Computer Vision 2
WS 2018/19

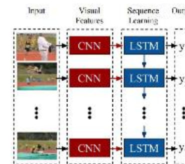
Part 17 – CNNs for Video Analysis II
22.01.2019

Prof. Dr. Bastian Leibe

RWTH Aachen University, Computer Vision Group
http://www.vision.rwth-aachen.de

Course Outline

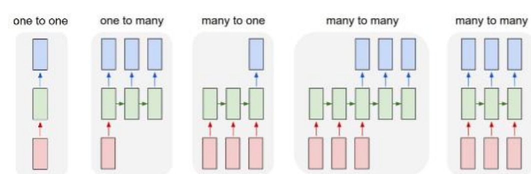
- Single-Object Tracking
- Bayesian Filtering
- Multi-Object Tracking
- Visual Odometry
- Visual SLAM & 3D Reconstruction
  - Online SLAM methods
  - Full SLAM methods
- Deep Learning for Video Analysis
  - CNNs for video analysis
  - CNNs for motion estimation
  - Video object segmentation



Topics of This Lecture

- Recap: CNNs for Video Analysis
- Matching and correspondence estimation
  - Metric learning
  - Spatial Transformer Networks
  - Correspondence networks
- Optical Flow Estimation
  - FlowNet
  - FlowNet2

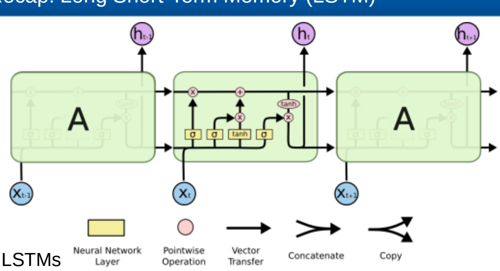
Recap: Recurrent Networks



- Feed-forward networks
  - Simple neural network structure: 1-to-1 mapping of inputs to outputs
- Recurrent Neural Networks
  - Generalize this to arbitrary mappings

Image source: Andrej Karpathy

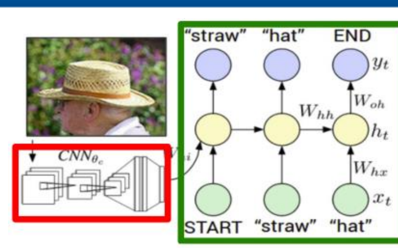
Recap: Long Short-Term Memory (LSTM)



- LSTMs
  - Inspired by the design of memory cells
  - Each module has 4 layers, interacting in a special way.
  - Effect: LSTMs can learn longer dependencies (~100 steps) than RNNs

Image source: Christopher Olah, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

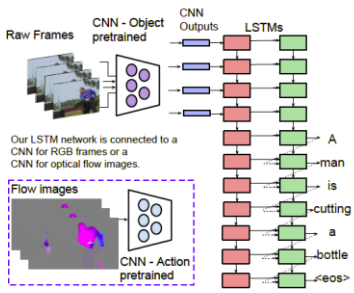
Recap: Image Tagging



- Simple combination of CNN and RNN
  - Use CNN to define initial state $\mathbf{h}_0$ of an RNN.
  - Use RNN to produce text description of the image.

Slide adapted from Andrej Karpathy

Recap: Video to Text Description



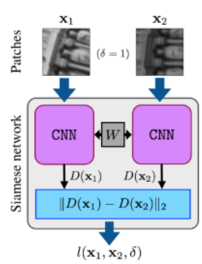
Source: Subhashini Venugopalan, ICCV'15

Topics of This Lecture

- Recap: CNNs for Video Analysis
- Matching and correspondence estimation
  - Metric learning
  - Spatial Transformer Networks
  - Correspondence networks
- Optical Flow Estimation
  - FlowNet
  - FlowNet2

Learning Similarity Functions

- Siamese Network
  - Present the two stimuli to two identical copies of a network (with shared parameters)
  - Train them to output similar values if the inputs are (semantically) similar.
- Used for many matching tasks
  - Face identification
  - Stereo estimation
  - Optical flow
  - …

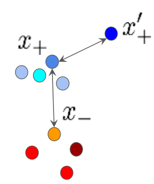


Metric Learning: Contrastive Loss

- Mapping an image to a metric embedding space
  - Metric space: distance relationship = class membership

$$\|f(x) - f(x_+)\| \to 0$$

$$\|f(x) - f(x_-)\| \geq m$$

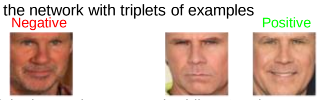


Yi et al., LIFT: Learned Invariant Feature Transform, ECCV 16

Slide credit: Christopher Choy

Metric Learning: Triplet Loss

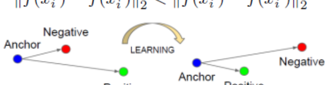
- Learning a discriminative embedding
  - Present the network with triplets of examples



  - Apply triplet loss to learn an embedding $f(\cdot)$ that groups the positive example closer to the anchor than the negative one.

$$\|f(x_i^a) - f(x_i^p)\|_2^2 < \|f(x_i^a) - f(x_i^n)\|_2^2$$

⇒ Used … ntification

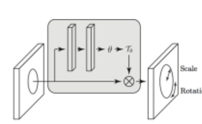


Patch Normalization with Spatial Transformer Nets

- Patch Normalization
  - Key component of local feature matching
  - Finding the scale and rotation
  - Invariant to perspective transformation

[SIFT patch normalization]

- Spatial Transformer Network
  - Adaptively apply transfomation



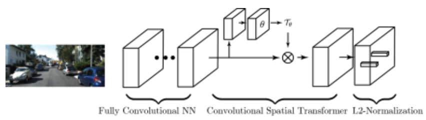
[Spatial Transformer Network]

Slide credit: Christopher Choy

Jaderberg et al., Spatial Transformer Network, NIPS 2015

Universal Correspondence Network

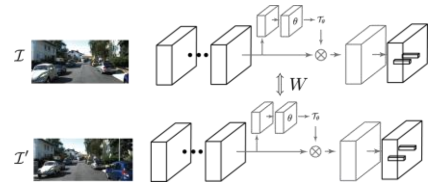
- Computing a patch descriptor



Slide credit: Christopher Choy

Universal Correspondence Network

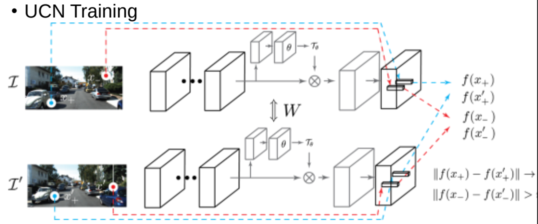
- Siamese architecture for matching patches



Slide credit: Christopher Choy

Universal Correspondence Network

- UCN Training



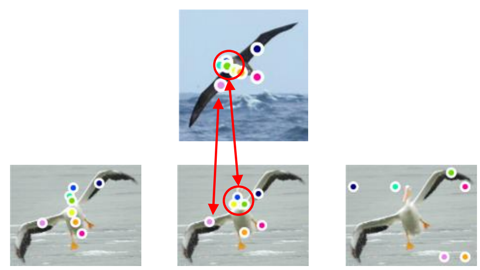
- Contrastive loss

$$\|f(x_+) - f(x'_+)\| \rightarrow 0$$

$$\|f(x_-) - f(x'_-)\| > m$$

Slide credit: Christopher Choy

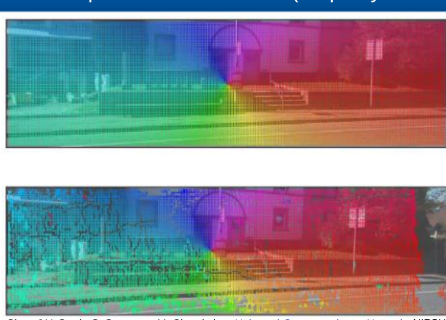
Semantic Correspondences with UCN



Ground truth | UCN | VGG Conv4

Slide credit: Christopher Choy

Exact Correspondences with UCN (Disparity Estimation)



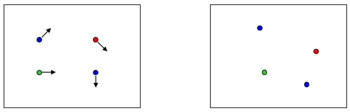
C. Choy, J.Y. Gwak, S. Savarese, M. Chandraker, Universal Correspondence Network, NIPS'16

Slide credit: Christopher Choy

Topics of This Lecture

- Recap: CNNs for Video Analysis
- Matching and correspondence estimation
  - Metric learning
  - Spatial Transformer Networks
  - Correspondence networks
- Optical Flow Estimation
  - FlowNet
  - FlowNet2

Recap: Estimating Optical Flow

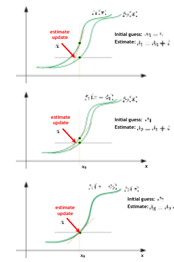


$I(x,y,t-1)$ $I(x,y,t)$

- Optical Flow
  - Given two subsequent frames, estimate the apparent motion field $u(x,y)$ and $v(x,y)$ between them.
- Key assumptions
  - Brightness constancy: projection of the same point looks the same in every frame.
  - Small motion: points do not move very far.
  - Spatial coherence: points move like their neighbors.

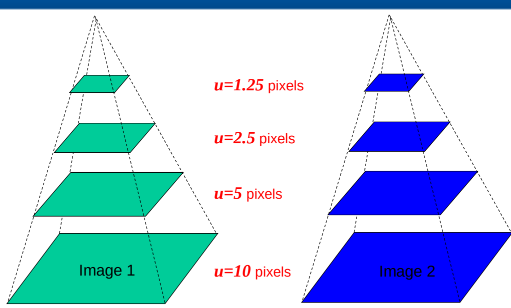
Recap: Iterative LK Refinement

- Estimate velocity at each pixel using one iteration of LK estimation.
- Warp one image toward the other using the estimated flow field.
- Refine estimate by repeating the process.



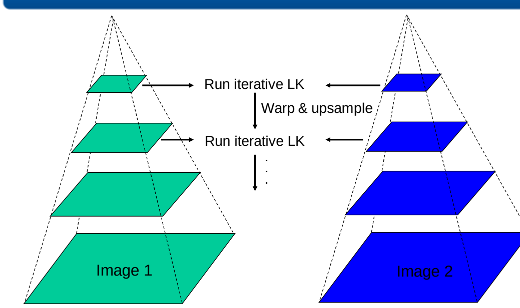
- Iterative procedure
  - Results in subpixel accurate localization.
  - Converges for small displacements.

Recap: Coarse-to-fine Optical Flow Estimation



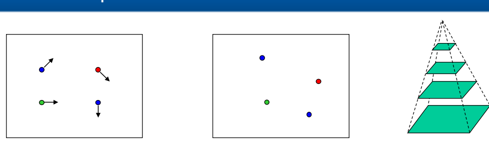
Gaussian pyramid of image 1 Gaussian pyramid of image 2

Recap: Coarse-to-fine Optical Flow Estimation



Gaussian pyramid of image 1 Gaussian pyramid of image 2

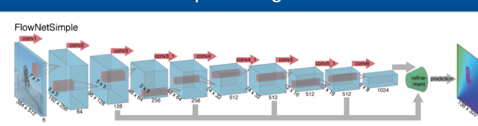
CNNs for Optical Flow Estimation



$I(x,y,t-1)$ $I(x,y,t)$

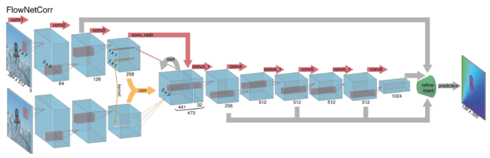
- How can we achieve this with Deep Networks?
  - Intuition: need to match local image patches
  - CNNs can capture local context, so spatial smoothing should not be necessary
  - But iterative and coarse-to-fine estimation may be necessary.

FlowNet: FlowNetSimple Design



- Simple initial design
  - Simply stack two sequential images together and feed them through the network
  - In order to compute flow, the network has to compare image patches
  - But it has to figure out on its own how to do that…

FlowNet: FlowNetCorr Design

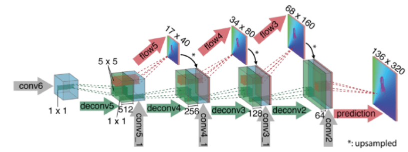


- Correlation network
  - Central idea: compute a correlation score between two feature maps

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

  - Then refine the correlation scores and turn them into flow predictions

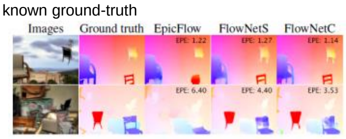
FlowNet



- Flow refinement stage (both network designs)
  - After series of conv and pooling layers, the resolution has been reduced
  - Refine the coarse pooled representation by upconvolution layers (unpooling + upconvolution)
  - Skip connections to preserve high-res information from early layers

FlowNet: Training

- Training on FlyingChairs dataset
  - Synthetic dataset with known ground-truth
- Example prediction
  - Both networks can capture fine details

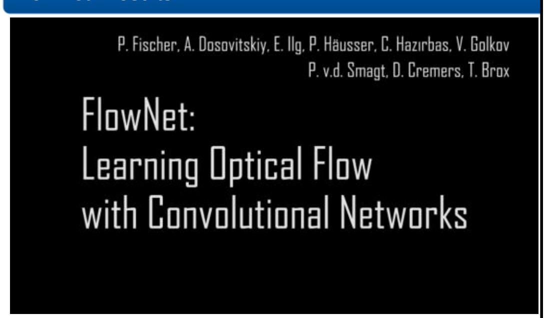


FlowNet: Comparing the two designs

Method	Sintel Clean train	Sintel Clean test	Sintel Final train	Sintel Final test	KITTI train	KITTI test	Middlebury train AEE	Middlebury train AAE	Middlebury test AEE	Middlebury test AAE	Chairs test	Time (sec) CPU	Time (sec) 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

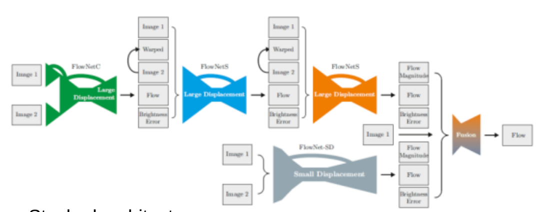
- Comparison (avg endpoint errors)
  - Both FlowNetS and FlowNetC can effectively learn to estimate flow
  - FlowNetC overfits to the training data slightly more
  - Finetuning (+ft) and variational refinement (+v) improve results further
  - Performance close to pre-CNN methods, but much faster to compute

FlowNet: Results



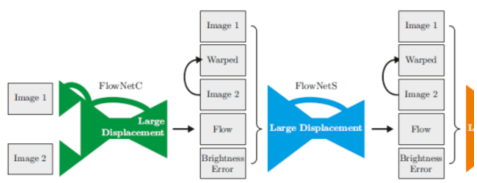
P. Fischer et al., FlowNet: Learning Optical Flow with Convolutional Networks, ICCV 2015.

FlowNet 2.0: Improved Design



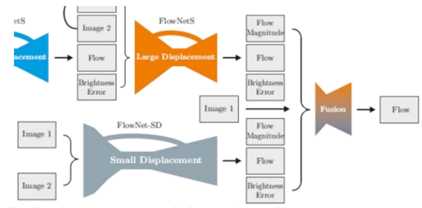
- Stacked architecture
  - Several instances of FlowNetC and FlowNetS stacked together to estimate large-displacement flow
  - Sub-network specialized on small motions
  - Fusion layer

FlowNet 2.0: Detailed View



- Stacked FlowNets
  - Estimates large motion in a coarse-to-fine approach
  - Second image is warped at each level with the intermediate optical flow
  - Intermediate flow and (warped brightness) error are concatenated
  - ⇒ Difficulty of the learning task is reduced at each level

FlowNet 2.0: Detailed View



- Small Displacement Module and Fusion
  - For small displacements, FlowNet2-CSS is not accurate
  - Separate FlowNet2-SD module replaces 5×5 and 7×7 by multiple 3×3 kernels and assumes a stride 1 instead of stride 2 at the first layer
  - Small and simple network to fuse the outputs

FlowNet 2.0: Comparison

	Method	Sintel *clean* AEE		Sintel *final* AEE		KITTI 2012 AEE		KITTI 2015			Middlebury AEE		Runtime ms per frame	
		train	*test*	*train*	*test*	*train*	*test*	AEE *train*	Fl-all *train*	Fl-all *test*	*train*	*test*	CPU	GPU
Accurate	EpicFlow† [22]	2.27	4.12	3.56	6.29	**3.09**	**3.8**	9.27	27.18%	**27.10%**	0.31	0.39	42,600	–
	DeepFlow† [32]	2.66	5.38	3.57	7.21	4.48	5.8	10.63	26.52%	29.18%	**0.25**	0.42	51,940	–
	FlowFields [2]	**1.86**	**3.75**	**3.06**	**5.81**	3.33	3.5	**8.33**	**24.43%**	–	0.27	**0.33**	22,810	–
	LDOF (CPU) [7]	4.64	7.56	5.96	9.12	10.94	12.4	18.19	38.11%	–	0.44	0.56	64,900	–
	LDOF (GPU) [27]	4.76	–	6.32	–	10.43	–	18.20	38.05%	–	0.36	–	–	6,270
	PCA-Layers [33]	3.22	5.73	4.52	7.89	5.99	5.2	12.74	27.26%	–	0.66	–	**3,300**	–
Fast	EPPM [4]	–	**6.49**	–	**8.38**	–	9.2	–	–	–	–	**0.33**	–	200
	PCA-Flow [33]	**4.04**	6.83	**5.18**	8.65	**5.48**	**6.2**	**14.01**	**39.59%**	–	**0.70**	–	140	–
	DIS-Fast [16]	5.61	9.35	6.31	10.13	11.01	14.4	21.20	53.73%	–	0.92	–	70	–
	FlowNetS [11]	4.50	6.96‡	5.45	7.52‡	8.26	–	–	–	–	1.09	–	–	**18**
	FlowNetC [11]	4.31	6.85‡	5.87	8.51‡	9.35	–	–	–	–	1.15	–	–	32
FlowNet 2.0	FlowNet2-s	4.55	–	5.21	–	8.89	–	16.42	56.81%	–	1.27	–	–	**7**
	FlowNet2-ss	3.22	–	3.85	–	5.45	–	12.84	41.03%	–	0.68	–	–	14
	FlowNet2-css	2.51	–	3.54	–	4.49	–	11.01	35.19%	–	0.54	–	–	31
	FlowNet2-css-ft-sd	2.50	–	3.50	–	4.71	–	11.18	34.10%	–	0.43	–	–	31
	FlowNet2-CSS	2.10	–	3.23	–	**3.55**	–	**8.94**	29.77%	–	0.44	–	–	69
	FlowNet2-CSS-ft-sd	2.08	–	3.17	–	4.05	–	10.07	30.73%	–	0.38	–	–	69
	FlowNet2	**2.02**	**3.96**	**3.14**	6.02	4.09	–	10.06	30.37%	–	**0.35**	**0.52**	–	123
	FlowNet2-ft-sintel	(1.45)	4.16	(2.01)	**5.74**	3.61	–	9.84	**28.20%**	–	0.35	–	–	123
	FlowNet2-ft-kitti	3.43	–	4.66	–	(1.28)	**1.8**	(2.30)	(8.61%)	**11.48%**	0.56	–	–	123

- Comparison (avg endpoint errors)
  - Similar accuracy as best pre-CNN methods (but much faster)

References and Further Reading

- RNNs
  - R. Pascanu, T. Mikolov, Y. Bengio, On the difficulty of training recurrent neural networks, JMLR, Vol. 28, 2013.
  - A. Karpathy, The Unreasonable Effectiveness of Recurrent Neural Networks, blog post, May 2015.
- LSTM
  - S. Hochreiter , J. Schmidhuber, Long short-term memory, Neural Computation, Vol. 9(8): 1735–1780, 1997.
  - A. Graves, Generating Sequences With Recurrent Neural Networks, ArXiV 1308.0850v5, 2014.
  - C. Olah, Understanding LSTM Networks, blog post, August 2015.

References and Further Reading

- Optical Flow
  - P. Fischer, A. Dosovitskiy, E. Ilg, P. Haeusser, C. Hazirbas, V. Golkov, P. v.d. Smagd, D. Cremers, T. Brox, FlowNet: Learning Optical Flow with Convolutional Networks, ICCV 2015.
  - E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, T. Brox, FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks, CVPR 2017.
  - A. Ranjan, M.J. Black, Optical Flow Estimation using a Spatial Pyramid Network, CVPR 2017.