Optical flow based Pixel Distribution Learning on Motion Segmentation

Youwei Chen, Zhiyuan Lu, Chuqing Fu

CMPUT 414 group

Team 3 Galaxy 3C-321

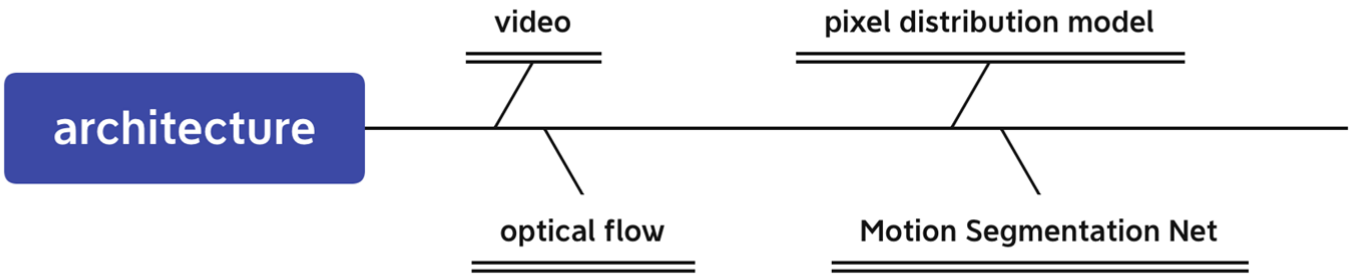
Motion Segmentation

Motion Segmentation is the task of **identifying** the independently **moving objects (pixels)** in the video and separating them from the background motion.

Introduction

Idea:

Pixel distribution model (random feature selection) **to the extracted optical flow**. The pixel distribution also serves as an input to the **motion segmentation net**.



Aim: track the motion of vehicles across frames

What is Optical Flow?

Optical flow is the motion of objects between consecutive frames of sequence, caused by the relative movement between the object and camera.

Motion/Optical flow vectors

How to compute?

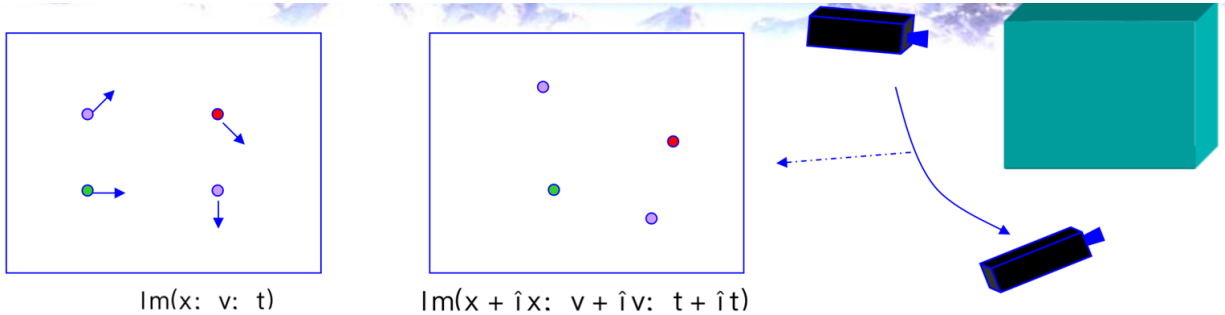
Solve pixel correspondence problem
given a pixel in Im1, look for same pixels in Im2

Possible assumptions

color constancy: perceived color of objects remains relatively constant under varying illumination conditions.

For grayscale images, this is **brightness constancy**

small motion: points do not move very far
This is called the **optical flow problem**



$$\mathrm{Im}(x+\delta x, y+\delta y, t+\delta t) = \mathrm{Im}(x,y,t) + \frac{\partial \mathrm{Im}}{\partial x}\delta x + \frac{\partial \mathrm{Im}}{\partial y}\delta y + \frac{\partial \mathrm{Im}}{\partial t}\delta t + h.o.t.$$

Keep linear terms

- Use constancy assumption and rewrite:

$$0 = \underbrace{\frac{\partial \mathrm{Im}}{\partial x}}_{\text{unknown}}\delta x + \underbrace{\frac{\partial \mathrm{Im}}{\partial y}}_{\text{unknown}}\delta y + \underbrace{\frac{\partial \mathrm{Im}}{\partial t}}_{\text{known}}\delta t$$

- Notice: Linear constraint, but no unique solution

Taylor expansion of intensity variation

Solving for optic flow

Rewrite as dot product

$$-\frac{\partial \mathrm{Im}}{\partial t}\delta t = \left(\frac{\partial \mathrm{Im}}{\partial x}, \frac{\partial \mathrm{Im}}{\partial y}\right) \cdot \begin{pmatrix} \delta x \\ \delta y \end{pmatrix} = \nabla \mathrm{Im} \cdot \begin{pmatrix} \delta x \\ \delta y \end{pmatrix}$$

- Each pixel gives one equation in two unknowns:

$$\boldsymbol{k} = \boldsymbol{n} * \boldsymbol{f}$$

Image spatial gradient normal $\boldsymbol{n}$: $\nabla \mathcal{I}\mathrm{m}$, later: M

The image motion / optic flow $\boldsymbol{f} = (\delta x \psi \delta y)^{\mathrm{T}}$, later u

Image temporal gradient k: $\partial \mathrm{Im}/\partial t$, later dIm

- Typically solve for motion in 2x2, 4x4, 8x8 or larger image patches.
- Over determined equation system:

$$\begin{pmatrix} \vdots \\ -\frac{\partial \mathrm{Im}}{\partial t} \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots & \vdots \\ \frac{\partial \mathrm{Im}}{\partial x} & \frac{\partial \mathrm{Im}}{\partial y} \\ \vdots & \vdots \end{pmatrix} \begin{pmatrix} \delta x \\ \delta y \end{pmatrix}$$

$$\mathrm{dIm} = \mathrm{M} * \mathrm{u}$$

- Can be solved in least squares sense using Matlab

$$\mathrm{u} = \mathrm{M} \backslash \mathrm{dIm}$$

- Can also be expressed using normal equations:

$$(\mathrm{M}^{\mathrm{T}}\mathrm{M})\mathrm{u} = \mathrm{M}^{\mathrm{T}}\mathrm{dIm}$$

Solve for optic flow using several simultaneous equations

Traditional Algo. Lucas-Kanade

1. Suppose ATA is easily invertible
2. Suppose there is not much noise in the image
3. assumptions are not violated

Errors?

SSD/Lucas-Kanade tracking algorithm

1. Estimate velocity at each pixel by solving Lucas-Kanade equations
2. Warp H towards I using the estimated flow field

- use image warping techniques

1. Repeat until convergence

Iterative Refinement

Optical Flow by CNNs

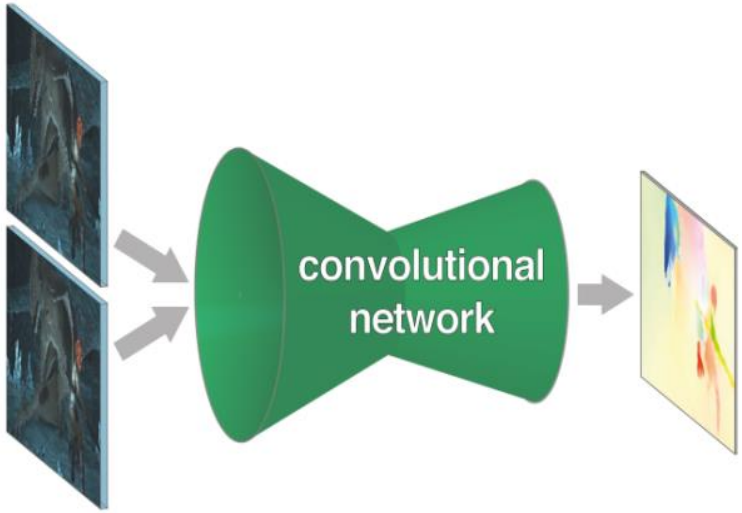
Optical flow estimation requires the network to learn feature representations and match them at different locations in two images.

- The network needs to extract both spatial and temporal information.

Try to reduce the following limitations in the traditional algorithms:

- Brightness/Intensity consistency
- Small motion

FlowNet-General Structure



1. compress the motion features in the contractive part.

2. refine the motion features in the expanding part.

FlowNetS and FlownetC

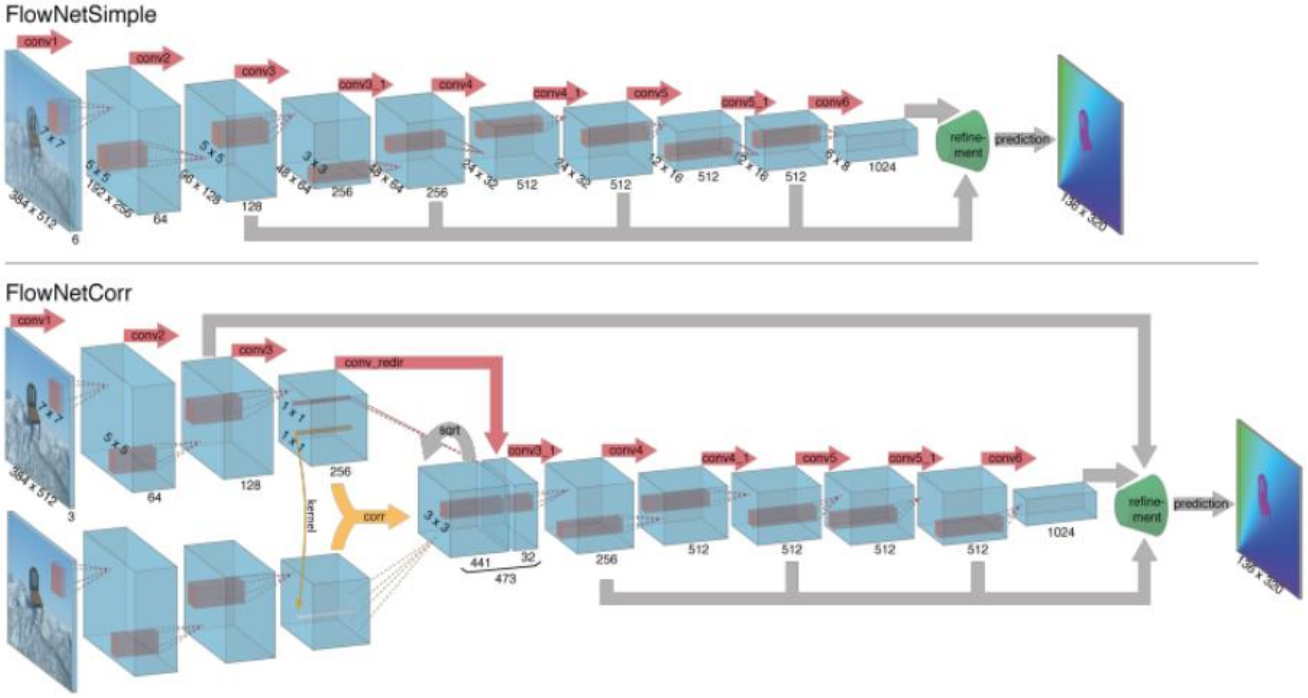


Figure 2. The two network architectures: FlowNetSimple (top) and FlowNetCorr (bottom).

Correlation and Refinement

- Correlation layer: To perform multiplicative patch comparisons between two extracted feature maps

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

- Refinement layer: To increase the image resolution after series of convolutional layers and pooling

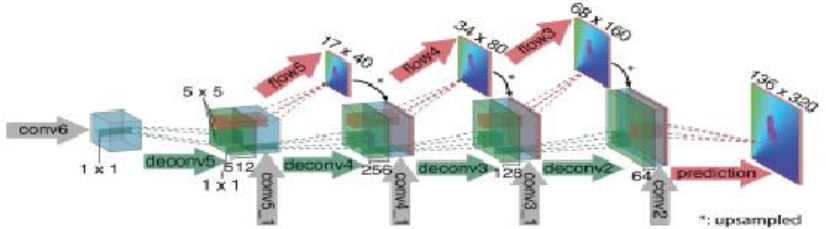
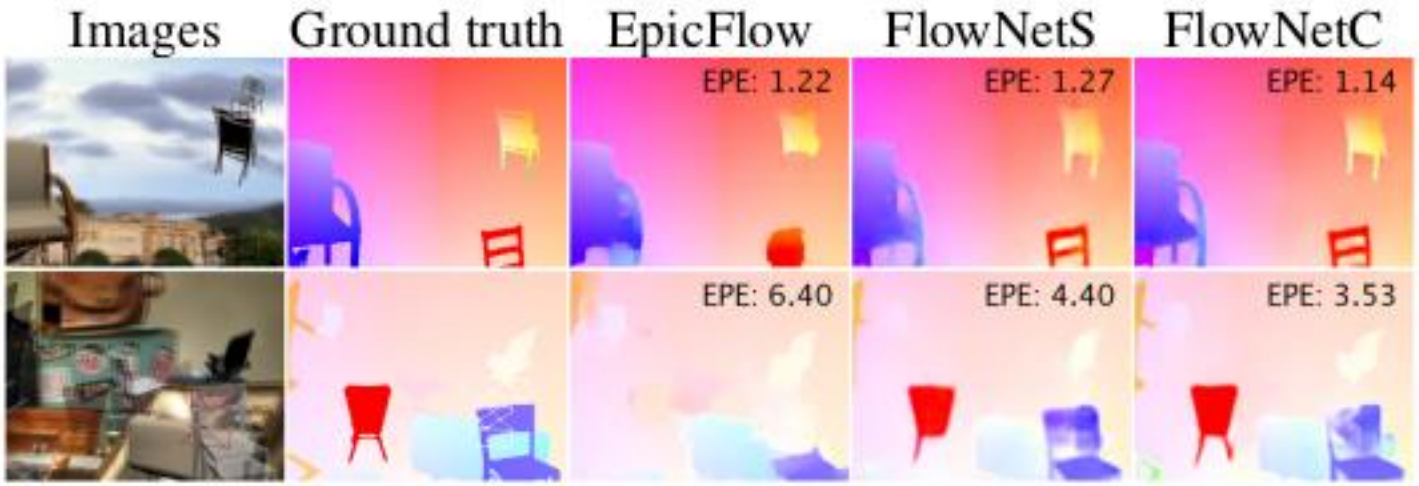


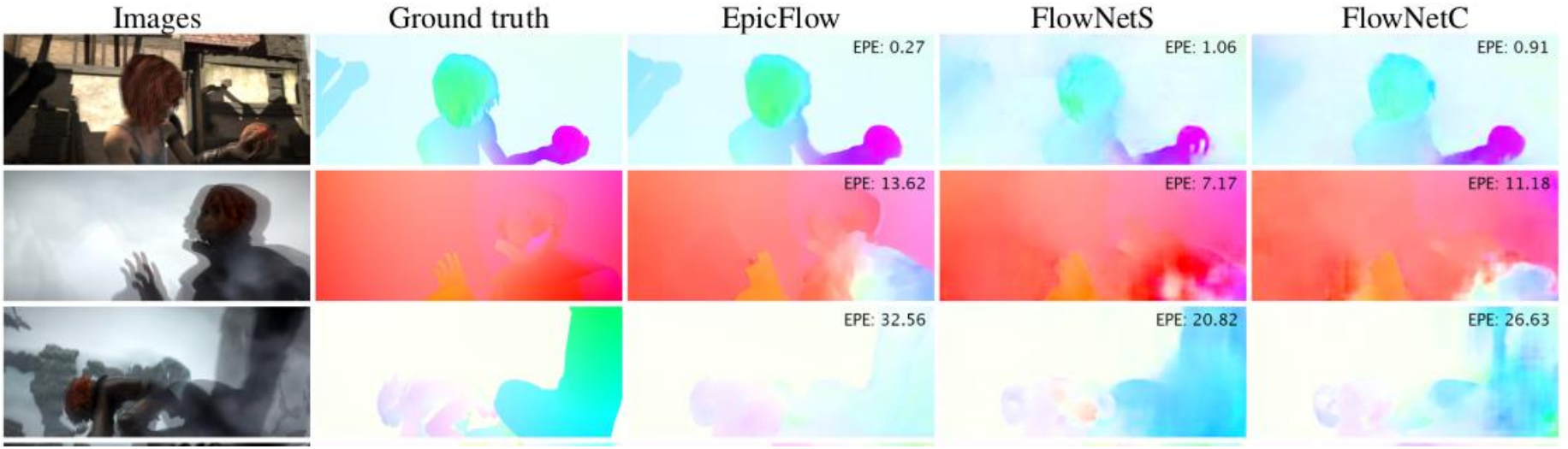
Figure 3. Refinement of the coarse feature maps to the high resolution prediction.

FlowNet Performance-Flying Chair



Flying Chair dataset:
Synthetic, consists of 22, 872 frame pairs

FlowNet Performance-Sintel



Sintel Dataset: Synthetic animation scene, consists of 1041 frame pairs
Some of the frames have motion blur.

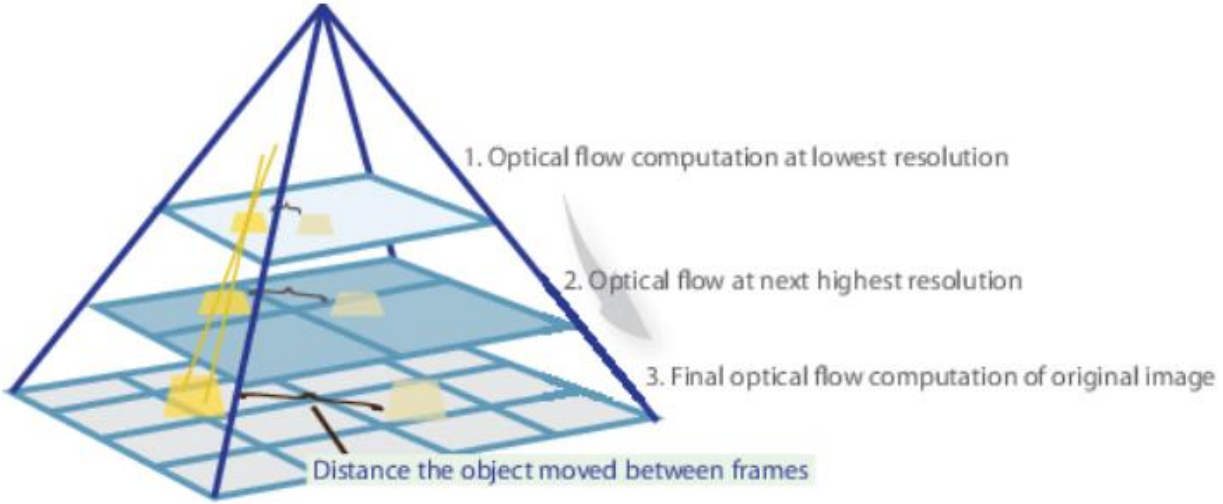
- FlowNetC slightly more overfits to the training data.
- FlowNetC have more problem with large displacement.

In FlowNet 2.0 paper, Ilg el al. says:

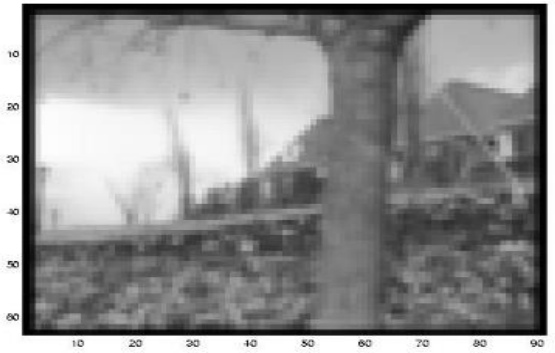
***"FlowNetC outperforms FlowNetS**. The result we got with FlowNetS and S short corresponds to the one reported in Dosovitskiy et al. [11]. However, we obtained much better results on FlowNetC. We conclude that Dosovitskiy et al. [11] **did not train FlowNetS and FlowNetC under the exact same conditions**. When done so, the FlowNetC architecture compares favorably to the FlowNetS architecture."*

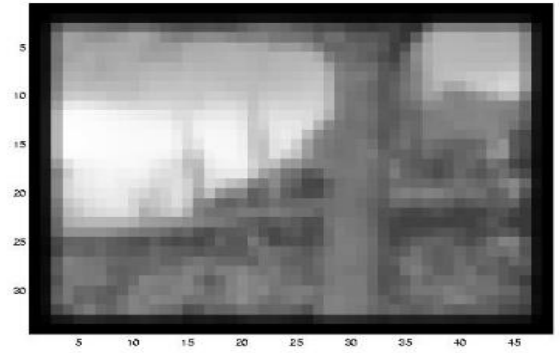
Are there any other methods to solve the large motion problem?

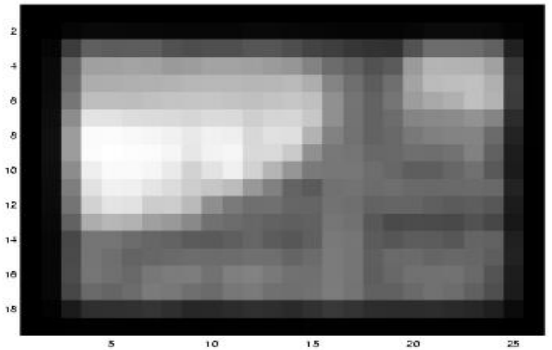
spatial pyramid algorithm->reduce the image resolution



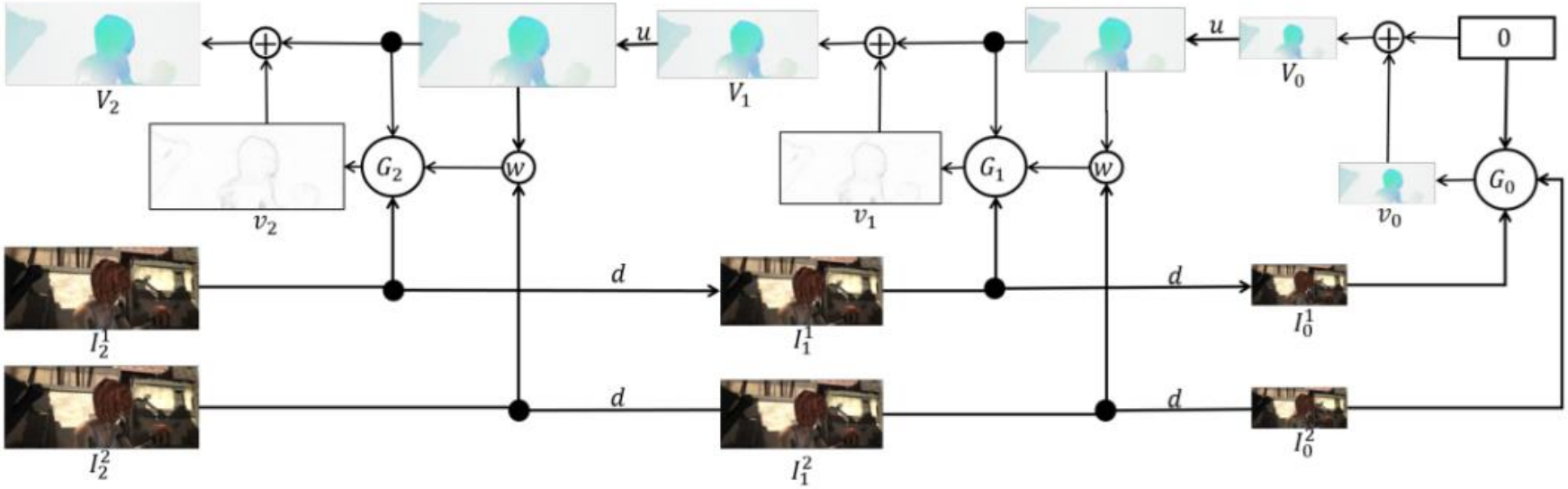

20
40
60
80
100
120
20
40
60
80
100
120
140
160


10
20
30
40
50
60
10
20
30
40
50
60
70
80
90


5
10
15
20
25
30
5
10
15
20
25
30
35
40
45

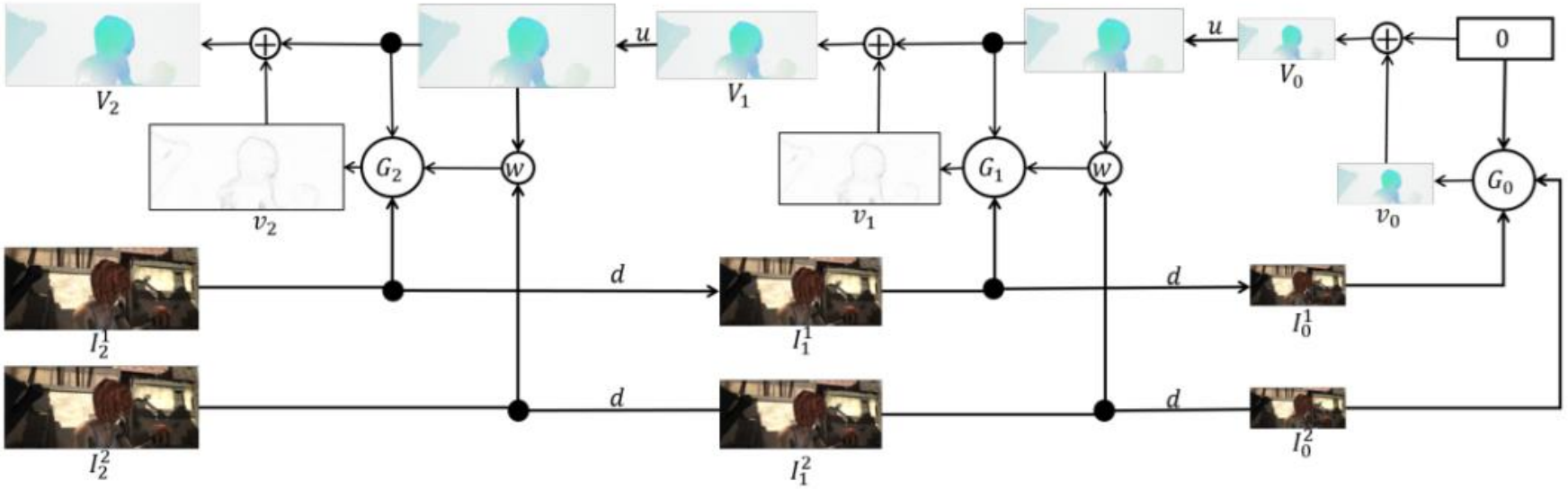

2
4
6
8
10
12
14
16
18
5
10
15
20
25

SPyNet



d(.) downsampling function, pyramidal down image (w, h) -> (w/2, h/2)
u(,) upsampling function
w(I, V): warping function, warp the image I according to the flow field V.

Compute the Flow Field V



Let {G0, ..., Gk} be the set of trained convnet models, k denotes the level of pyramid.

$$v_k = G_k(I_k^1, w(I_k^2, u(V_{k-1})), u(V_{k-1}))$$

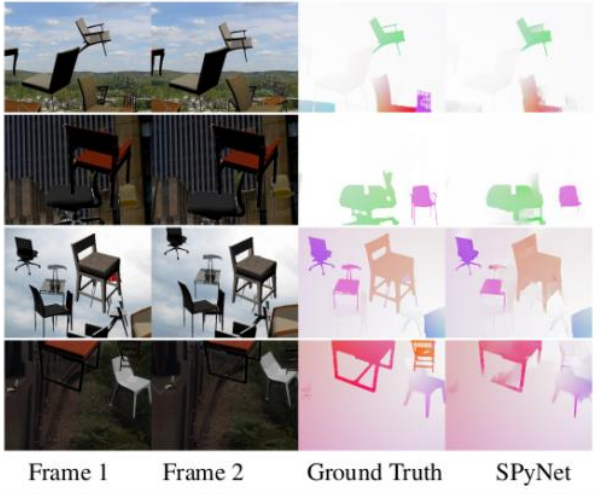
$$V_k = u(V_{k-1}) + v_k.$$

SPyNet Performance

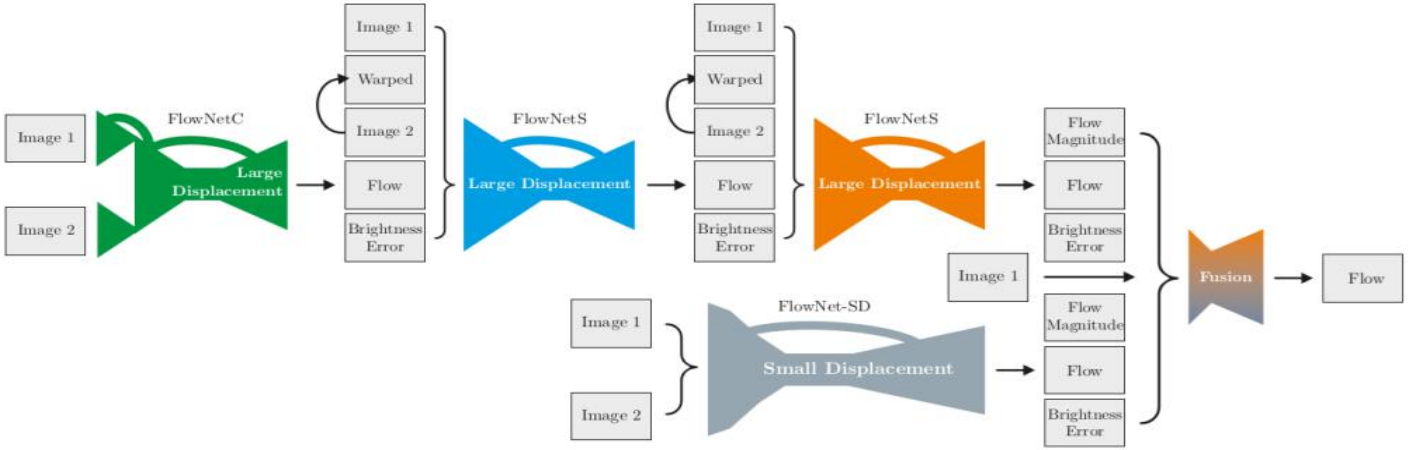
- The model size is 96% smaller than FlowNet.
- Achieve an accuracy comparable to FlowNet, surpassing it in several benchmarks.

Since the SPyNet combines the traditional methods with deep learning, it also inherits some limitations in traditional methods:

ie. Large motions in small objects are difficult to capture when pyramidal down the images.

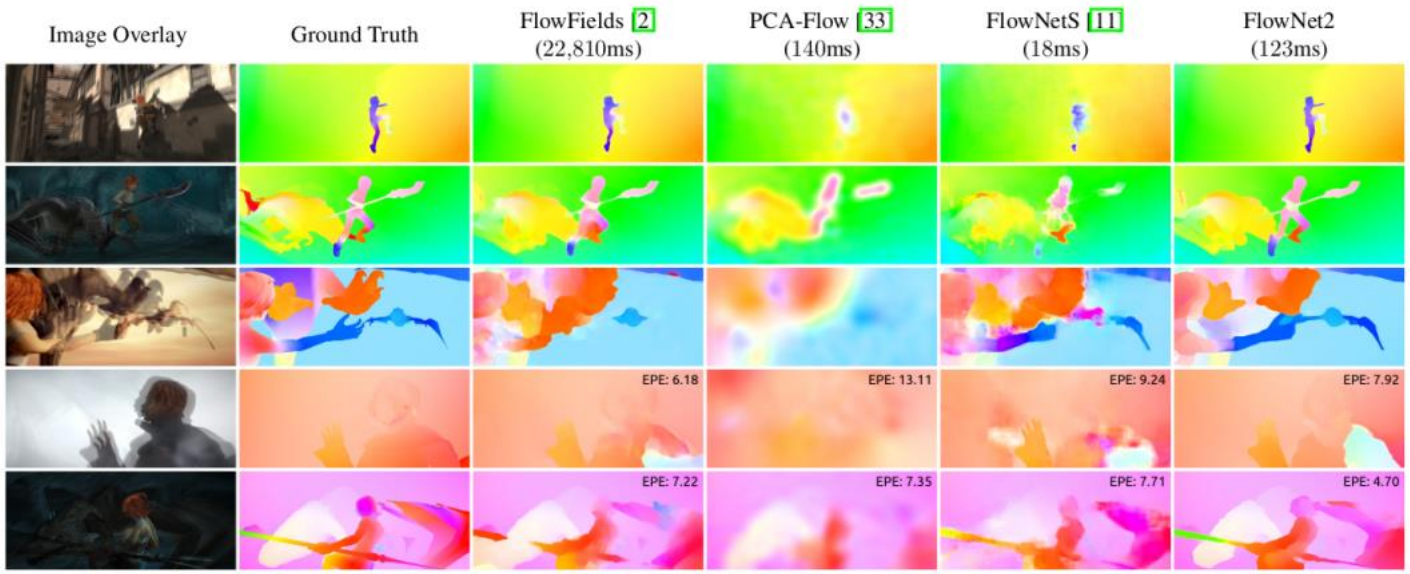


FlowNet 2.0



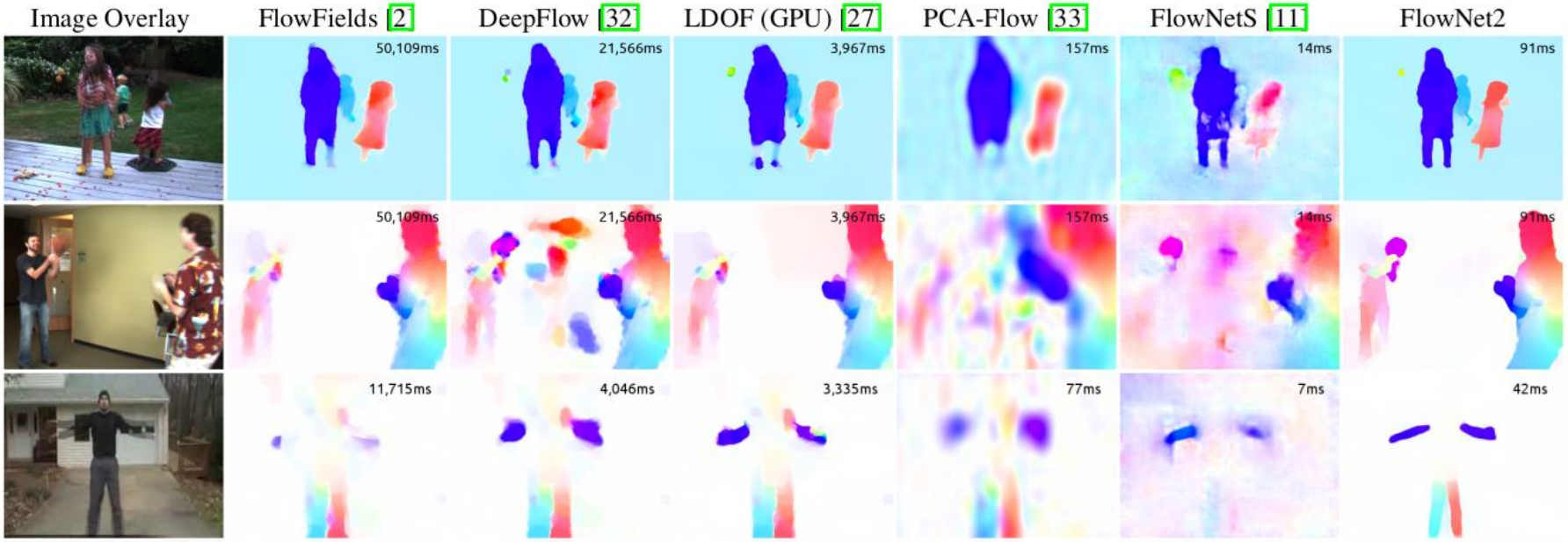
1. stack I1, I2, pass the stacked images to FlowNet, get the flow estimate field wi = <ui, vi>.T
2. warp I2 according to the flow field wi. I2,i(x, y) = I2(x+u, y+v)
3. compute the error ei = ||I2,i - I1||
4, pass I2,i and ei to the next network

FlowNet 2.0 Performance



1. Achieves better accuracy than FlowNetS, FlowNetC, and SPyNet
2. But the model size is 2x greater than that of the FlowNet. (over 160M parameters)

FlowNet 2: Performance on Motion Seg.



Traditional Method(LK) vs. CNN

Traditional Algorithm:
Does not rely on training datasets. It is more generic
Efficient to compute
A lot of assumptions are applied.

CNN:
Generally more accurate than traditional methods.
Relies on training datasets. Overfitting.

Pixel Distribution Learning

What is Pixel Distribution Learning?

Learning a distribution of randomly permuted temporal/spatial pixels to perform segmentation.

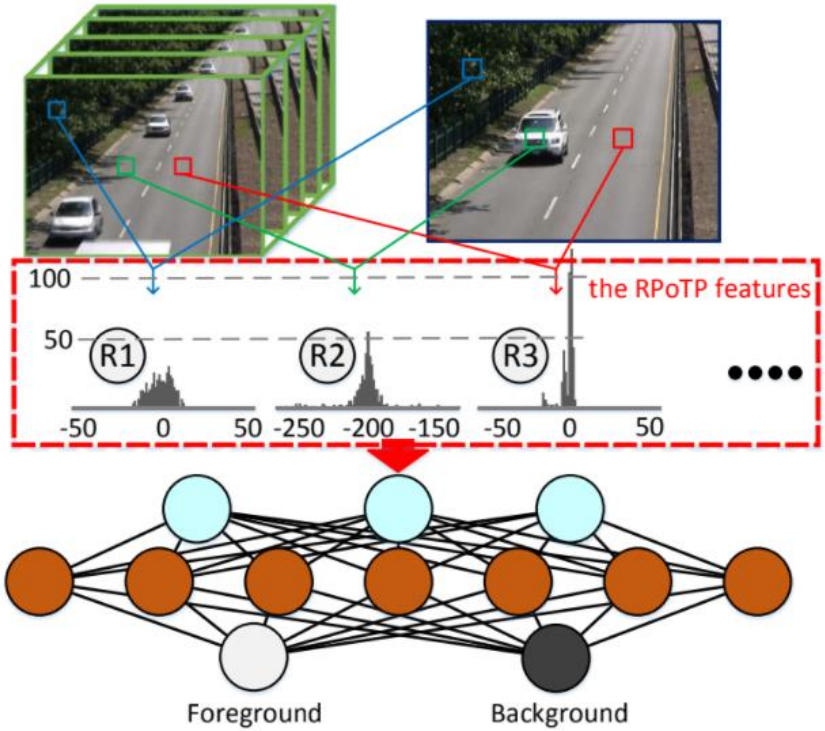
Pixel Distribution Learning

Why distribution?

- More generalized model, less data needed

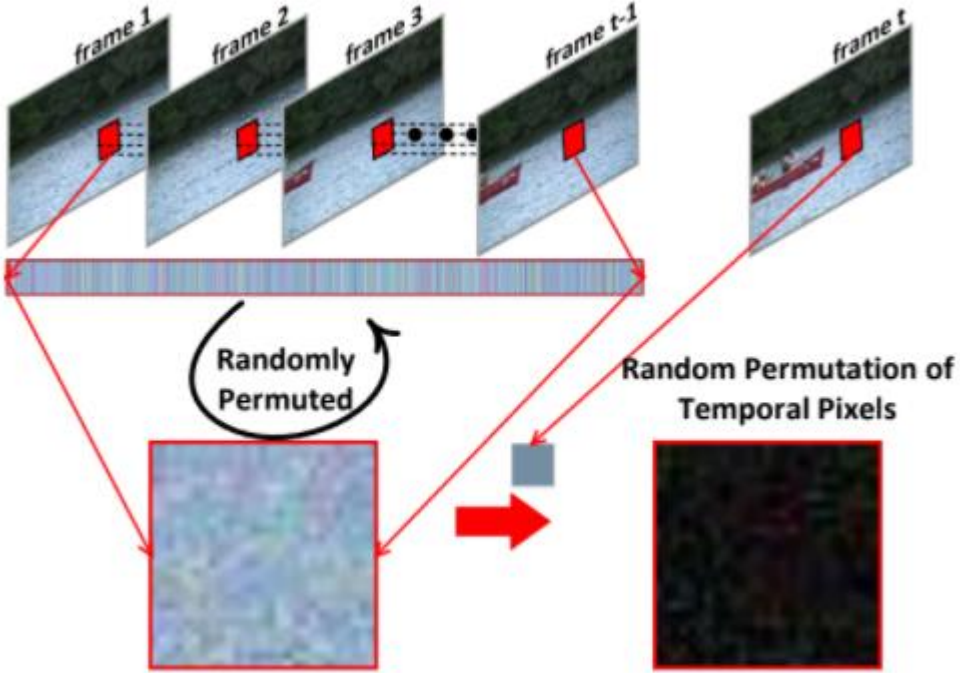
Temporal - Background Subtraction of Videos

- Random Permutation of Temporal Pixels
- Deep Pixel Distribution Learning

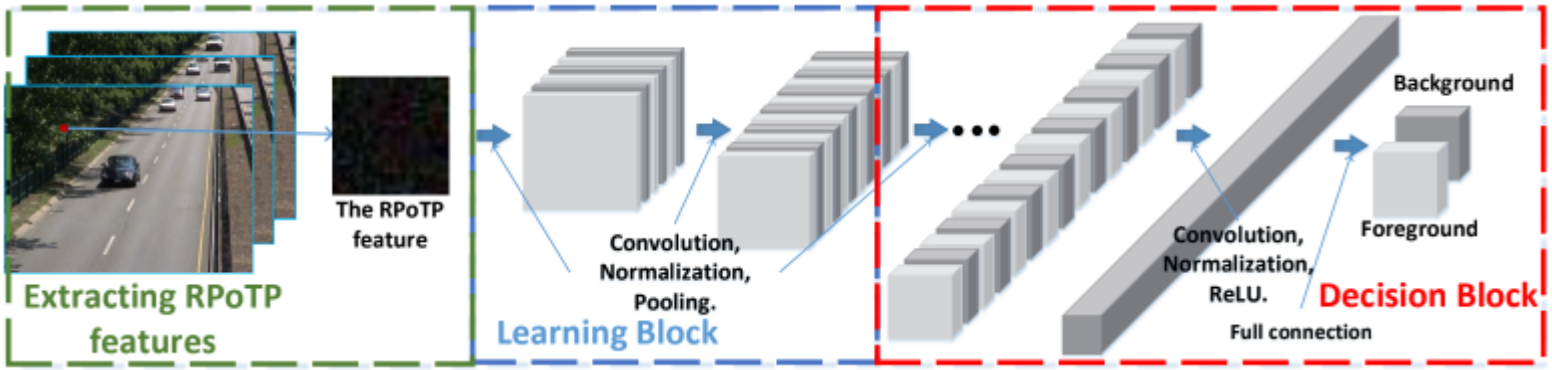


[10]

Temporal - Background Subtraction of Videos



Temporal - Background Subtraction of Videos



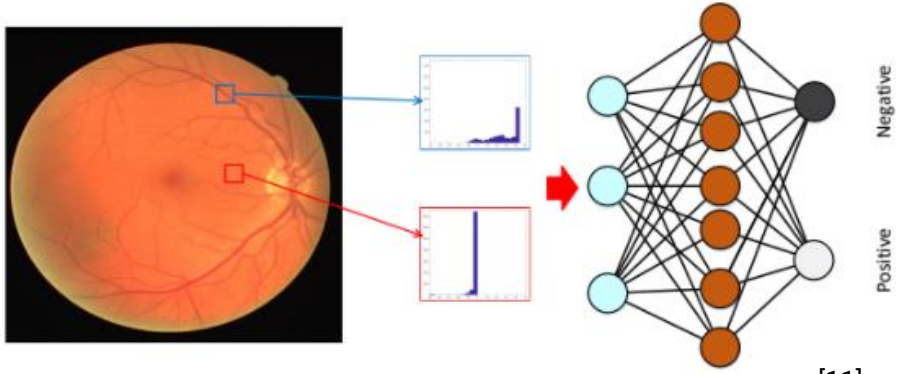
Network Architecture

Temporal - Background Subtraction of Videos

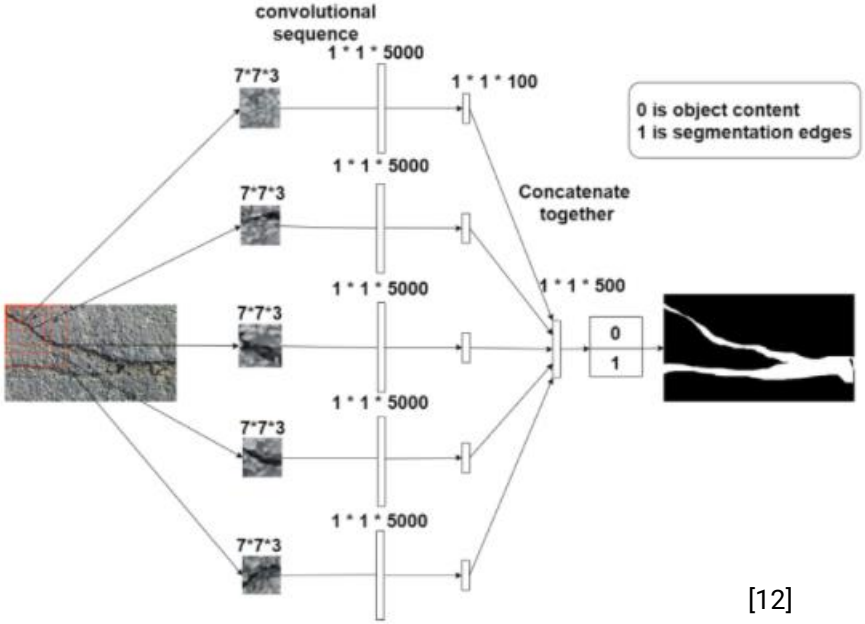
Why random?

- Avoid spurious temporal correlations
  Unless the dataset is extremely large

Spatial - Image Segmentation



[11]

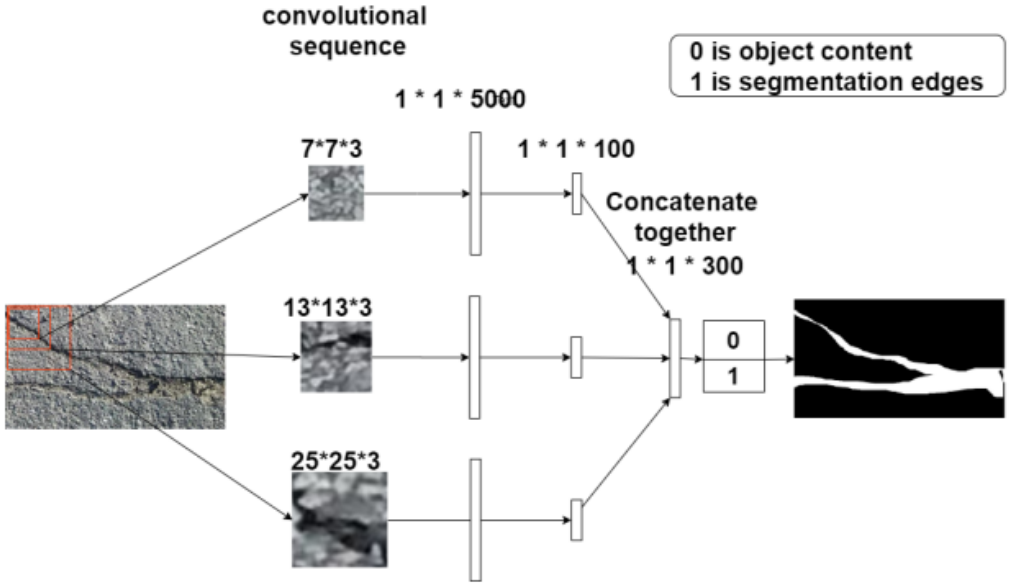


[12]

- Random Permutation of Spatial Pixels
- Deep Pixel Distribution Learning

Spatial - Image Segmentation

- Multi-Scale
- Down-sampling to the same size



Temporal - Background Subtraction of Videos

Why random?

- Avoid over-fitting

Pixel Distribution Learning

Cons?

- Noises
- time-consuming

References

[1]https://www.sciencedirect.com/topics/engineering/motion-segmentation
[2]https://paperswithcode.com/dataset/fbms
[3]https://lmb.informatik.uni-freiburg.de/resources/datasets/
[4]http://www.cvlibs.net/datasets/kitti/index.php
[5]http://ugweb.cs.ualberta.ca/~vis/courses/CompVis/
[6]https://nanonets.com/blog/optical-flow/
[7] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazırbas¸, V. Golkov,¨P. van der Smagt, D. Cremers, and T. Brox, “FlowNet: Learning optical
flow with convolutional networks,” ICCV, pp. 2758–2766, 2015.
[8] A. Ranjan and M. J. Black, “Optical flow estimation using a spatial pyramid network,” CVPR, pp. 4161–4170, 2017.
[9] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, “FlowNet2.0: Evolution of optical flow estimation with deep networks,” CVPR, pp. 2462–2470, 2017.
[10] C. Zhao, T. Cham, X. Ren, J. Cai, and H. Zhu, “Background subtraction based on deep pixel distribution learning,” in IEEE Int. Conf. Multimedia and Expo (ICME), July 2018, pp. 1–6.
[11] Xuanyi Wu, Jianfei Ma,Yu Sun, Chenqiu Zhao, Anup Basu (2021).Multi-Scale Deep Pixel Distribution Learning for Concrete Crack Detection 2020 International Conference on Pattern Recognition (ICPR).
[12] C. Zhao and A. Basu, "Pixel Distribution Learning for Vessel Segmentation under Multiple Scales," 2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC), 2021, pp. 2717-2721, doi: 10.1109/EMBC46164.2021.9629614.

Thanks for your time. Any questions?