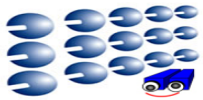
Deep Learning-based Approaches for Stereo Reconstruction

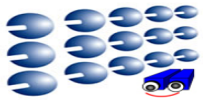
Vlad-Cristian Miclea, Sergiu Nedevschi

Technical University of Cluj-Napoca

September 7, 2017



Contents



Introduction

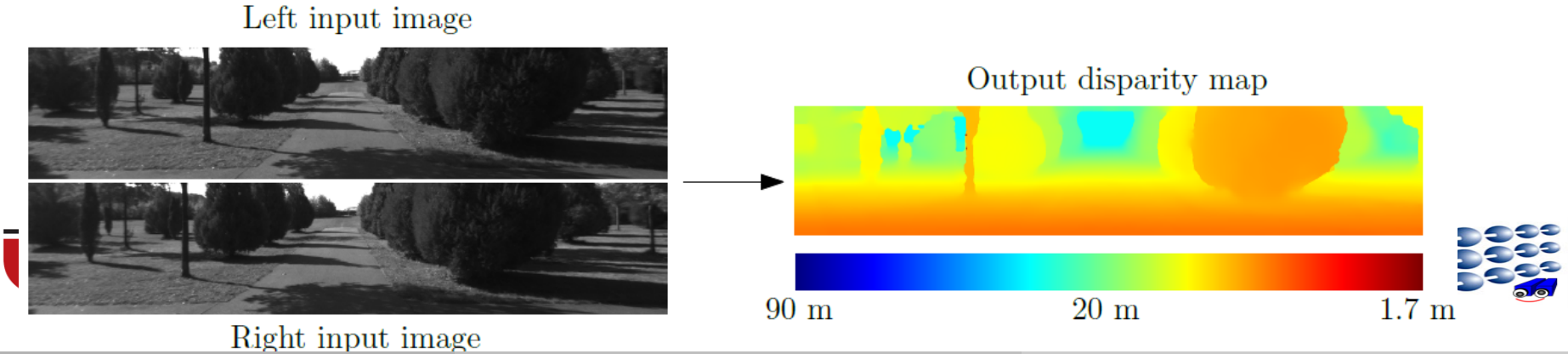
Stereo matching

- 2 simultaneously taken images
- Search corresponding pixel only on scanlines (rectification)

$$y' = y$$
$$x' = x + D(x, y)$$
$$z(x, y) = \frac{f \times B}{D(x, y)}$$

Classic Stereo Taxonomy:

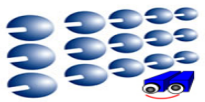
1. Cost computation
2. Cost aggregation
3. Selection/Optimization
4. Disparity refinement



Cost computation – Patched-based methods

Intensity-based methods

- Build a 3D cost volume of $h * w * d_{max}$
- Sum of Absolute Diff $C_{SAD}(p,d) = \sum_{q \in N_p} |I_L(q) - I_R(q-d)|$
- Sum of Squared Diff $C_{SSD}(p,d) = \sum_{q \in N_p} [I_L(q)^2 - I_R(q-d)]^2$
- Normalized Cross Correl. $C_{NCC}(p,d) = \frac{\sum_{q \in N_p} I_L(q) * I_R(q-d)}{\sqrt{\sum_{q \in N_p} [I_L(q)^2] * \sum_{q \in N_p} [I_R(q-d)^2]}}$



Cost computation – Patched-based methods

Intensity-based methods

- Build a 3D cost volume of $h * w * d_{max}$
- Sum of Absolute Diff $C_{SAD}(p,d) = \sum_{q \in N_p} |I_L(q) - I_R(q-d)|$
- Sum of Squared Diff $C_{SSD}(p,d) = \sum_{q \in N_p} [I_L(q)^2 - I_R(q-d)]^2$
- Normalized Cross Correl. $C_{NCC}(p,d) = \frac{\sum_{q \in N_p} I_L(q) * I_R(q-d)}{\sqrt{\sum_{q \in N_p} [I_L(q)^2] * \sum_{q \in N_p} [I_R(q-d)^2]}}$

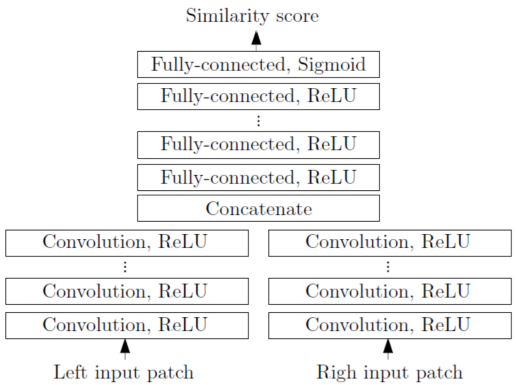
Non-parametric methods – invariance

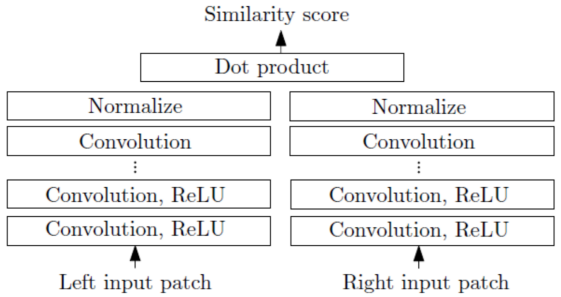
- ZSAD, ZSSD, ZNCC (subtract mean)
- Matrix rank-transform
- Mutual Information
- Census Transform (CT) - revisited below

Cost computation – Learning-based methods (ConvNets)

MC-CNN [Zbontar and LeCun 2015]

- ConvNets – really good at extracting features
- Can use directly to predict depth from single image
- Siamese network – compare feature to feature
- Trained on patches, with stereo ground truth
  - positive pairs sampled directly from disparity maps
  - negative pairs sampled with moderate perturbation
- 2 architectures - Accurate (67s) and Fast (0.5s)



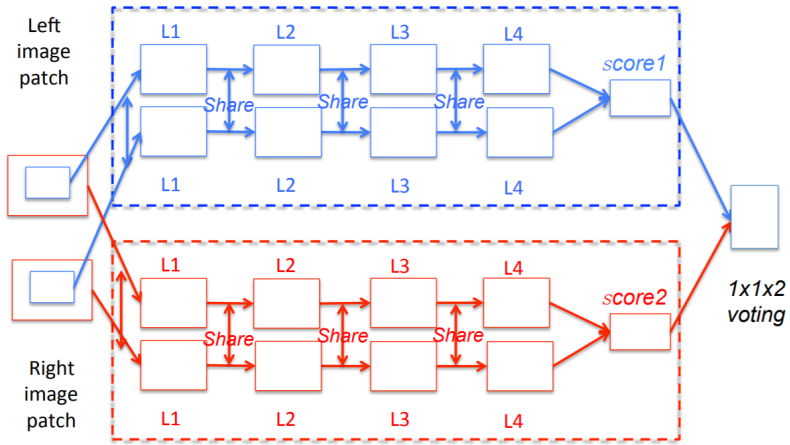


Cost computation – Learning-based methods (ConvNets)

MC-CNN [Zbontar and LeCun 2015]

DeepEmbed [Chen+ 2015]

- Also siamese network
- Combine computation at two scales (full and half resolution)
- Improved computation time



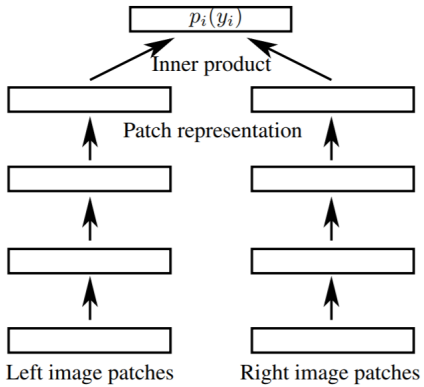
Cost computation – Learning-based methods (ConvNets)

MC-CNN [Zbontar and LeCun 2015]

DeepEmbed [Chen+ 2015]

Content-CNN [Luo+ 2016]

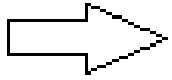
- Smaller siamese network
- Multi-class classification loss (not only binary)
- Larger receptive field – better



Theory – Census Transform

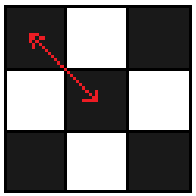
- Step function to encapsulate info

32	25	25
33	63	55
22	1	99

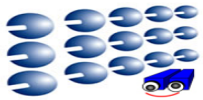


$$\mathbf{11110111}0$$

$$\xi(p_1, p_2) = \begin{cases} 0 & : p_1 \leq p_2 \\ 1 & : p_1 > p_2 \end{cases}$$



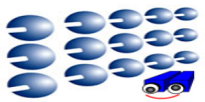
$$CT(p_1) = \otimes_{p_2 \in N(p_1)} \xi(p_1, p_2)$$



Theory – Census Transform

- Step function to encapsulate info
- Final cost - Hamming distance

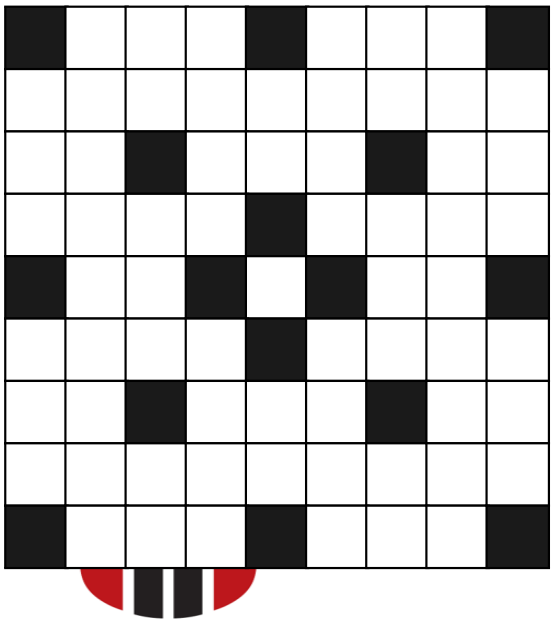
$$C(p_1) = \sum_{CensusMask(i)=1} \oplus(CT_l(p_1(i)), CT_r(p_1'(i)))$$

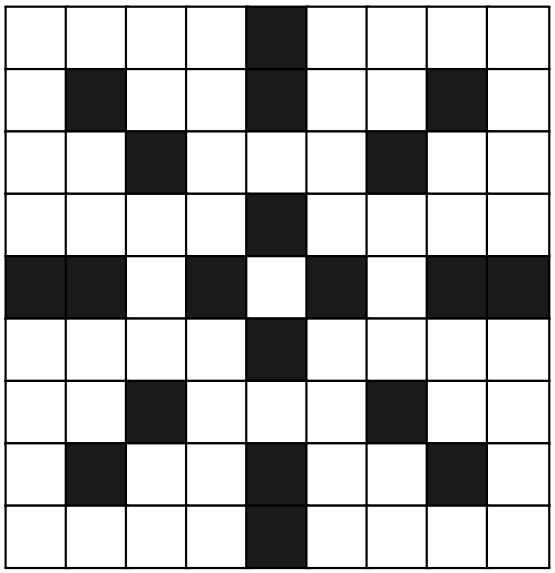


Optimal pixels for Census

Idea 1

- Can use all neighbor pixels (Dense) - High cost, few information
- Better - sparse masks
- Better - optimally select pixels (somehow similar to extracting features, but still patched-based)





Optimal pixels for Census

Idea 1

Idea 2

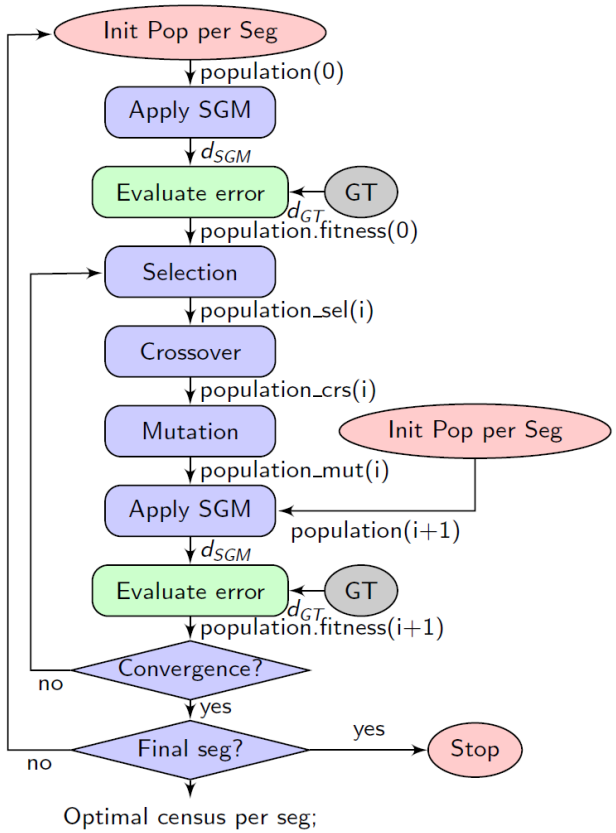
- Semantically segment the image using a fast ConvNet [ERFNet 2017] $\approx$ 20 ms
- Use fewer classes: road, sidewalk, vegetation, building, large vertical objects, small vertical objects
- Find an optimal Census mask for each particular segment – stochastic optimization

Optimal pixels for Census

Idea 1

Idea 2

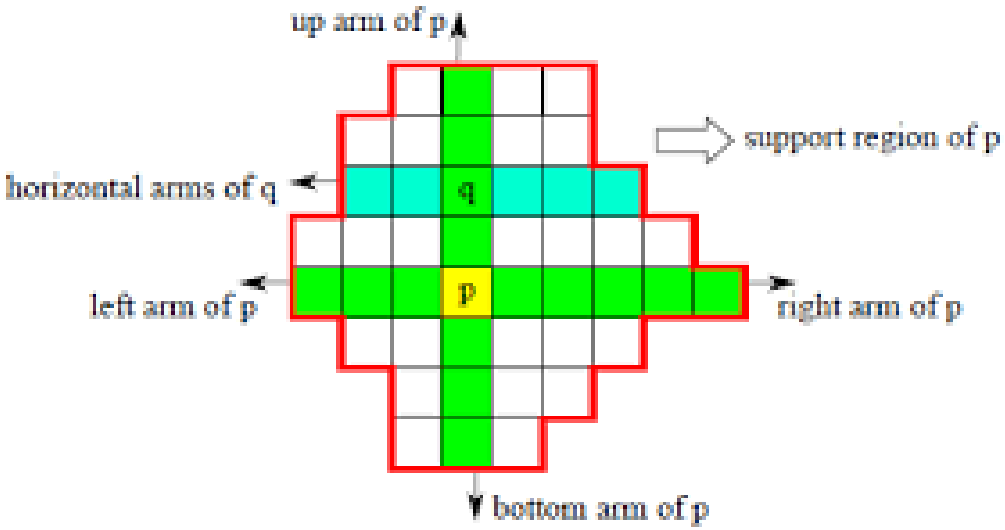
- Semantically segment the image using a fast ConvNet [ERFNet 2017] $\approx$ 20 ms
- Use fewer classes: road, sidewalk, vegetation, building, large vertical objects, small vertical objects
- Find an optimal Census mask for each particular segment – stochastic optimization
- Genetic algorithm to generate this



Cost aggregation

Classic aggregation methods

- Increase reliability by using more information – aggregate through cost volume
- [Fife+ 2013] – Aggregation window should not be larger than 7x7
- [AdCensus] – introduce Cross-based Cost Aggregation (CBCA)
- Length of the arms – parameterized by Intensity and Location

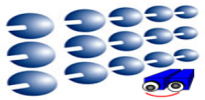


Cost aggregation

Classic aggregation methods

CBCA with Semantic Segments

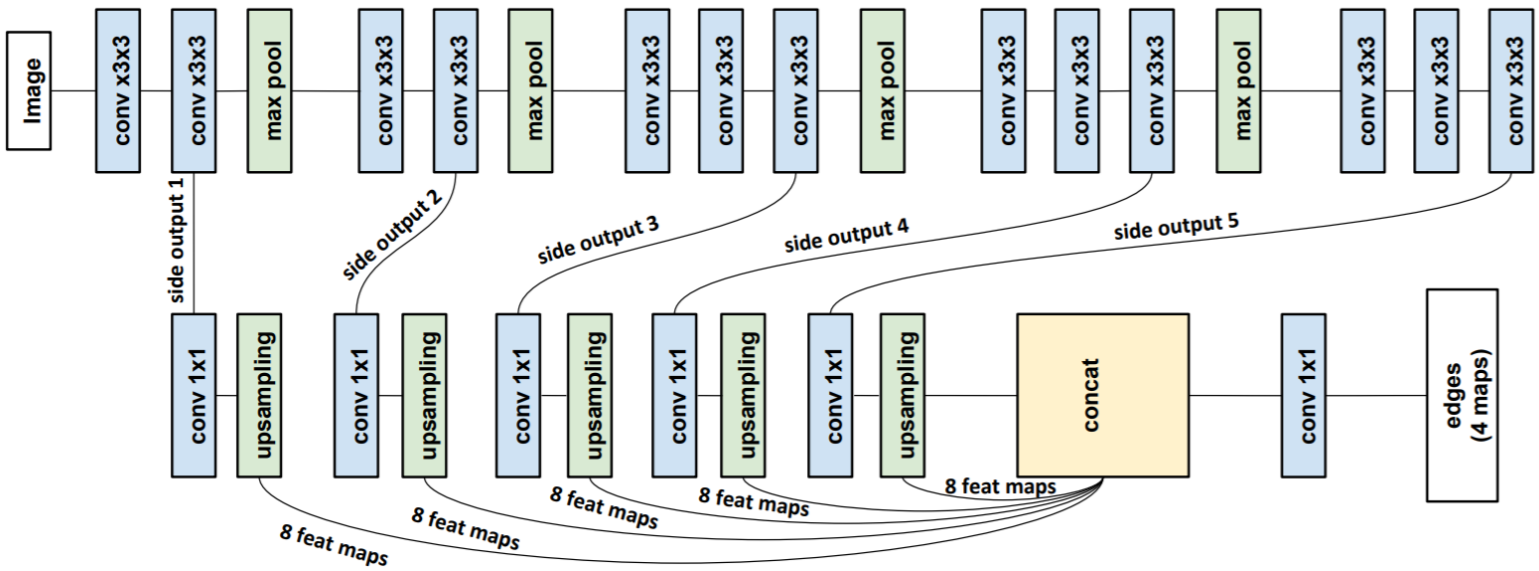
- Introduce a new parameter in CBCA – not to extend outside a segment
- Can also introduce genetic algorithms to optimize the aggregation window



Cost aggregation

Learning-based aggregation [Kuzmin+ 2016]

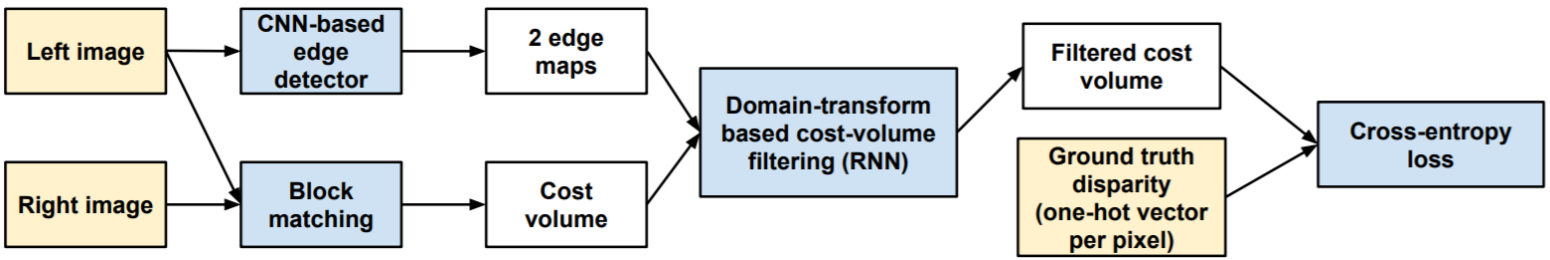
- SAD and Census for cost computation
- Residual network to detect edges (object boundary detection)
- Combines edges detected at different image scales in a final map



Cost aggregation

Learning-based aggregation [Kuzmin+ 2016]

- SAD and Census for cost computation
- Residual network to detect edges (object boundary detection)
- Combines edges detected at different image scales in a final map
- Aggregation scheme – as with semantic segmentation
- Also perform a cost-volume filtering
- Can compare with a GT (one-hot vector) – directly obtain disparity



Do we have enough information?

- Photometric variations

Do we have enough information?

- Photometric variations
- Foreshortening

Do we have enough information?

- Photometric variations
- Foreshortening
- Reflections

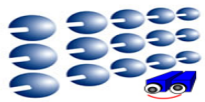
Do we have enough information?

- Photometric variations
- Foreshortening
- Reflections
- Transparent surfaces
- Texture-less areas
- Repetitive patterns
- Complex occlusions

Disparity Optimization

Global optimization – Energy minimization

- Incorporate a regularization term $E(D) = E_{Data}(D) + E_{Smooth}(D)$
- Data – matching costs; Smoothness – priors (adjacent pixels - similar)
- Minimize Energy Function
  $E(D) = \sum_{p \in I} C_p(d_p) + \sum_{(p,q) \in N} V_{p,q}(d_p, d_q)$
- Hard to solve, but can be approximated
  - Graph Cuts [Kolmogorov+ 2002], Belief Propagation[Sun+ 2003]



Disparity Optimization

Global optimization – Energy minimization

- Minimize Energy Function
  $E(D) = \sum_{p \in I} C_p(d_p) + \sum_{(p,q) \in N} V_{p,q}(d_p, d_q)$

Semi-Global Matching

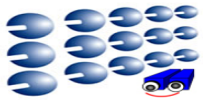
$$L_r(p,d) = C_p(d) + min(L_r(p-r,d), L_r(p-r,d-1) + P_1,$$
$$L_r(p-r,d+1) + P_1, min_{i \neq i \pm 1} L_r(p-r,i) + P_2)$$

- Multiple 1D energy optimizations 8/16 dir.) to approximate 2D
- Penalize for small/large disparity changes – P_1 and P_2
- Final cost, sum from all directions
- WTA – argmin at each position in volume

Global optimization – SGM

Optimal penalties

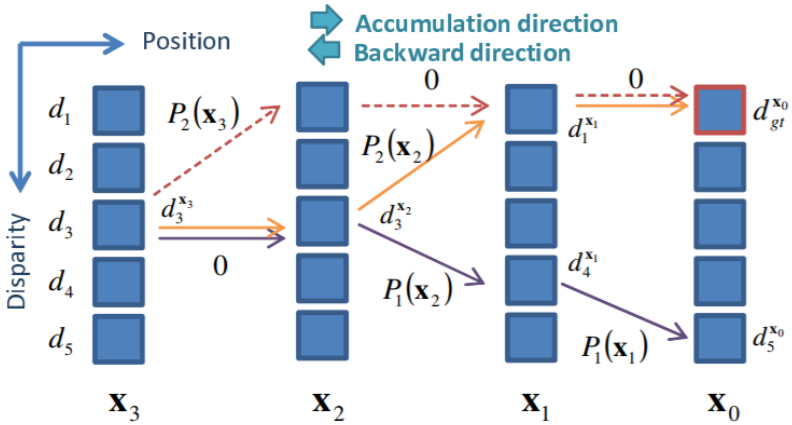
- Penalties – key to good regularization
- P_1 - empirically chosen
- P_2 - dependent on intensity changes
- Using Segments – Optimal P_1 per segment (using GA)

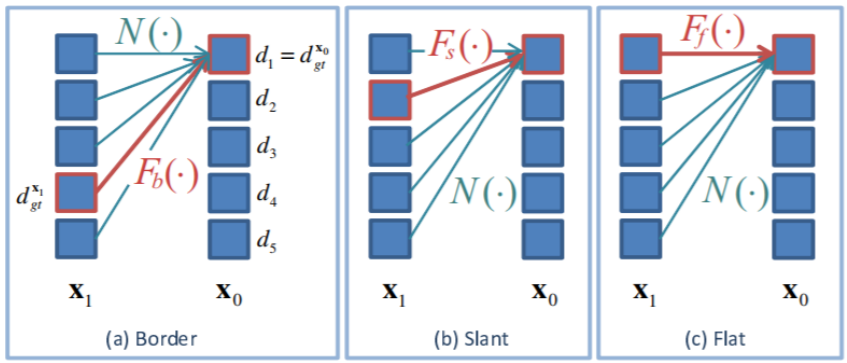


Global optimization – SGM

SGM-Net [Seki+ 2017]

- Costs obtained using MC-CNN
- Training:
  - Path cost to generate an optimal path
  - Uses a Hinge loss – add $L_r(p, d)$ on a path – forward/backward prop. depending only on penalties
  - Neighbor cost to ensure correct penalty sequence (3 cases)
- Testing: P_1 and P_2 provided for each pixel





Disparity Refinement

Pixel level

- Left-Right consistency check
- Median filter
- Edge-preserving filters – Bilateral filter, Guided filter, Multilateral filter
- Also can benefit from segmentation
- Also can benefit from edge-detecting ConvNet

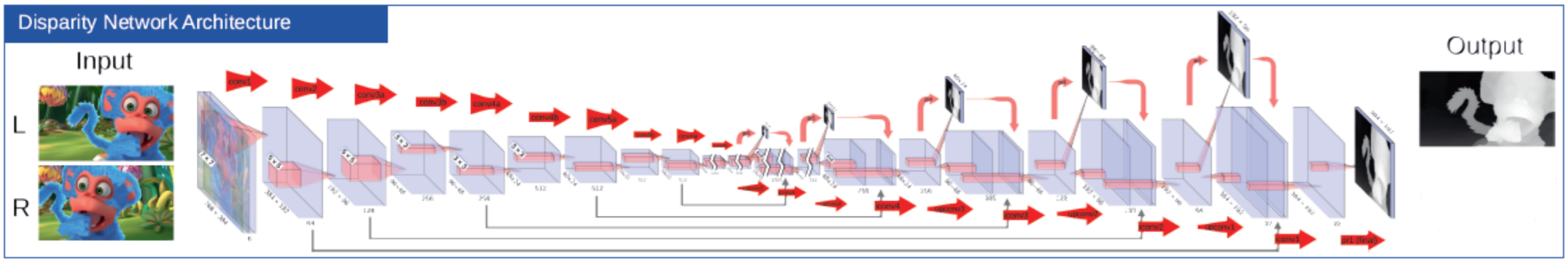
Sub-Pixel level

- Large distances – need accuracy
- Optimal interpolation function – $f(c_{d-1}, c_d, c_{d+1})$
- Parabola, Symmetric V, SinFit [Haller 2011], LUTs [IV 2016]
- Optimal function trained for each segment (using GA) [IV 2017]

Other Stereo ConvNets

DispNetC [Mayer+ 2016]

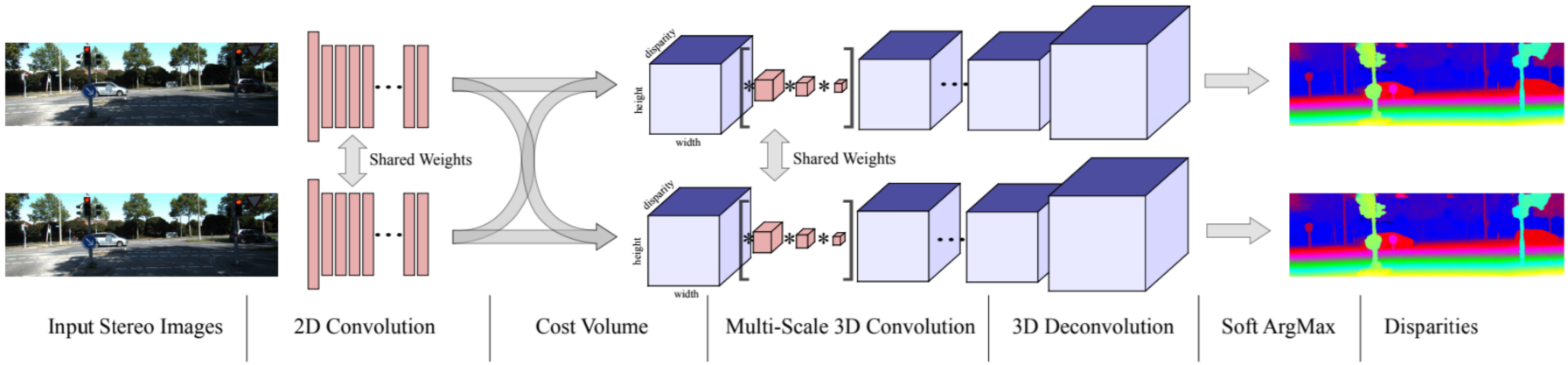
- New dataset for training (fine tuning on Kitti)
- Stack left and right information to get more features
- Convolution/Deconvolution-like architecture
- Very good computation time



Other Stereo ConvNets

GC-Net [Kendall+ 2017]

- 2D convolution for matching cost (feature-based as MC-CNN)
- Extensive use of 3D convolutions (encorporate lots of info) – capture context
- New method for argmin (differentiable soft-argmin) – assign probabilities to costs in volume

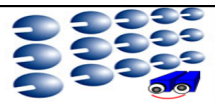


Evaluation

Metrics

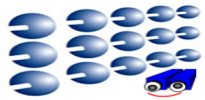
- Real images – Kitti benchmark: sparse GT, traffic scenarios
- Synthetic images – DispNetC and Middlebury: Dense GT
- Metric: percent of correct matches (with a threshold T=3 or T=5)

Method	D1-all	Density	Runtime	Environment
GC-NET	**2.87 %**	100.00 %	0.9 s	Nvidia GTX Titan X
SGM-Net	3.66 %	100.00 %	67 s	Titan X
MC-CNN-acrt	3.89 %	100.00 %	67 s	Nvidia GTX Titan X (CUDA, Torch)
DispNetC	4.34 %	100.00 %	0.06 s	Nvidia GTX Titan X (Caffe)
Content-CNN	4.54 %	100.00 %	1 s	Nvidia GTX Titan X
Deep Embed	4.24 %	100.00 %	3 s	1 core @ 2.5 Ghz (C/C++)
DeepCostAggr	6.37 %	99.98 %	**0.03 s**	GPU @ 2.5 Ghz (C/C++)
SS-OptCen	6.34 %	100.00 %	**0.03 s**	Nvidia GTX 1080 (CUDA + Torch)


Table: Results on Kitti2015 dataset for presented approaches


Conclusions

- Stereo problem (geometric) – now mainly using learning
- Lots of ConvNet-based approaches for each step in pipeline
- New End-to-End methods – train to directly obtain disparity map
- Can get close to real-time
- Can benefit from ConvNets in other domains (edge detection, semantic segmentation) to increase performance



Acknowledgement and some references

Acknowledgement

The work was supported by the EU H2020 project UP-Drive under grant no. 688652.

Some References

[IV 2017]: Vlad-Cristian Miclea, Sergiu Nedevschi, Semantic segmentation-based stereo reconstruction with statistically improved long range accuracy, IV 2017;
[ERFNet 2017]: E. Romera, J. M. Alvarez, L. M. Bergasa and R. Arroyo, ERFNet: Efficient Residual Factorized ConvNet for Real-time Semantic Segmentation, T-ITS 2017;
[Zbontar and LeCun 2015]: Jure Zbontar and Yann, LeCun, Computing the stereo matching cost with a convolutional neural network, CVPR 2015 [SGM 2017]: Akihito Seki and Marc Pollefeys, SGM-Nets: Semi-Global Matching With Neural Networks, CVPR 2017;
[Kendall+ 2017]: A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy, A. Bachrach and A. Bry: End-to-End Learning of Geometry and Context for Deep Stereo Regression, ICCV 2017;

Thank you for your attention!