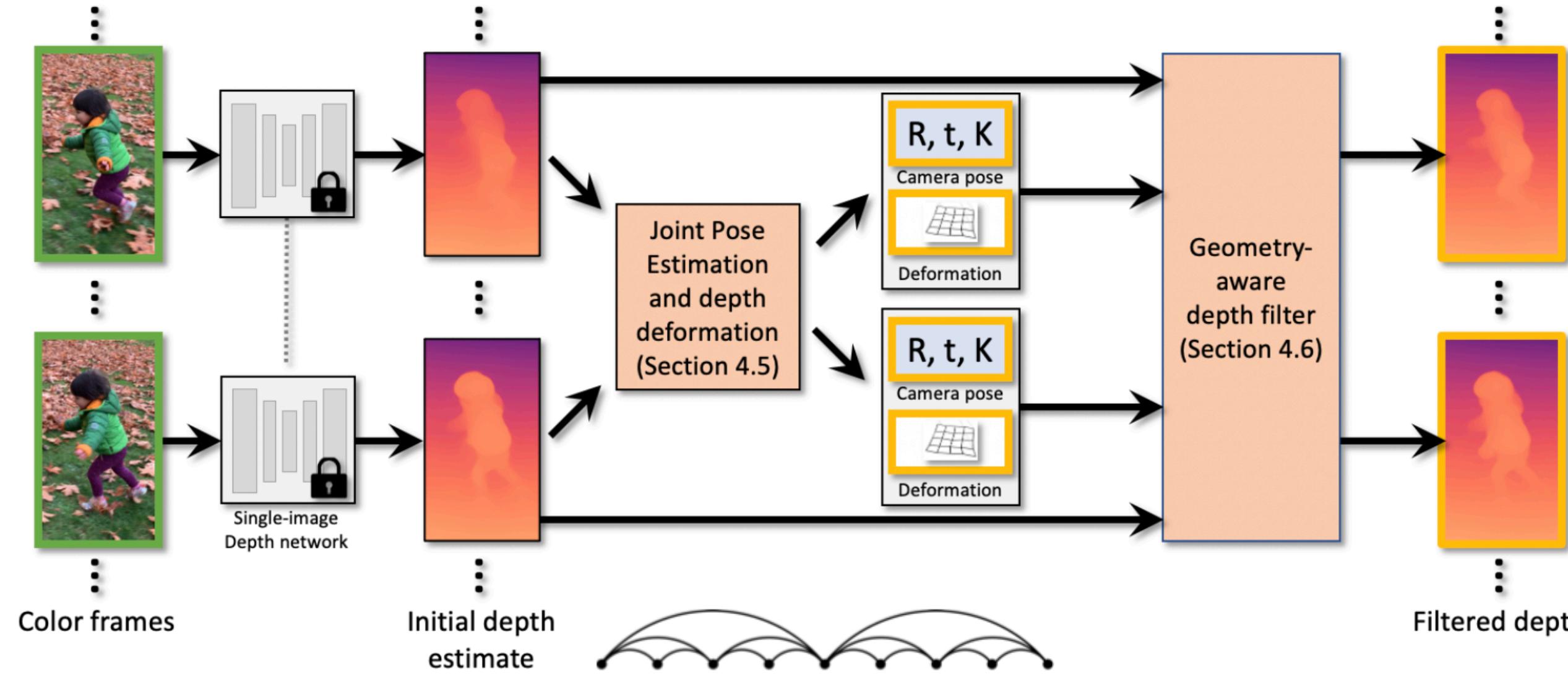


Boulder



[YouTube Video](#)

# Robust Consistent Video Depth Estimation



$P = \{(i, j) : |i - j| = k, i \bmod k = 0, k = 1, 2, 4, \dots\} \rightarrow$  set of frame pairs

$f_{i \rightarrow j} \rightarrow$  dense optical flow field

mapping a pixel in frame  $i$  to its corresponding location in frame  $j$

$m_{i \rightarrow j}^{\text{flow}} \rightarrow$  binary mask (forward-backward consistent flow pixels)

$m_i^{\text{dyn}} \rightarrow$  binary segmentation mask (Mask R-CNN)  
indicating likely dynamic pixels ("person", "animal", "vehicle")

## Depth Estimation

$p \rightarrow$  2D pixel coordinate

$c_i(p) = s_i d_i(p) \tilde{p} \rightarrow$  3D coordinate in frame  $i$ 's 3D camera coordinate system

$s_i \rightarrow$  per-frame scale coefficient

$d_i \rightarrow$  CNN-estimated depth map

$\tilde{p} = [p_x, p_y, 1]^T \rightarrow$  homogenous-augmented pixel coordinate

$c_{i \rightarrow j}(p) = K_j R_j^T (R_i K_i^{-1} c_i(p) + t_i - t_j) \rightarrow$  projection of  $c_i(p)$  into the camera coordinate system of another frame  $j$

Kopf, Johannes, et al. "Robust Consistent Video Depth Estimation." ArXiv:2012.05901 [Cs], Dec. 2020. arXiv.org, <http://arxiv.org/abs/2012.05901>.

$$R_i, R_j \rightarrow \text{rotation of frames } i \text{ and } j \\ t_i, t_j \rightarrow \text{translation of frames } i \text{ and } j \\ K_i, K_j \rightarrow \text{intrinsics of frames } i \text{ and } j$$

$$\arg \min_{\theta^{\text{depth}}} \sum_{(i,j) \in P} \sum_{p \in m_{i \rightarrow j}^{\text{flow}}} \mathcal{L}_{i \rightarrow j}^{\text{repro}}(p), \quad \text{s.t. fixed } \theta^{\text{cam}}$$

$$\mathcal{L}_{i \rightarrow j}^{\text{repro}}(p) = \mathcal{L}_j^{\text{sim}}(c_{i \rightarrow j}(p), c_j(f_{i \rightarrow j}(p))) \rightarrow \text{reprojection loss}$$

$$\theta^{\text{cam}} = \{R_i, t_i, K_i, s_i\} \rightarrow \text{camera parameters}$$

$$\theta^{\text{depth}} \rightarrow \text{depth CNN parameters}$$

## Joint Pose Estimation and Depth Deformation

Reverse the roles of  $\theta^{\text{depth}}$  and  $\theta^{\text{cam}}$ !

Replace  $s_i$  with  $\phi_i(p) = \sum_k b_k(p) s_i^k$ .

$\phi_i(p) \rightarrow$  depth deformation model

$s_i^k \rightarrow$  scale factors (defined on a regular grid across the image)

$b_k(p) \rightarrow$  bilinear coefficients

## Geometry-aware Depth Filter

$$d_i^{\text{final}}(p) = \sum_{q \in N(p)} \sum_{j=i-\tau}^{i+\tau} z_{j \rightarrow i}(\tilde{f}_{i \rightarrow j}(q)) w_{i \rightarrow j}(q)$$

$z_{j \rightarrow i} \rightarrow$  reprojected depth (scalar  $z$ -component of  $c_{j \rightarrow i}$ )

$\tilde{f} \rightarrow$  flow field between far frames

(chaining flow maps between consecutive frames)

$$w_{i \rightarrow j}(q) = \exp(-\lambda_f \mathcal{L}^{\text{ratio}}(c_i(p), c_{j \rightarrow i}(\tilde{f}_{i \rightarrow j}(q))))$$

– Sintel – DAVIS – Cellphone Vides Datasets

$$\mathcal{L}^{\text{ratio}}(a, b) = \frac{\max(a_z, b_z)}{\min(a_z, b_z)} - 1$$



Boulder



# Questions?

[YouTube Playlist](#)

---