Recent Advances in Computer Vision

UnFlow: Unsupervised Learning of Optical Flow with a Bidirectional Census Loss

Simon Meister, Junhwa Hur, Stefan Roth

Presented by:

Soumyadeep Bhattacharjee

st164269@stud.uni-stuttgart.de

04.06.2019

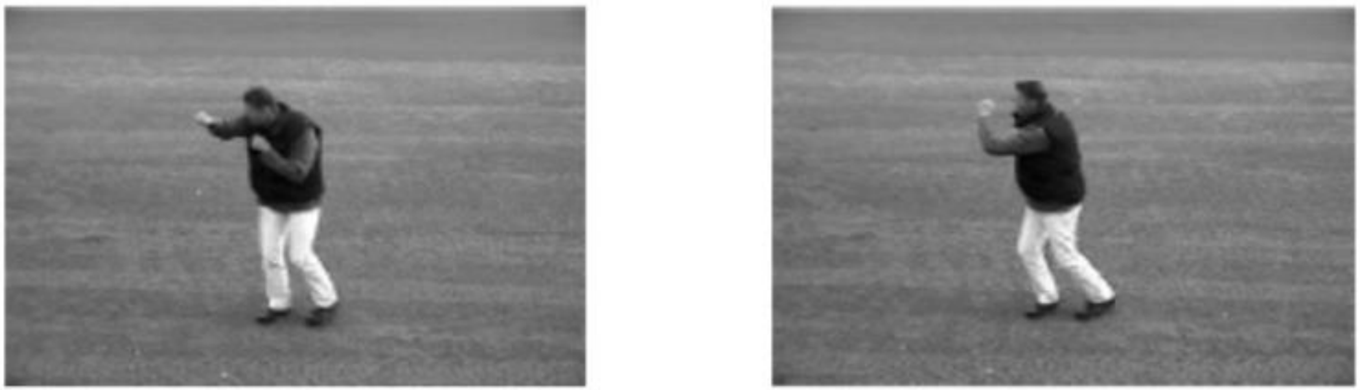
Contents

Introduction

Introduction

Optic Flow

Inter-frame displacement at each pixel resulting from spatio-temporal brightness variations



(a) Frame t (b) Frame $t + 1$

Introduction

Methods to determine Optic Flow

Traditional Methods:

- Block-matching
- Lucas–Kanade/Horn–Schunck
- General variational methods – extensions and modifications

Recent advances:

Deep Learning methods using Convolutional Neural Networks (CNNs)

- **Supervised**: Predict the output from the labelled input data.
- **Unsupervised**: Learn inherent structure from unlabeled input data.
- **Semi-supervised**: Mixture of supervised and unsupervised techniques.

Motivation

Motivation

Problem:

- Deep Learning methods are driven by large amounts of training data.
- Dense per-pixel ground truth is difficult to obtain for real world scenes

Alternative:

Using Synthetic Datasets [Sintel/SYNTHIA]

Pros:

- Abundant ground truth (easier to obtain)

Cons:

- Trade realism for quantity

Motivation

Domain Mismatch:

Intrinsic differences between Training and Testing images

Training Domain
(Flying Chairs, SYNTHIA)

Domains of interest
(KITTI, Middlebury)

Motivation

Solution:

An Unsupervised Loss based on:

- Occlusion aware bidirectional flow estimation
- Robust Census transform

This loss is used to train a CNN (end-to-end) to predict dense optic flow and *does not* assume access to the velocity pixel labels of the ground-truth

Advantages:

- Circumvents the need to leverage ground truth flow
- Removes dependency on Synthetic datasets

Related work

Supervised Learning

- The first end-to-end supervised learning of CNNs for optical flow was **FlowNet** (Dosovitskiy et al. 2015).

- A follow up work (Ilg et al. 2017) introduced the **FlowNet2** family of networks.
  - Improved by stacking multiple FlowNet networks for iterative refinement

- Other Hybrid methods (use CNN to extract relevant features) include:

  - **PatchBatch** (Gadot and Wolf 2016),
  - **Deep Discrete Flow** (Guney and Geiger 2016)

Related work

Unsupervised Learning

- Yu, Harley, & Derpanis (2016) suggested an unsupervised method based on FlowNet
    - Simplistic proxy loss based on the classical brightness constancy.
- Zhu et al. 2017 :
    - Combined proxy loss with proxy ground truth from a classical optical flow algorithm.
- Zhu and Newsam 2017
    - Replaced the underlying FlowNet architecture.

All of the unsupervised approaches were outperformed by the supervised FlowNetS.

Aim of this paper

- To investigate possible ways for improving the accuracy of unsupervised approaches
- To uncover whether Unsupervised methods are a viable alternative or addition to supervised learning.

Unsupervised Loss

Unsupervised Loss

Let $I_1, I_2 : P \rightarrow \mathbb{R}^3$ be two temporally consecutive frames.

Our goal is to estimate:

- Forward optical flow $\mathbf{w}^\mathrm{f} = \left(\mathrm{u}^\mathrm{f}, \mathrm{v}^\mathrm{f}\right)^\mathrm{T}$ from I_1 to I_2
- The inverse (backward) flow $\mathbf{w}^\mathrm{b} = \left(\mathrm{u}^\mathrm{b}, \mathrm{v}^\mathrm{b}\right)^\mathrm{T}$
- Symmetric Occlusion map = $\mathbf{o}^\mathrm{f}, \mathbf{o}^\mathrm{b}$

All loss terms are **symmetrical** (i.e., computed for both flow directions).

Unsupervised Loss

The unsupervised loss is a weighted sum of three individual loss terms:

$$E(\mathbf{w}^f, \mathbf{w}^b, o^f, o^b)$$

$$= \underbrace{E_D(\mathbf{w}^f, \mathbf{w}^b, o^f, o^b)}_{\text{Data Loss (Occlusion Aware)}} + \underbrace{\lambda_S E_S(\mathbf{w}^f, \mathbf{w}^b)}_{\text{Smoothness Loss}} + \underbrace{\lambda_c E_C(\mathbf{w}^f, \mathbf{w}^b, o^f, o^b)}_{\text{Consistency Loss}}$$

- λ_S : Constant penalty for deviation from smoothness
- λ_C : Constant penalty for deviation from consistency

Detection of occluded pixels

Idea: Pixels are marked occluded whenever the mismatch between $\boldsymbol{w}^f$ and $\boldsymbol{w}^b$ is too large.

- The forward occlusion flag $o_x^f = 1$, whenever the following constraint is violated:

$$\left|\mathbf{w}^{\mathrm{f}}(\mathbf{x}) + \mathbf{w}^{\mathrm{b}}\left(\mathbf{x} + \mathbf{w}^{\mathrm{f}}(\mathbf{x})\right)\right|^2 < \alpha_1\left(\left|\mathbf{w}^{\mathrm{f}}(\mathbf{x})\right|^2 + \left|\mathbf{w}^{\mathrm{b}}\left(\mathbf{x} + \mathbf{w}^{\mathrm{f}}(\mathbf{x})\right)\right|^2\right) + \alpha_2$$

- Similarly, the backward occlusion flag o_x^b is defined with $\boldsymbol{w}^f$ and $\boldsymbol{w}^b$ interchanged

$$\alpha_1 = 0.01, \alpha_2 = 0.5$$

Occlusion-aware Data loss

The occlusion-aware data loss is now defined as:

$$E_D\left(\mathbf{w}^f, \mathbf{w}^b, o^f, o^b\right) =$$

$$\sum_{\mathbf{x} \in \mathbf{P}} \Big[\left(1 - o_{\mathbf{x}}^f\right). \rho\left(f_D\left(I_1(\mathbf{x}), I_2\left(\mathbf{x} + \mathbf{w}^f(\mathbf{x})\right)\right)\right) + o_{\mathbf{x}}^f \lambda_p$$

$$+ \left(1 - o_{\mathbf{x}}^b\right). \rho\left(f_D\left(I_2(\mathbf{x}), I_1\left(\mathbf{x} + \mathbf{w}^b(\mathbf{x})\right)\right)\right) + o_{\mathbf{x}}^b \lambda_p \Big]$$

- $f_D(I_1(x), I_2(x'))$: Photometric Difference
- $\rho(x) = (x^2 + \epsilon^2)^\gamma$: Robust Generalized Charbonnier Penalty function ($\gamma = 0.45$)
- λ_p : Constant penalty for occluded pixels to avoid the trivial solution

Occlusion-aware Data loss

Drawbacks:

The brightness constancy is NOT invariant to illumination changes common in realistic situations

Alternative:

The ternary census transform (Zabih and Woodfill 1994; Stein 2004) is used to compensate for additive and multiplicative illumination changes.

Census Transform

Idea: To extract order-dependent information from local neighborhood

Benefit: Provides photometric invariance to changes that preserve the intensity order within a locality

21	9	13
18	11	11
24	10	7

→

$21 < 11$	$9 < 11$	$13 < 11$
$18 < 11$	X	$11 < 11$
$24 < 11$	$10 < 11$	$7 < 11$

=

0	1	0
0	X	0
0	1	1

$$b(x,y) := \begin{cases} \mathbf{0} & \text{if } x \leq y \\ \mathbf{1} & \text{if } x > y \end{cases}$$

Comparison with the central pixel yields a binary census signature: $(0,1,0,0,0,0,1,1)^T$

Census Transform

Ternary Census Transform

Benefit: Provides added robustness by introduction of threshold δ

$$t(x, y, \delta) := \begin{cases} \mathbf{1} & y - x > \delta \\ \mathbf{0} & |x - y| \le \delta \\ \mathbf{-1} & x - y > \delta \end{cases}$$

64	12	10
34	33	33
22	45	51

→

1	-1	-1
1	x	0
-1	1	1

↓

1	-1	-1	1	0	-1	1	1

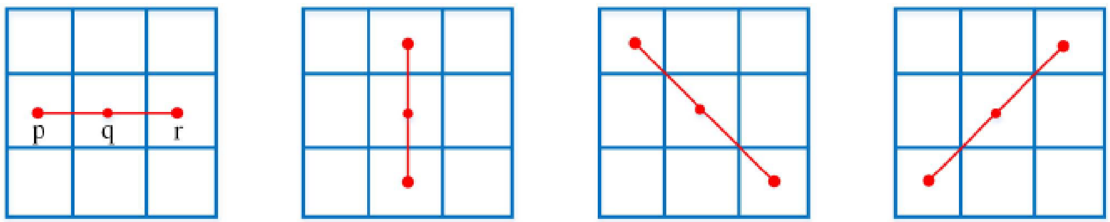
Stein, F. 2004. "Efficient computation of optical flow using the census transform."

Smoothness Loss

Idea: Encourage collinearity of neighboring flows; achieve regularization

Method: Penalize large second derivatives of the flow

$$
\begin{aligned}
&E_S\left(\mathbf{w}^{f}, \mathbf{w}^{b}\right) \\
&= \sum_{x \in P} \sum_{(\mathbf{s},\mathbf{r}) \in N(\mathbf{x})} \rho\left(\mathbf{w}^{f}(s) - 2\mathbf{w}^{f}(x) + \mathbf{w}^{f}(r)\right) + \rho\left(\mathbf{w}^{b}(s) - 2\mathbf{w}^{b}(x) + \mathbf{w}^{b}(r)\right)
\end{aligned}
$$



$N(\boldsymbol{x})$ consists of the horizontal, vertical, and both diagonal neighborhood around x

Consistency Penalty

Idea: For **non-occluded** pixels forward and backward flow must be similar.

Method: Deviations from flow consistency is penalized.

$$E_C\left(\mathbf{w}^f, \mathbf{w}^b, o^f, o^b\right) = \sum_{x \in P} \left(1 - o^f_{\mathbf{x}}\right) \,.\, \rho\left(\mathbf{w}^f(\mathbf{x}) + \mathbf{w}^b\left(\mathbf{x} + \mathbf{w}^f(\mathbf{x})\right)\right) + \left(1 - o^b_{\mathbf{x}}\right) \,.\, \rho\left(\mathbf{w}^b(\mathbf{x}) + \mathbf{w}^f\left(\mathbf{x} + \mathbf{w}^b(\mathbf{x})\right)\right)$$

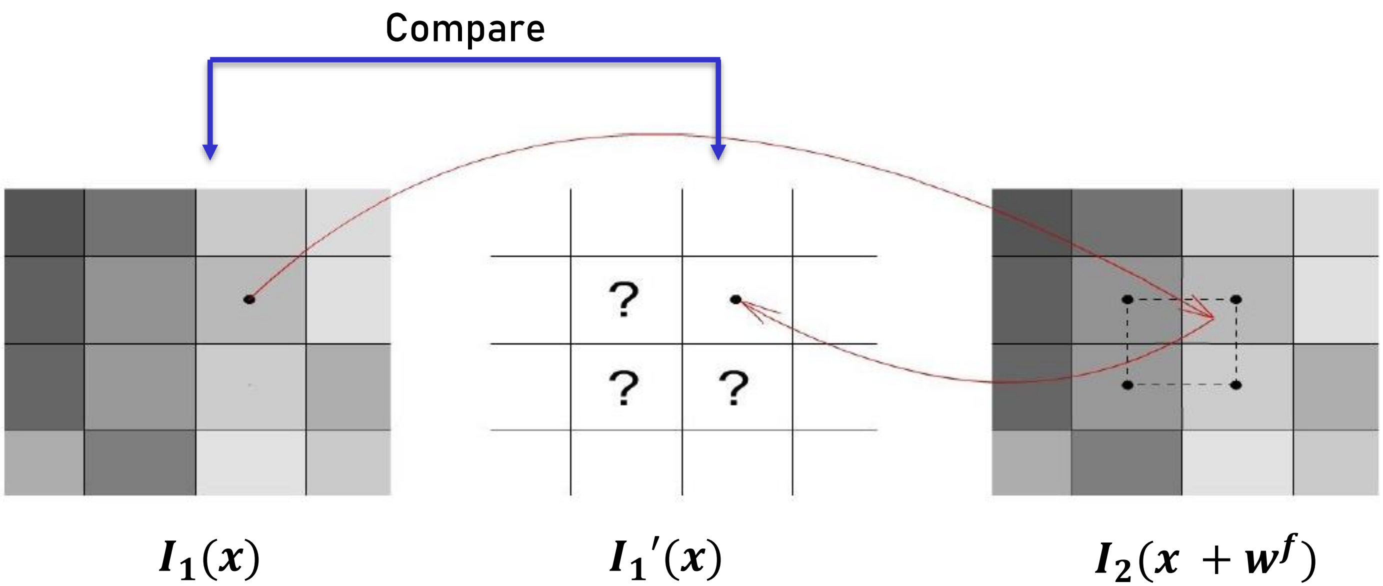
- w^f , w^b : Forward & backward flow
- $\rho(x)$: Charbonnier Penalty function
- o^f , o^b : Occlusion flags

Backward warping

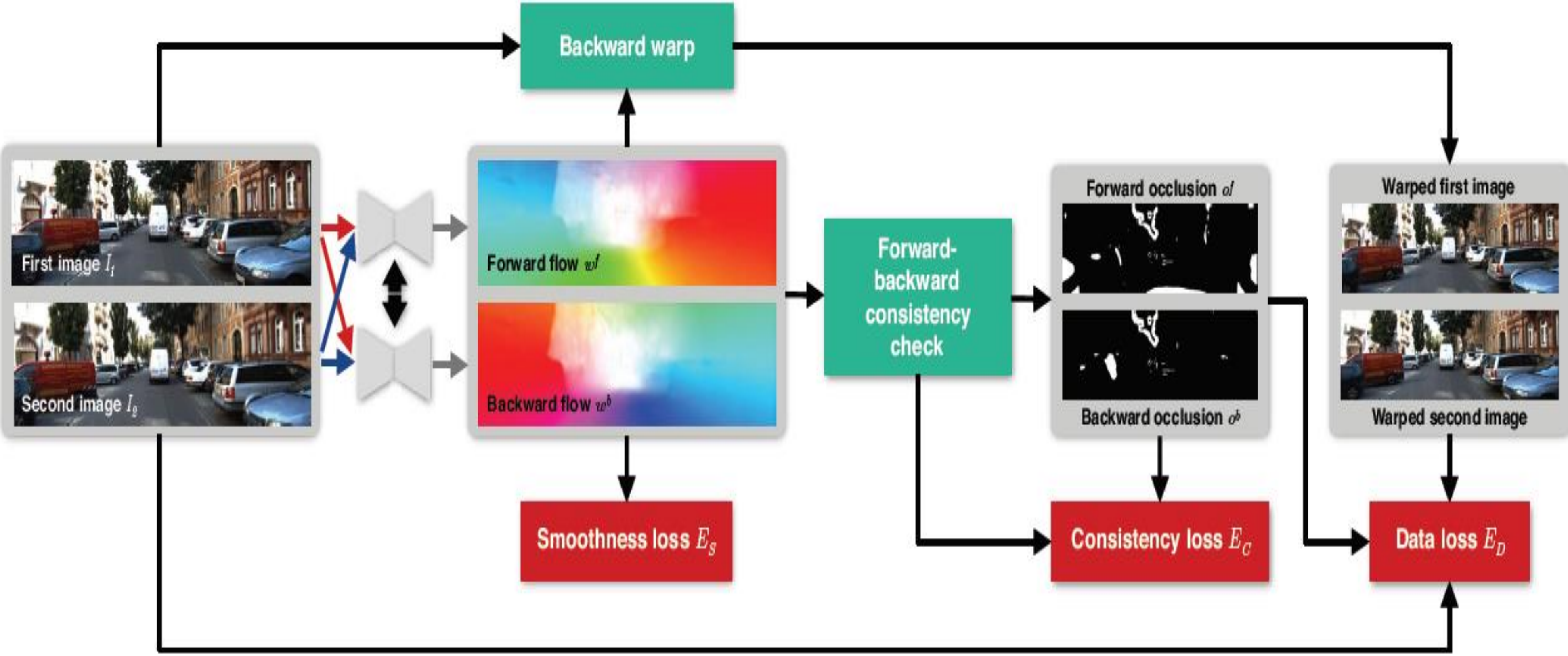
Idea: Compute losses in a subdifferentiable way for use with backpropagation

Method: Bilinear sampling at flow-displaced positions (*i.e.*, backward warping).

E.g.: To compare $\boldsymbol{I_1}(\boldsymbol{x})$ and $\boldsymbol{I_2}(\boldsymbol{x} + \boldsymbol{w}^{\boldsymbol{f}})$, we backward-warp $\boldsymbol{I_2}$ using $\boldsymbol{w}^{\boldsymbol{f}}$



Schematic of the unsupervised loss

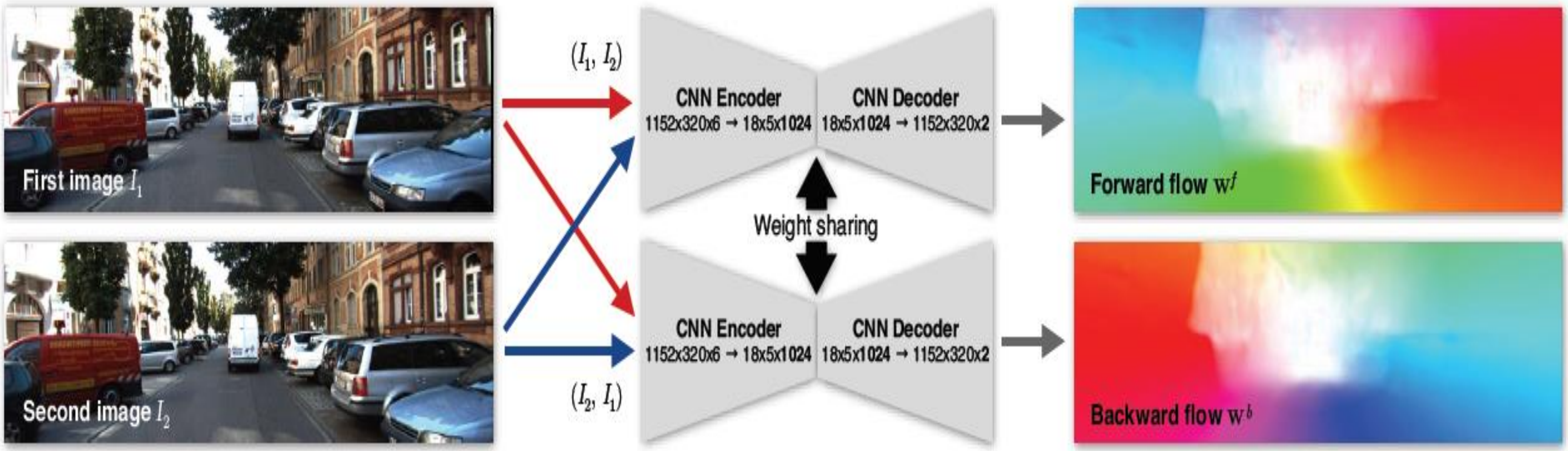


UnFlow

Training the CNN

- The CNN is bi-directionally trained using the comprehensive unsupervised loss (E) as the objective

Forward flow : Input images I_1(first frame) , I_2 (second frame)

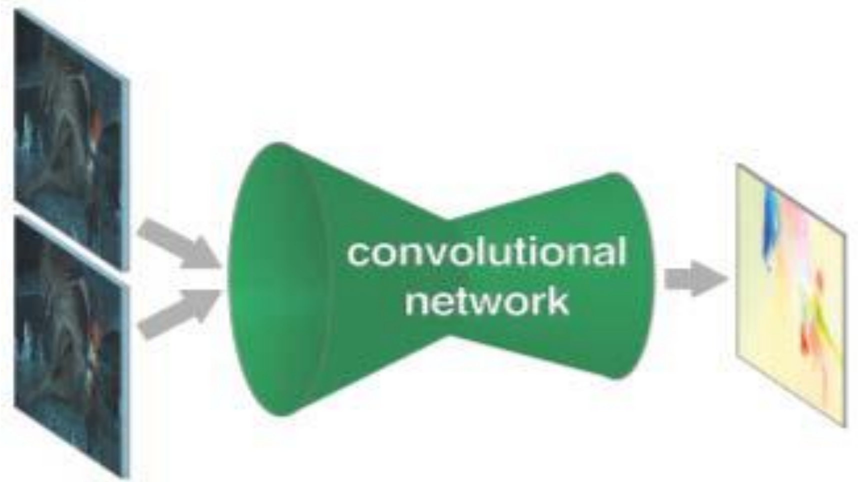


Backward flow : Input images I_2 , I_1 (order reversed)

Network architecture

The basic CNN used in UnFlow is based on FlowNet [Dosovitskiy et al. 2015]:

- FlowNet- Simple
- FlowNet - Correlated

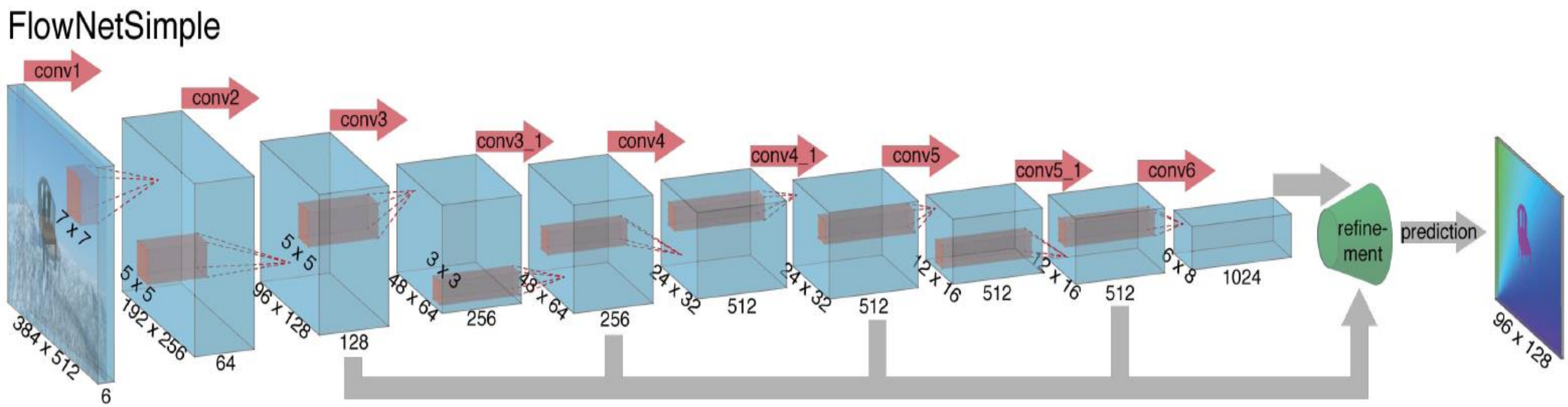


Encoder-Decoder architecture of FlowNet

Fischer et al. (2015), "FlowNet: Learning Optical Flow with Convolutional Networks"

Network architecture [Flow-Net Simple]

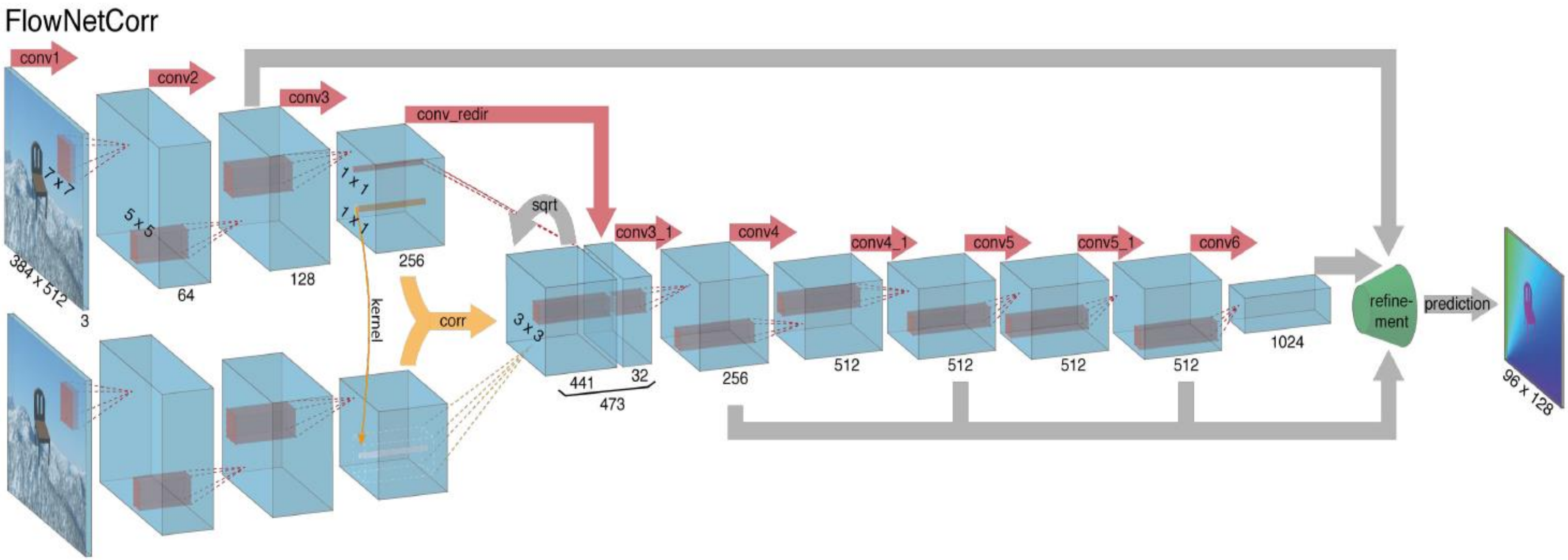
- Two input images are stacked together and fed through a generic network
- Allows the network to decide by itself how to process the image pair to extract the motion information.



Dosovitsky et al. (2015), "FlowNet: Learning optical flow with convolutional networks"

Network architecture [Flow-Net Corr]

- Processes two image frames in two separate input streams
- Explicitly correlates them at a later step
- Compresses the result with a CNN encoder down to a sixth of the original resolution

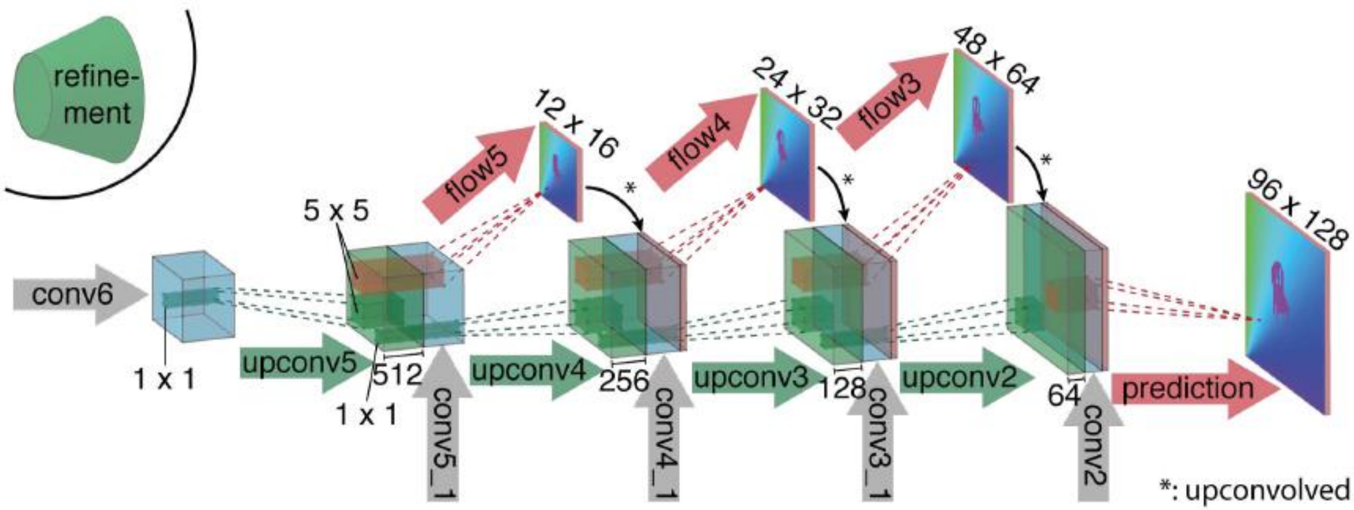


Dosovitsky et al. (2015), "FlowNet: Learning optical flow with convolutional networks"

Network architecture

The decoder part (**refinement network**),

- Implements a **skip-layer** architecture that combines information from various levels of the contractive part with "upconvolving" layers to iteratively refine the coarse flow predictions.
- Dense flow is predicted after each up-sampling.
- The last flow estimate is then bilinearly up-sampled to the original resolution.



Dosovitsky et al. (2015), "FlowNet: Learning optical flow with convolutional networks"

Computing the unsupervised loss

Idea:

To guide the learning process at multiple resolutions

Method:

- Calculate the loss for all intermediate predictions from the **refinement network**
- Combine them by taking a weighted average.

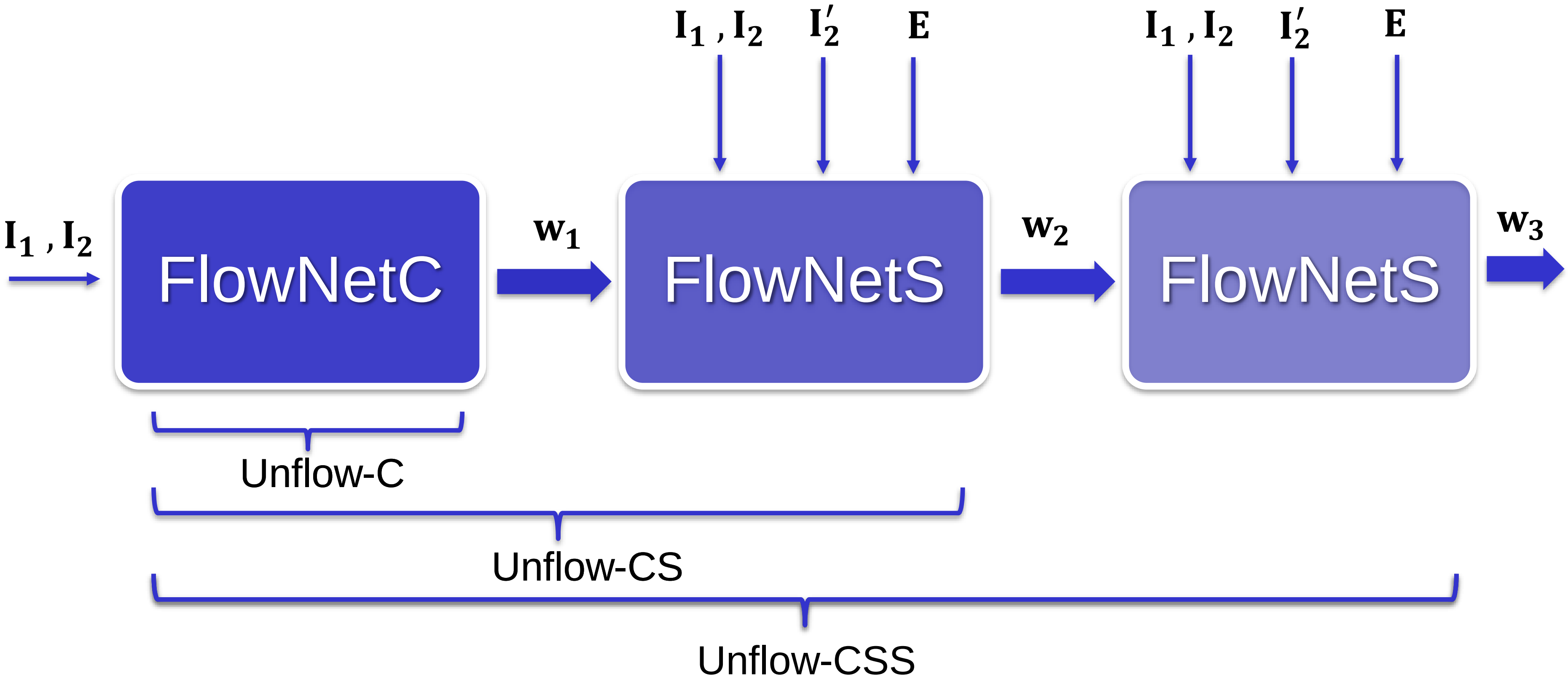
$$E_{unsup} = \sum_{i} \lambda_l^i E_i$$

E_i: Loss Evaluated from Layer i

λ_l^i : Weight

Iterative Refinement / Stacking

Idea: Improve results by training for large displacement flows



$\mathrm{I_1}, \mathrm{I_2}$: Images; $\mathrm{I_2'}$: Backward warped image ; E: Error estimate ; w : Flow estimate

Supervised loss for Fine-tuning (optional)

Idea:

Enable generic pre-training of supervised networks for datasets with limited amounts of ground truth.

Method:

Compute the network loss by comparing the bilinearly up-sampled **final flow** estimate to the **ground truth** flow at all pixels for which ground truth is available.

$$E_{sup}(\mathbf{w}^f) = \sum_{x \in P} v_x^f \, \rho(\mathbf{w}^f(\mathbf{x}) - \mathbf{w}_{gt}^f(\mathbf{x}))$$

$v_x^f = 1$, if there is valid ground truth at pixel x , else $v_x^f = 0$

Results

Metrics

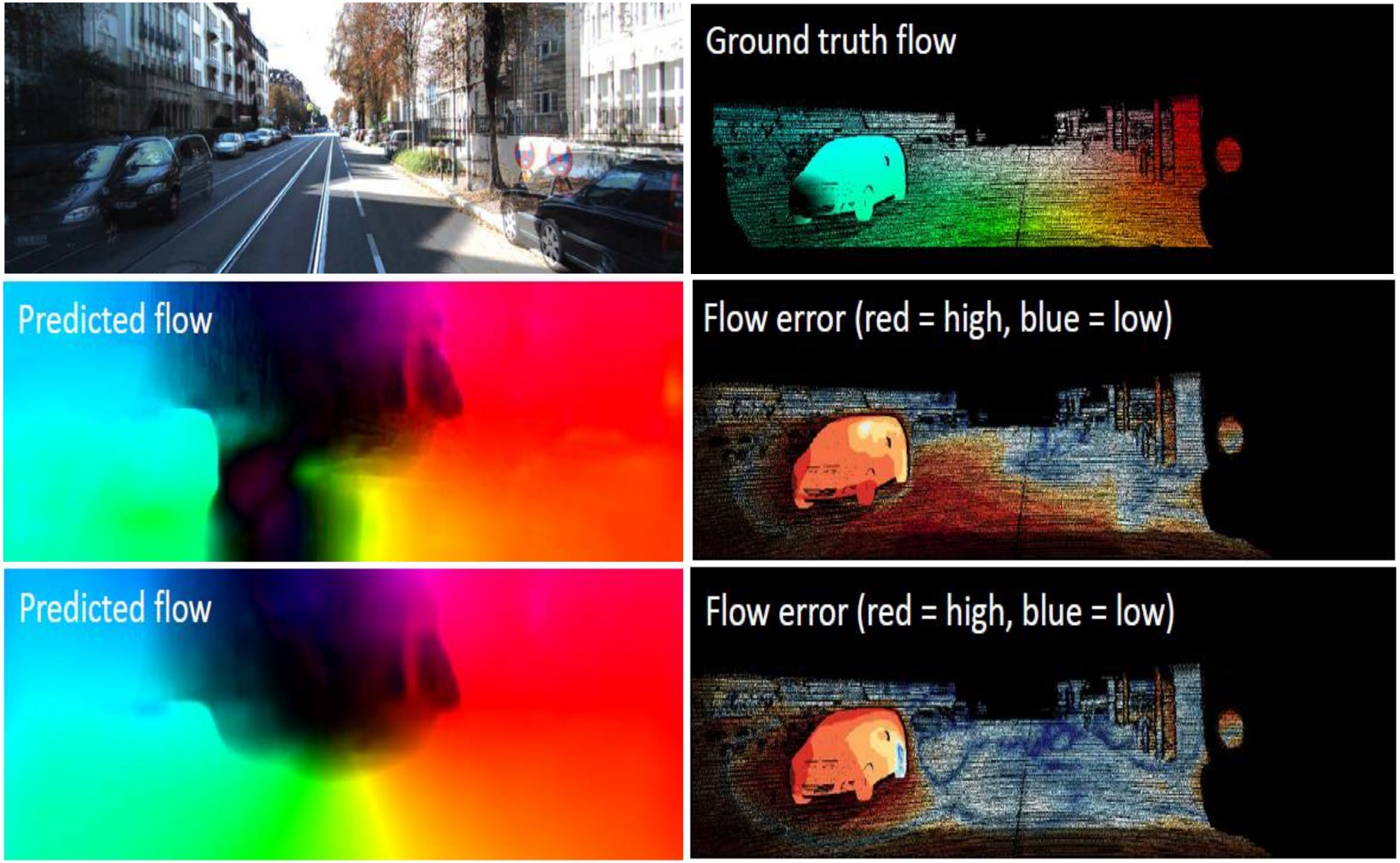
- Average Endpoint Error (AEE):
  Average Euclidean distance of prediction to ground-truth flow vectors
- KITTI Outliers:
  Ratio of pixels where flow estimate is wrong by both 3 pixels and 5% (at least)

Baseline vs. UnFlow (KITTI 2015)

Original Images (overlaid)

Baseline [Yu et al.]

UnFlow

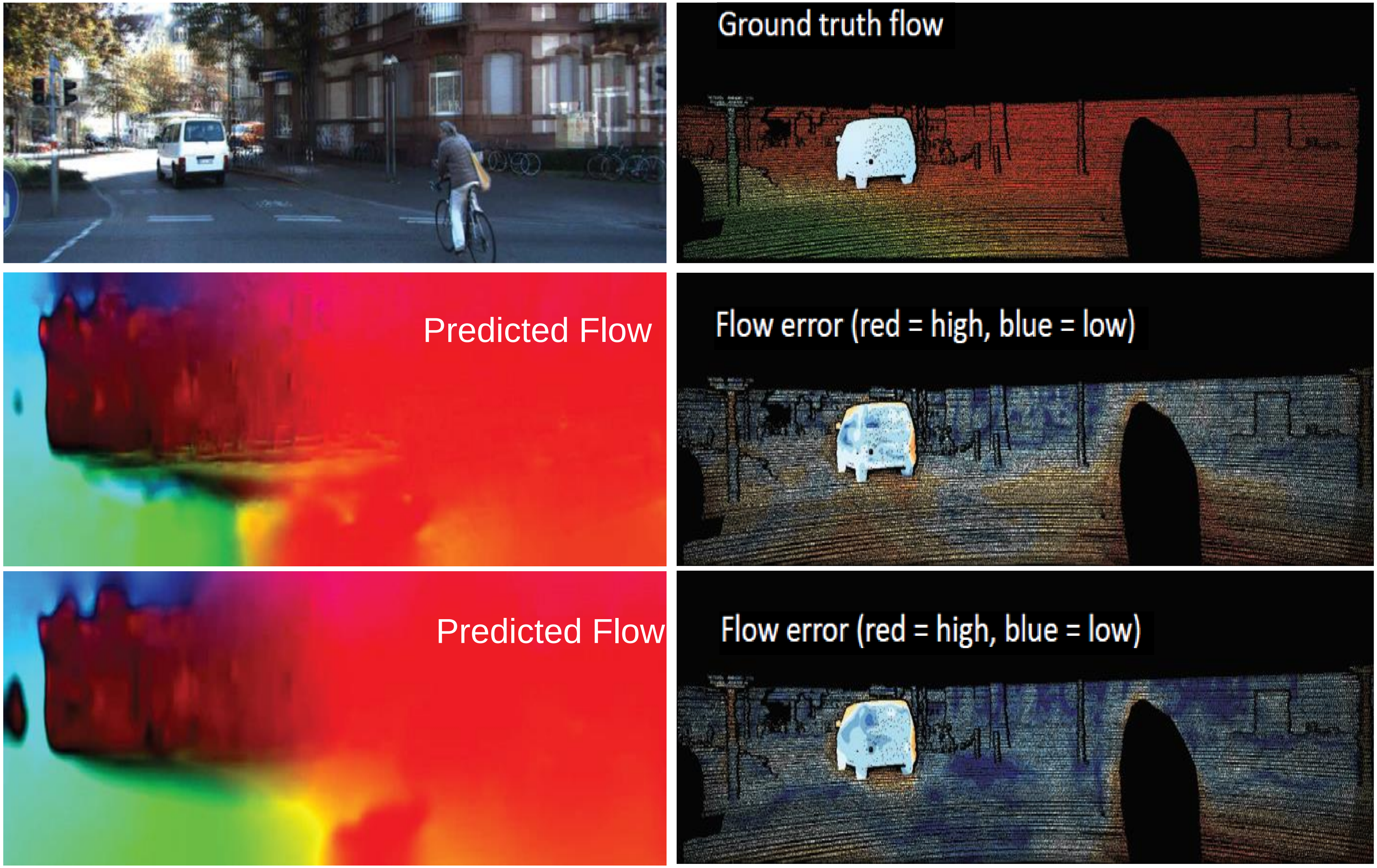


Baseline vs. UnFlow (KITTI 2015)

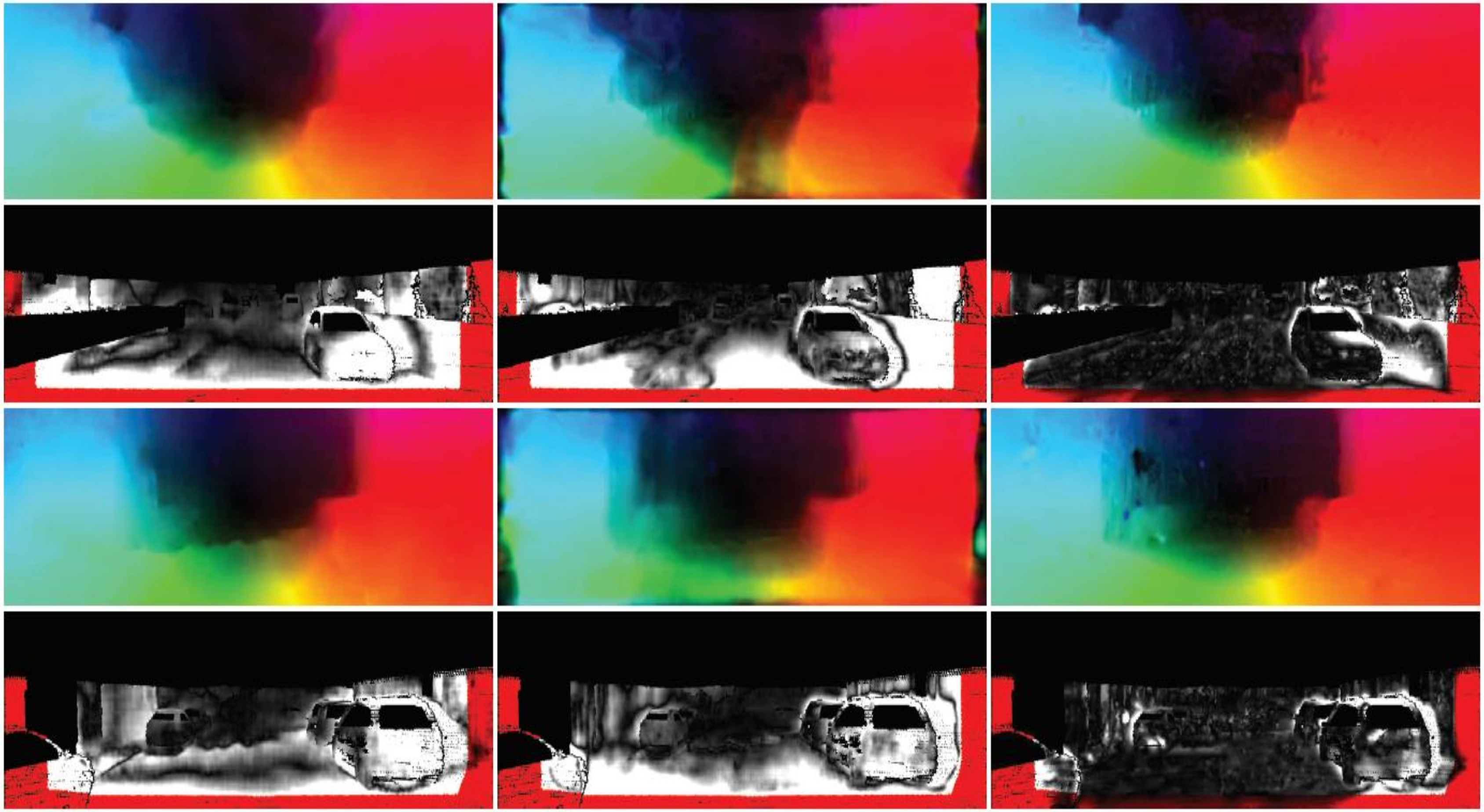
Original Images (overlaid)

Baseline [Yu et al.]

UnFlow



FlowNetS, UnupFlowNet vs. UnFlow (KITTI 2012)



FlowNetS | UnsupFlowNet | UnFlow

The KITTI 2012 error map scales linearly between 0 (black) and ≥ 5 pixels error (white).

Baseline (Yu et al.) vs. UnFlow-C [KITTI 2012]

- Brightness constancy → census loss
  - Reduces AEE by **35%**

Data loss	Smoothness	Occlusion	AEE (All)	Outliers (All)
Brightness	1st-order		7.20	31.93%
Census	1st-order		**4.66**	**20.85%**

Baseline (Yu et al.) vs. UnFlow-C [KITTI 2012]

- 1st → 2nd order smoothness
  - Reduces AEE by **5%** and outliers by **17%**

Data loss	Smoothness	Occlusion	AEE (All)	Outliers (All)
Brightness	1st-order		7.20	31.93%
Census	1st-order		4.66	20.85%
Census	**2nd-order**		**4.40**	**17.22%**

Baseline (Yu et al.) vs. UnFlow-C
[KITTI 2012]

- Forward-backward mechanisms (occlusion masking & consistency)
  - Reduces AEE by **14%**

Data loss	Smoothness	**Occlusion**	**AEE (All)**	**Outliers (All)**
Brightness	1st-order		7.20	31.93%
Census	1st-order		4.66	20.85%
Census	2nd-order		4.40	17.22%
Census	2nd-order	**Forward-backward check**	**3.78**	**16.44%**

Baseline (Yu et al.) vs. UnFlow-C [KITTI 2012]

- UnFlow reduces AEE and outliers by **48%**
- Similar observations on KITTI 2015

Data loss	Smoothness	Occlusion	AEE (All)	Outliers (All)
Brightness	1st-order		7.20	31.93%
Census	1st-order		4.66	20.85%
Census	2nd-order		4.40	17.22%
Census	**2nd-order**	**Forward-backward check**	**3.78**	**16.44%**

Previous Unsupervised networks vs. UnFlow

- UnFlow reduces AEE by up to **66%**

Method	**AEE (All) 2012 train**
UnsupFlownet [Yu et al.]	11.3
DSTFlow [Ren et al.]	10.43
UnFlow-C	**3.78**

Supervised networks vs. UnFlow

- UnFlow reduces AEE by up to **49%** (FlowNetS, 2012)

Method	**AEE (All) 2012 train**	**AEE (All) 2015 train**
UnsupFlownet [Yu et al.]	11.3	
DSTFlow [Ren et al.]	10.43	16.79
FlowNetS+ft [Dosovitskiy et al.]	7.5	
FlowNet2-C [Ilg et al.]		11.36
UnFlow-C	**3.78**	**8.80**

Supervised networks vs. UnFlow

- UnFlow even performs slightly better on off-domain data

Method	AEE (All) 2012 train	AEE (All) 2015 train	AEE (All) Middlebury
UnsupFlownet [Yu et al.]	11.3		
DSTFlow [Ren et al.]	10.43	16.79	
FlowNetS+ft [Dosovitskiy et al.]	7.5		0.98
FlowNet2-C [Ilg et al.]		11.36	
UnFlow-C	**3.78**	**8.80**	**0.88**

Limitations

- The amount of information that can be gained from the available data is limited by how realistically the problem is modeled by the loss.
- A parameter search for the weighting between the loss terms increases the total computation time for training a model on a new domain for the initial run

Summary

- Unsupervised learning of optic flow circumvents the need for ground truth
- The image data is modelled by an aggregated Loss incorporating:
  - Occlusion handling
  - Photometric invariance via Census transform
  - Second order flow smoothness
- This loss is used to train a CNN based on FlowNetS and FlowNetC
  - Iterative Refinement – UnFlow CSS
  - Optional Supervised loss for fine-tuning
- Reports substantial improvements over previous methods