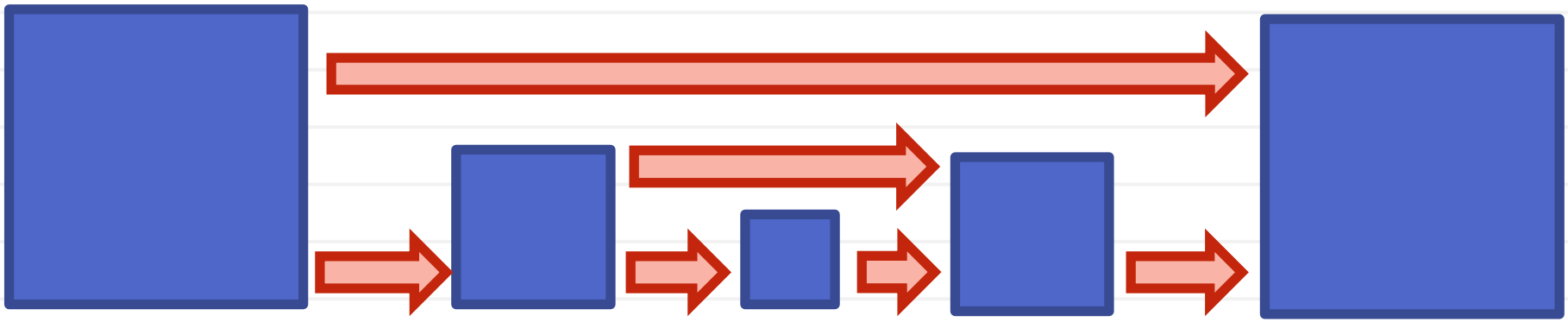
Lecture 6: Generative ConvNets

Recap: semantic segmentation

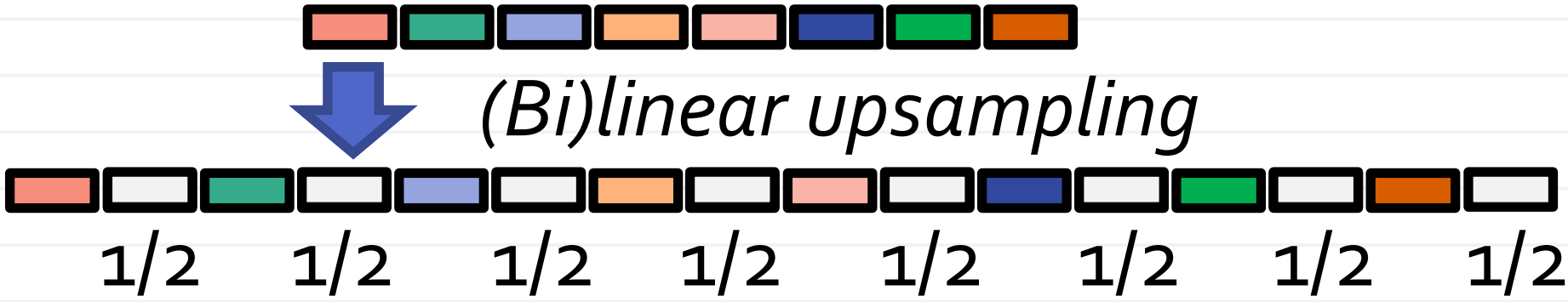
These architectures look approximately like:



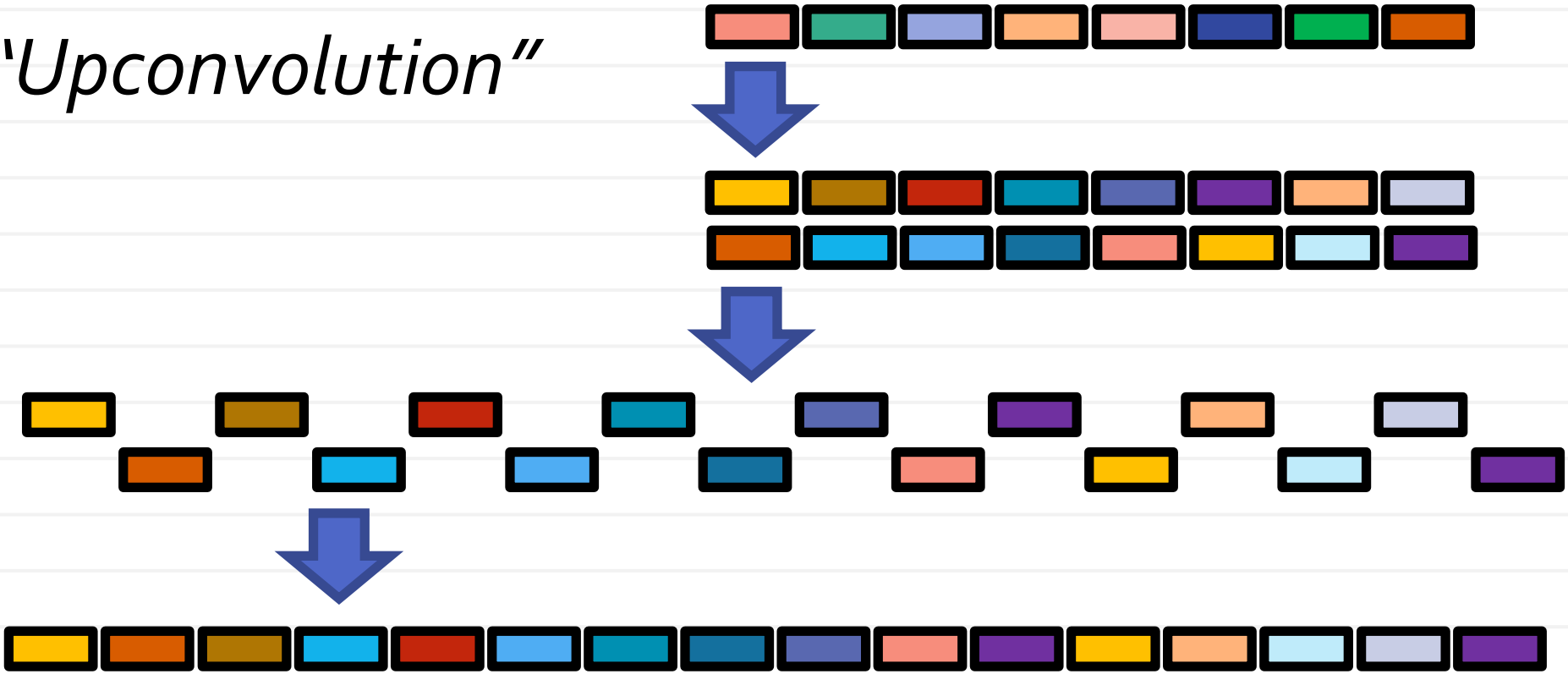
RGB image

Pixel-wise probabilities

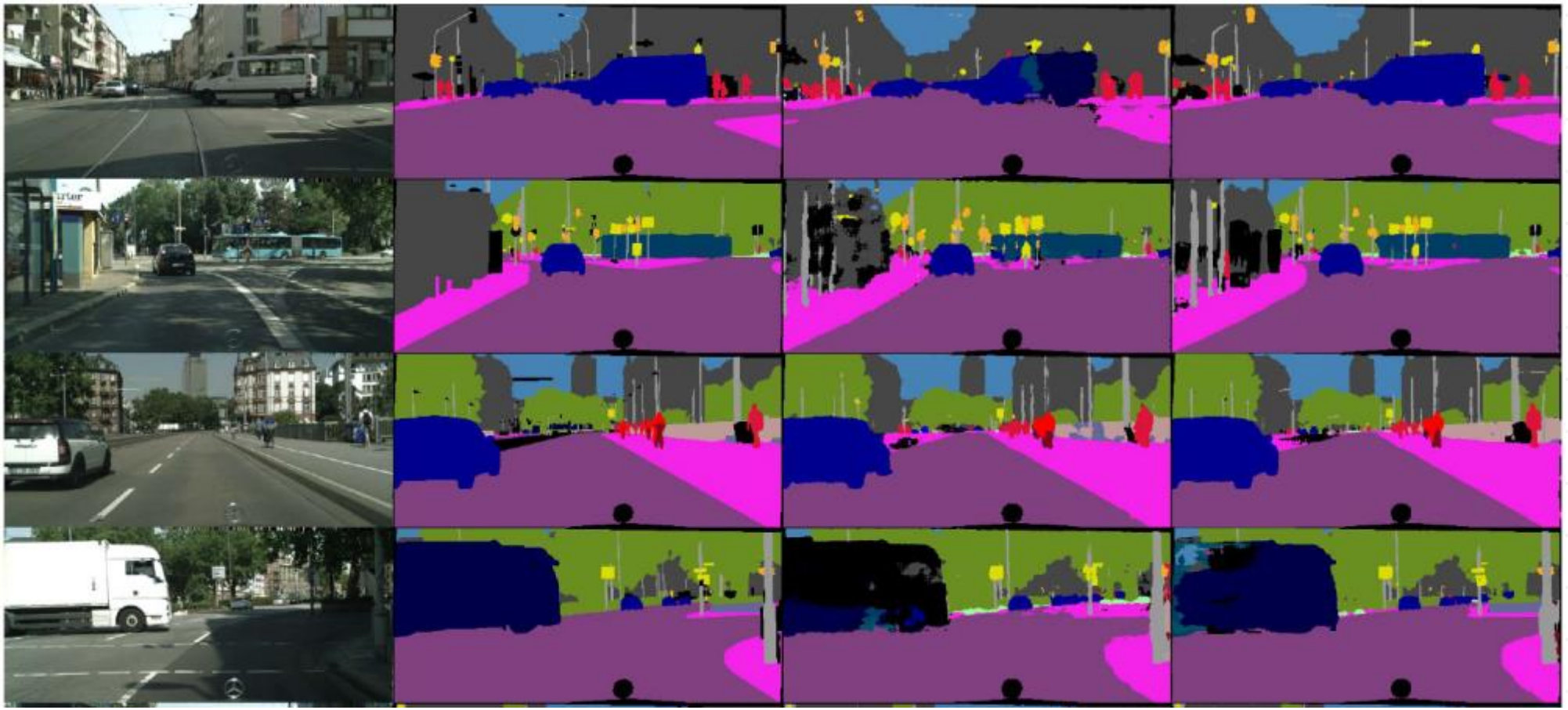
Recap: semantic segmentation



"*Upconvolution*"

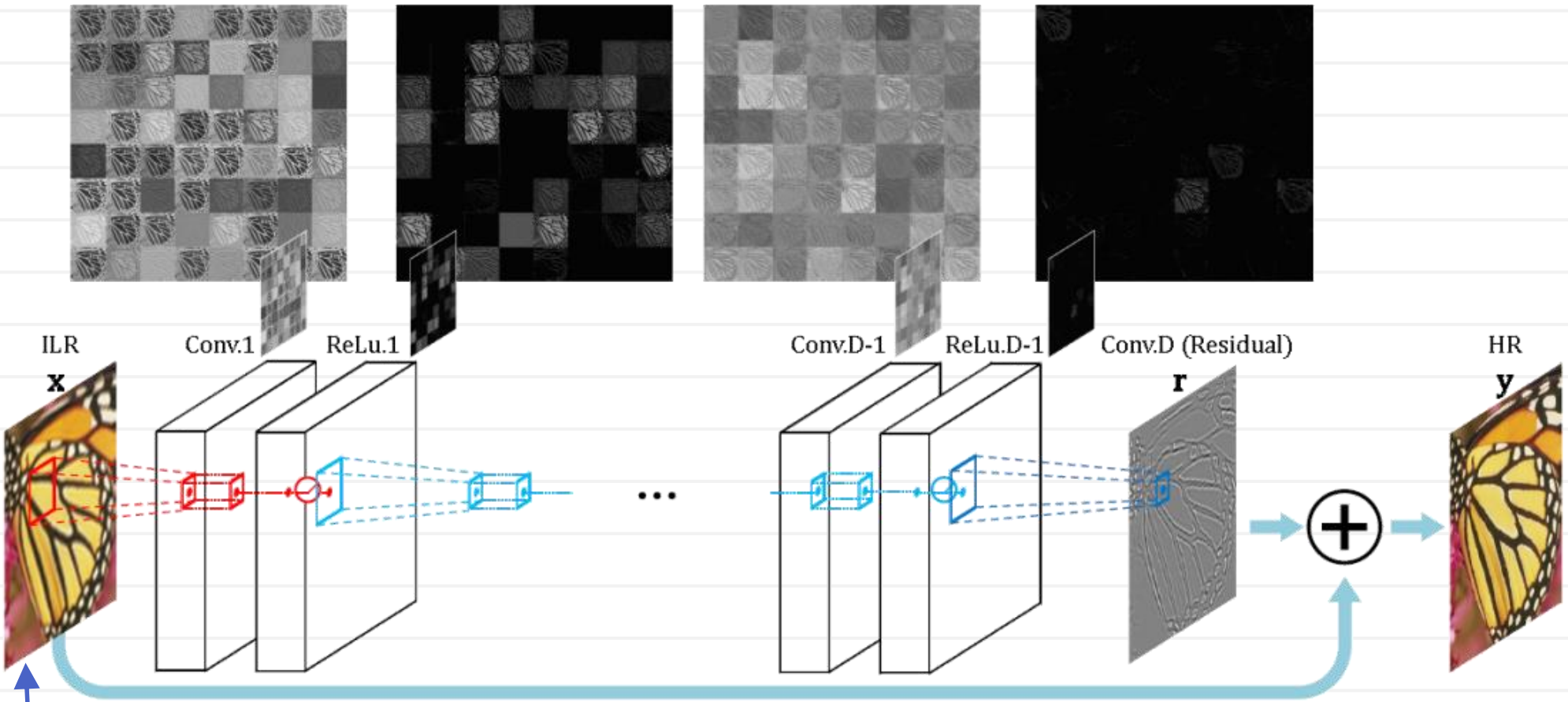


Recap: semantic segmentation



GT ENet ERFNet

Image superresolution with ConvNets



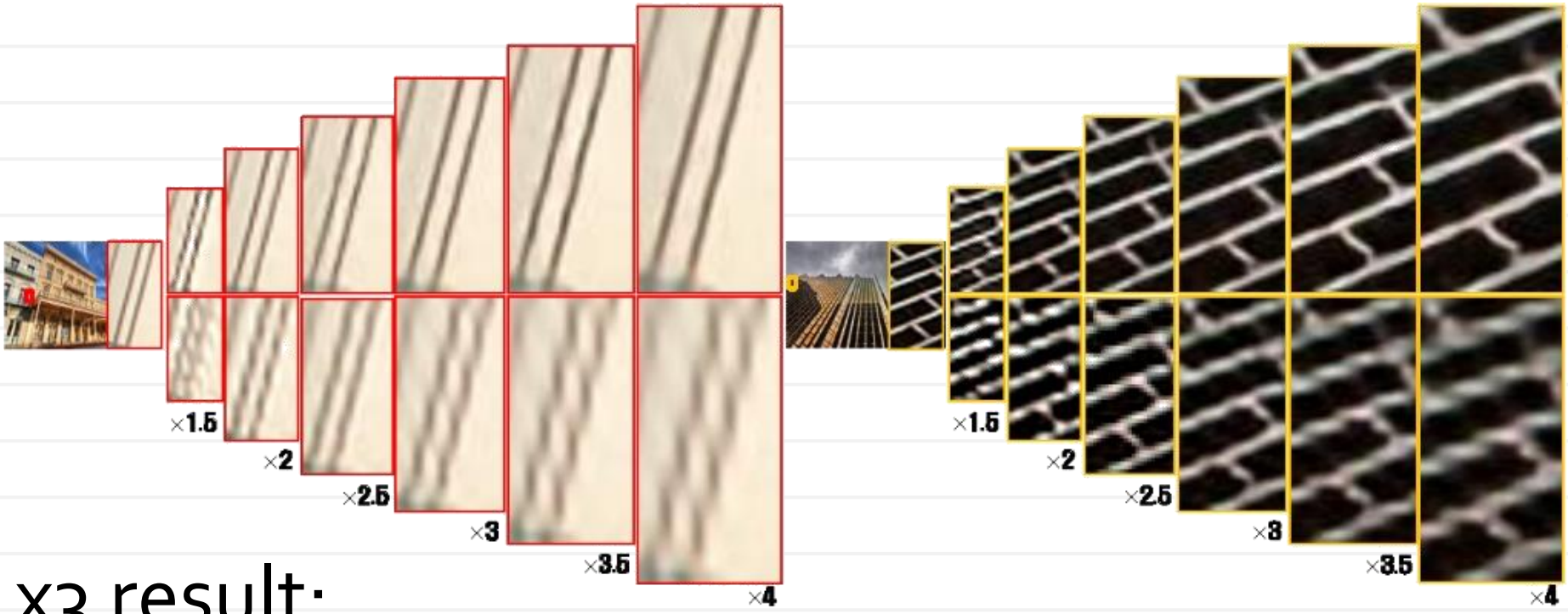
- 19 layers with 3x3 filters (VGG inspired)
- Training on synthetic data

$$\frac{1}{2}||\mathbf{y} - f(\mathbf{x})||^2$$

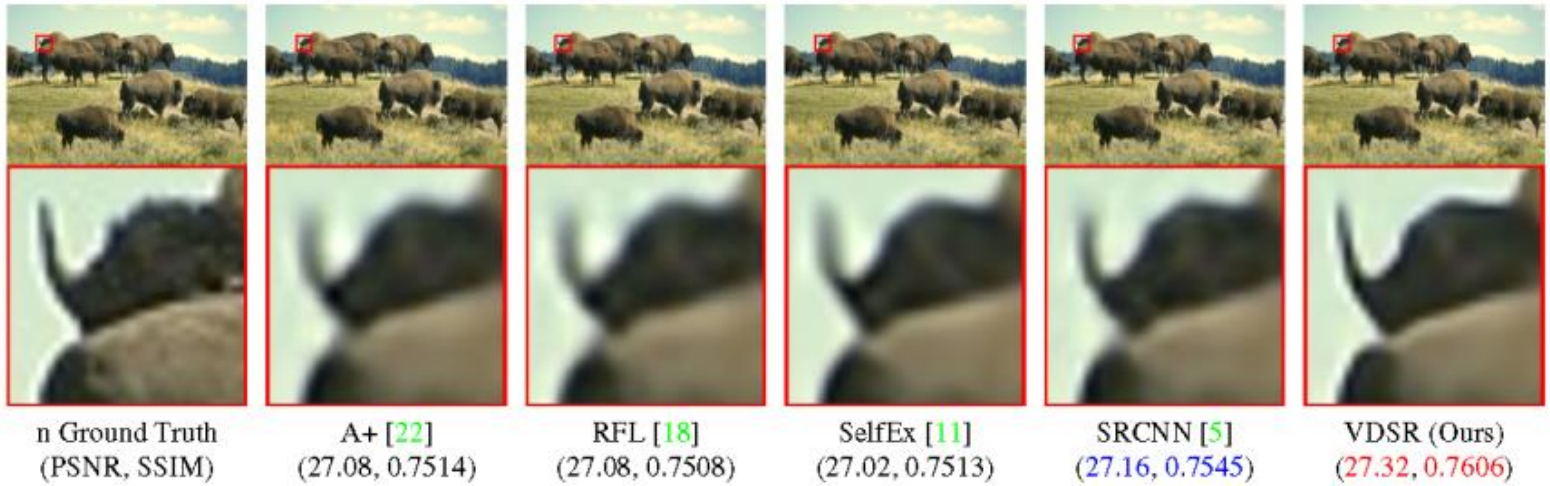
[Kim et al. CVPR16]

Interpolated low-res image

Image superresolution with ConvNets



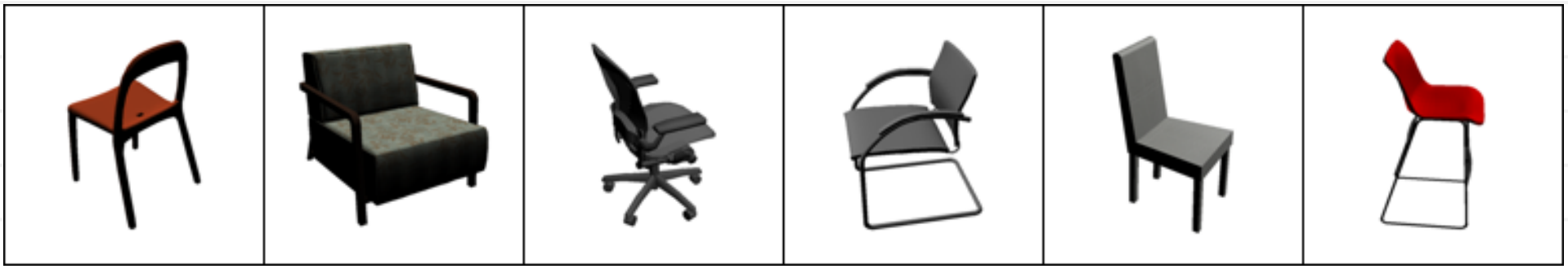
x3 result:



[Kim et al. CVPR16]

Feed-forward conditional generation

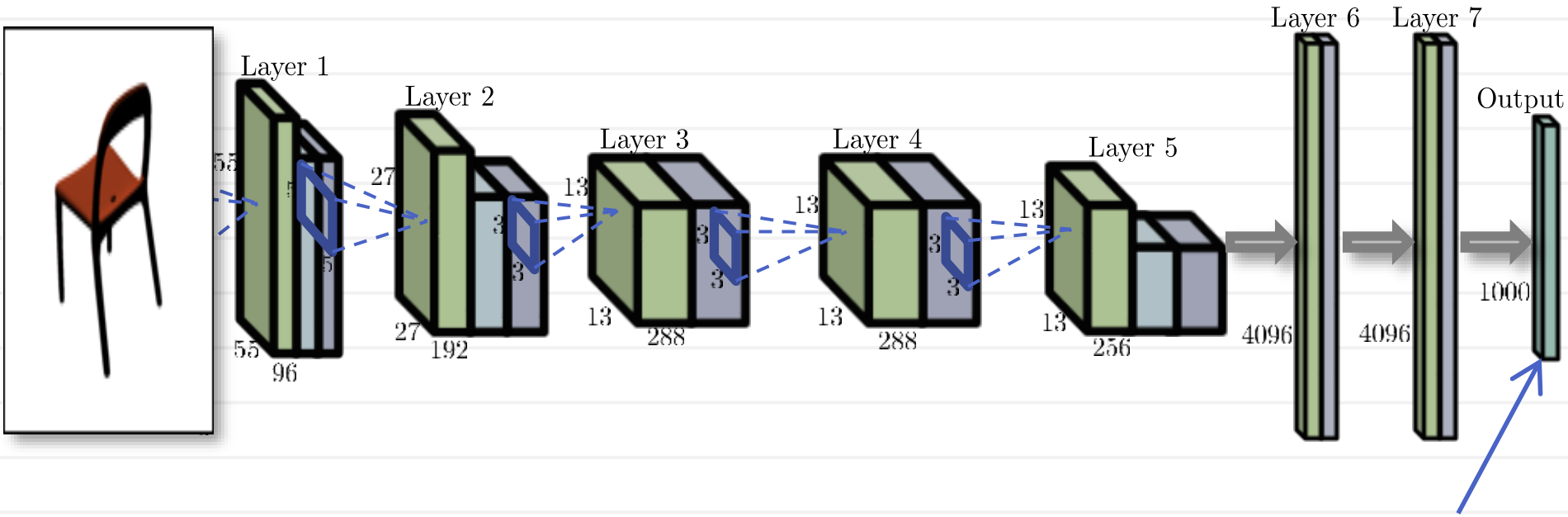
Source images:



Each image has "chair ID", and viewpoint direction

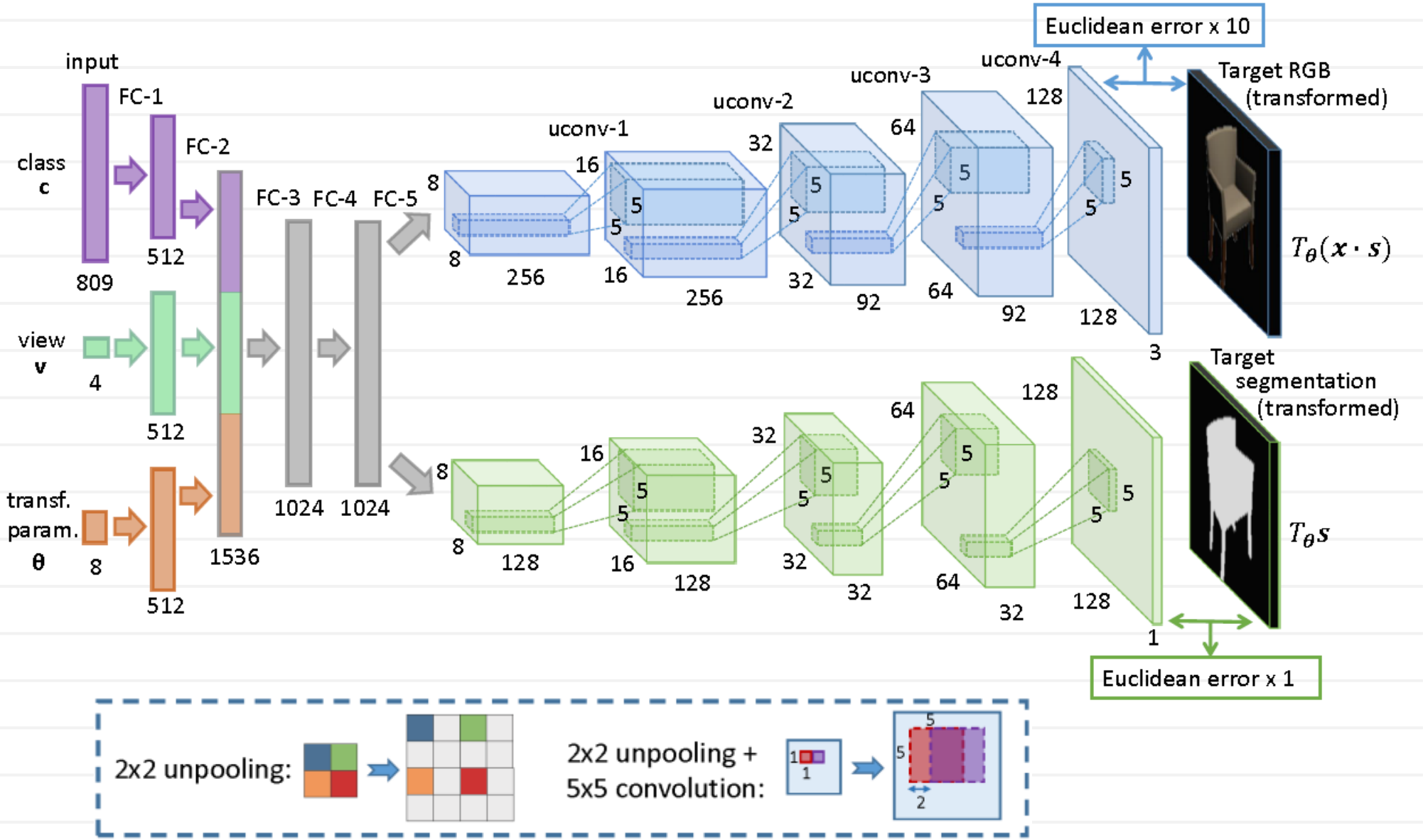
[Dosovitskiy et al. CVPR 2015]

"Natural" dataflow



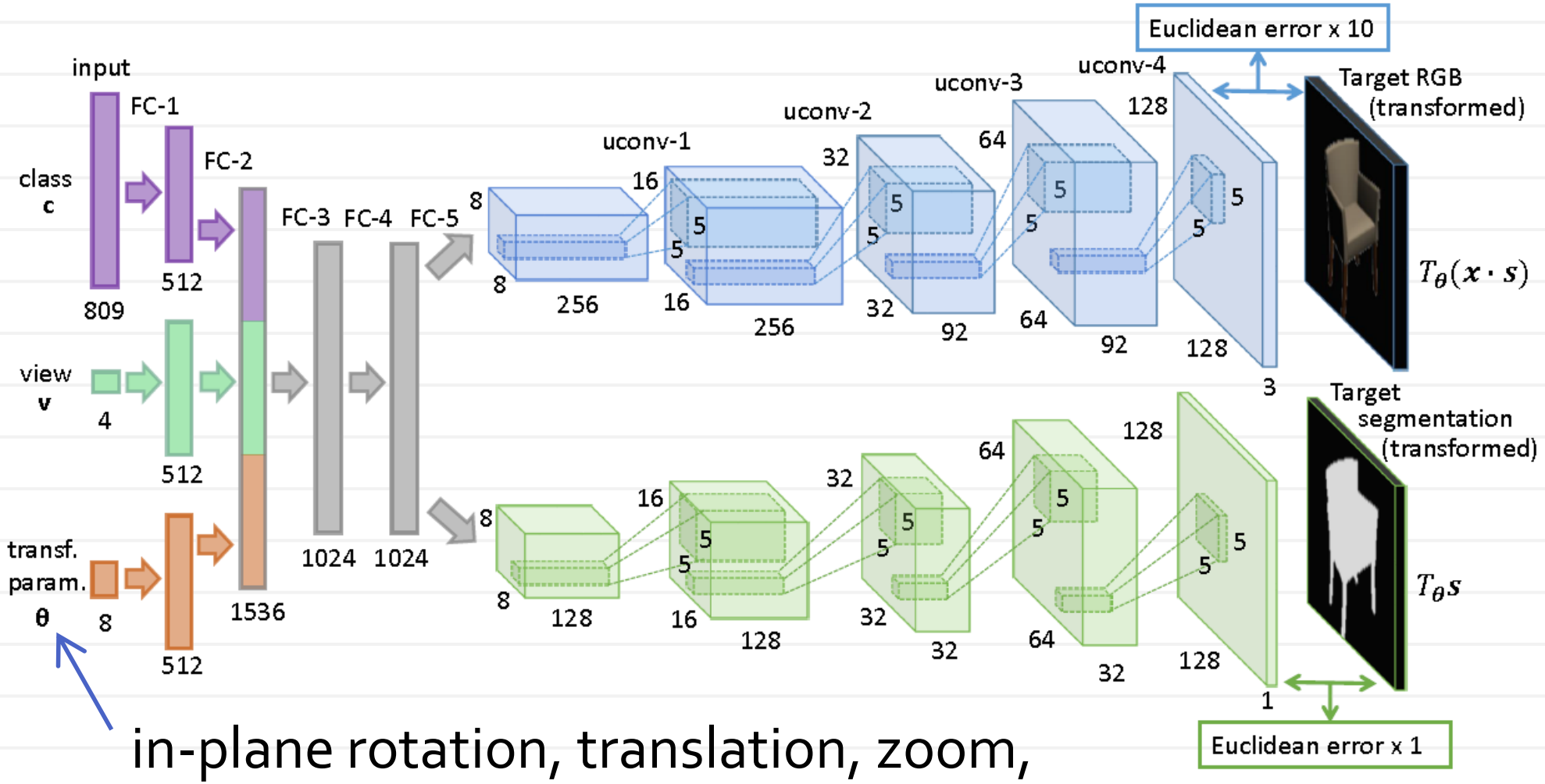
chair type
view angle
etc.

Feed-forward conditional generation



[Dosovitskiy et al. CVPR 2015]

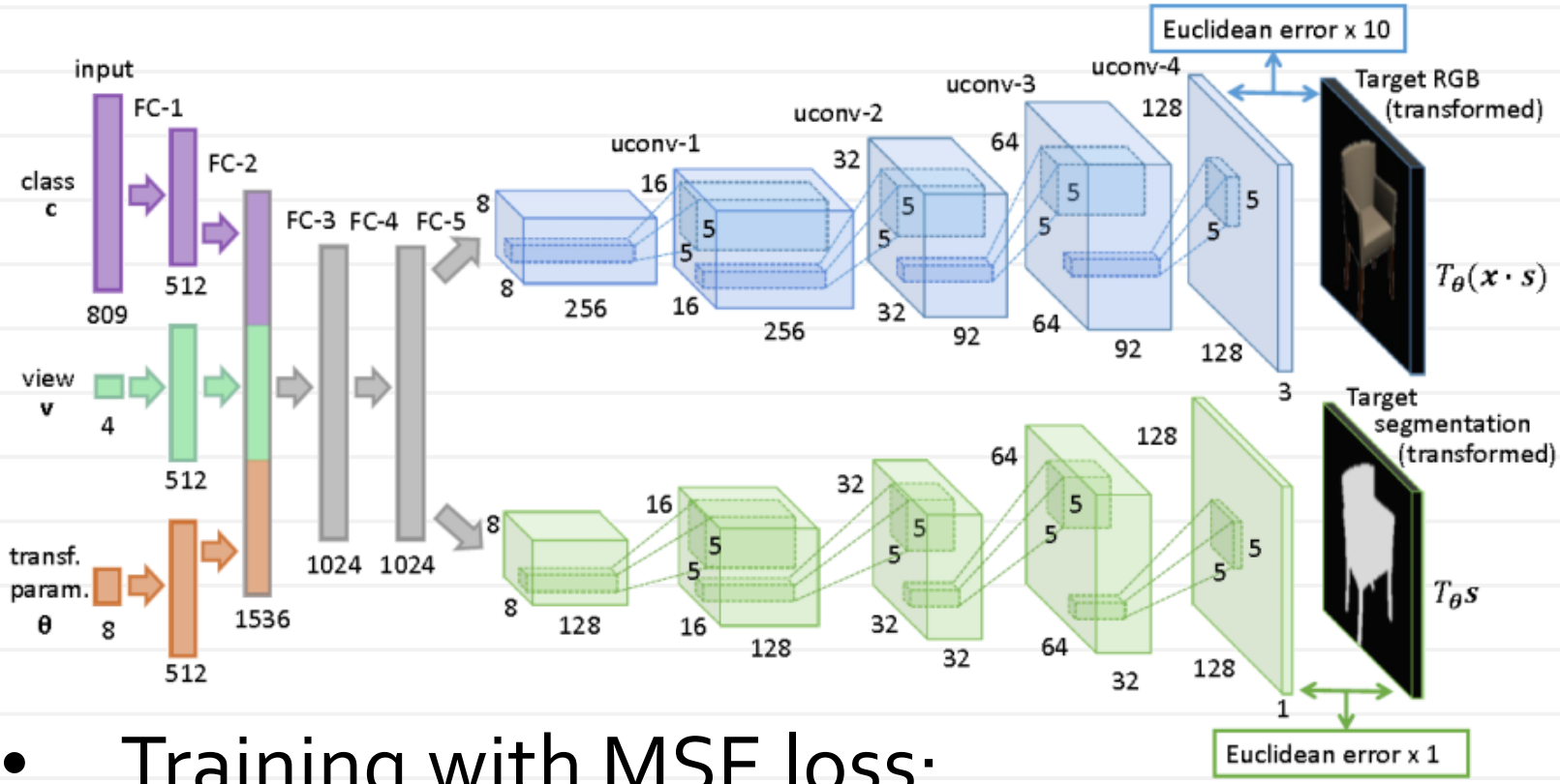
Feed-forward conditional generation



in-plane rotation, translation, zoom, stretching horizontally or vertically, changing hue, changing saturation, changing brightness

[Dosovitskiy et al. CVPR 2015]

Feed-forward conditional generation



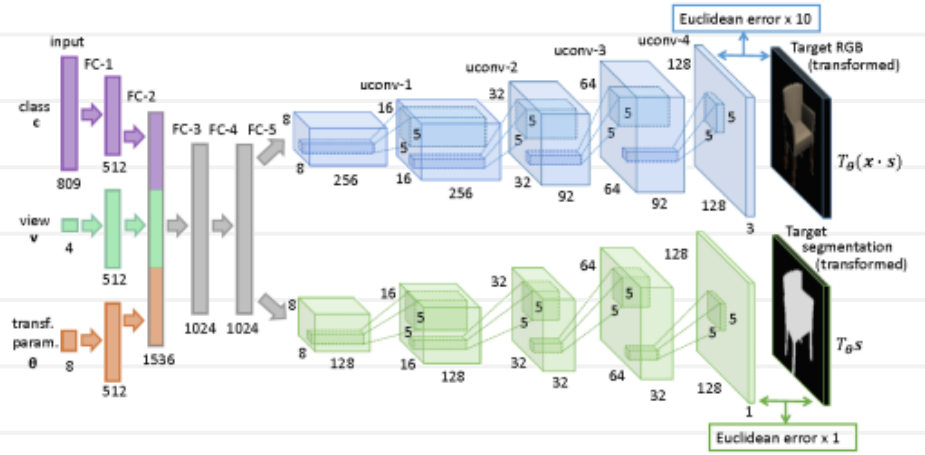
- Training with MSE loss:

$$\min_{\mathbf{W}} \sum_{i=1}^{N} \lambda \| u_{RGB}(h(\mathbf{c}^i, \mathbf{v}^i, \theta^i)) - T_{\theta^i}(\mathbf{x}^i \cdot \mathbf{s}^i) \|_2^2$$

$$+ \| u_{segm}(h(\mathbf{c}^i, \mathbf{v}^i, \theta^i)) - T_{\theta^i} \mathbf{s}^i \|_2^2,$$

[Dosovitskiy et al. CVPR 2015]

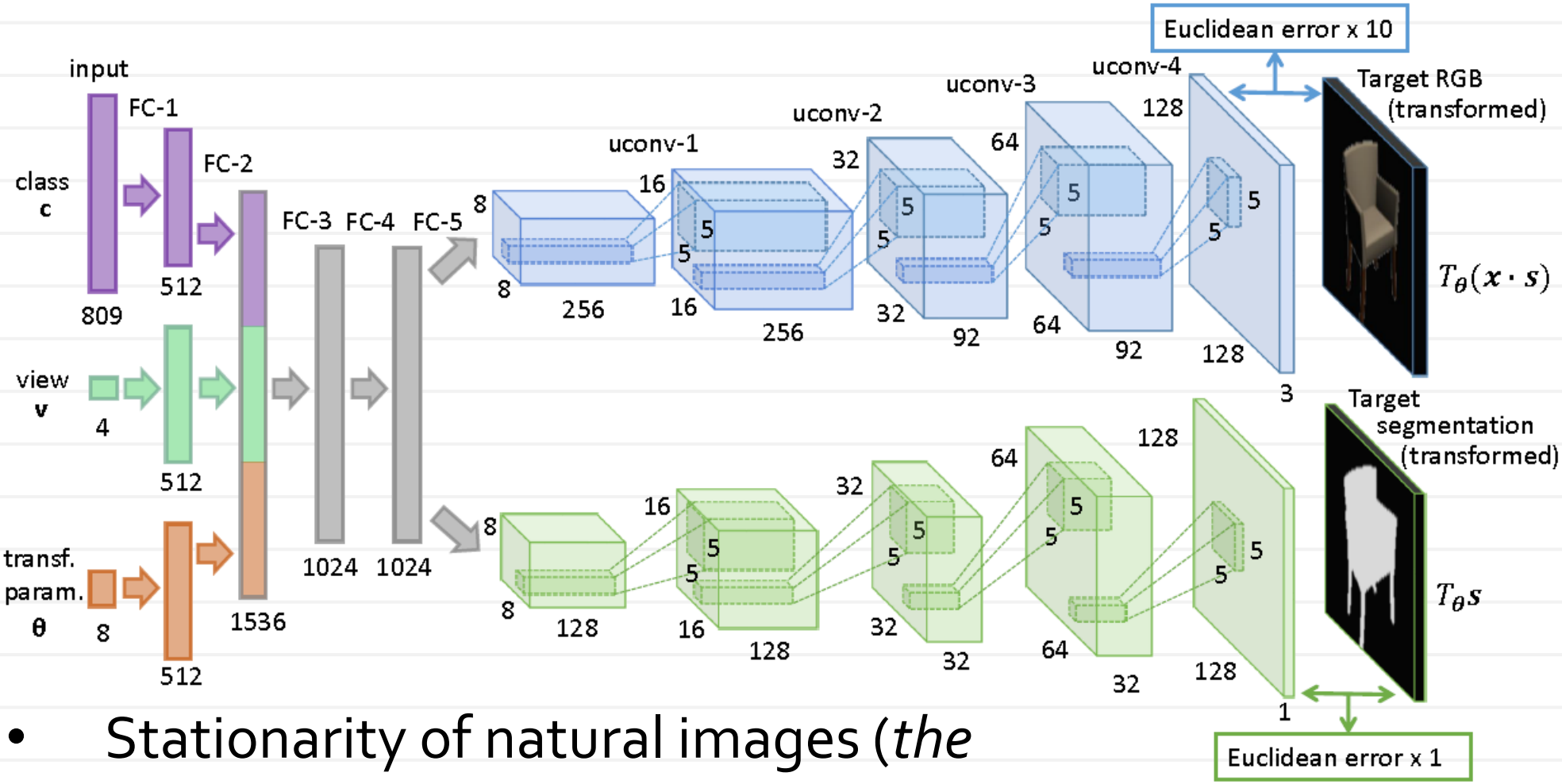
Feed-forward conditional generation



Morphing between chairs

[Dosovitskiy et al. CVPR 2015]

Why does it work?



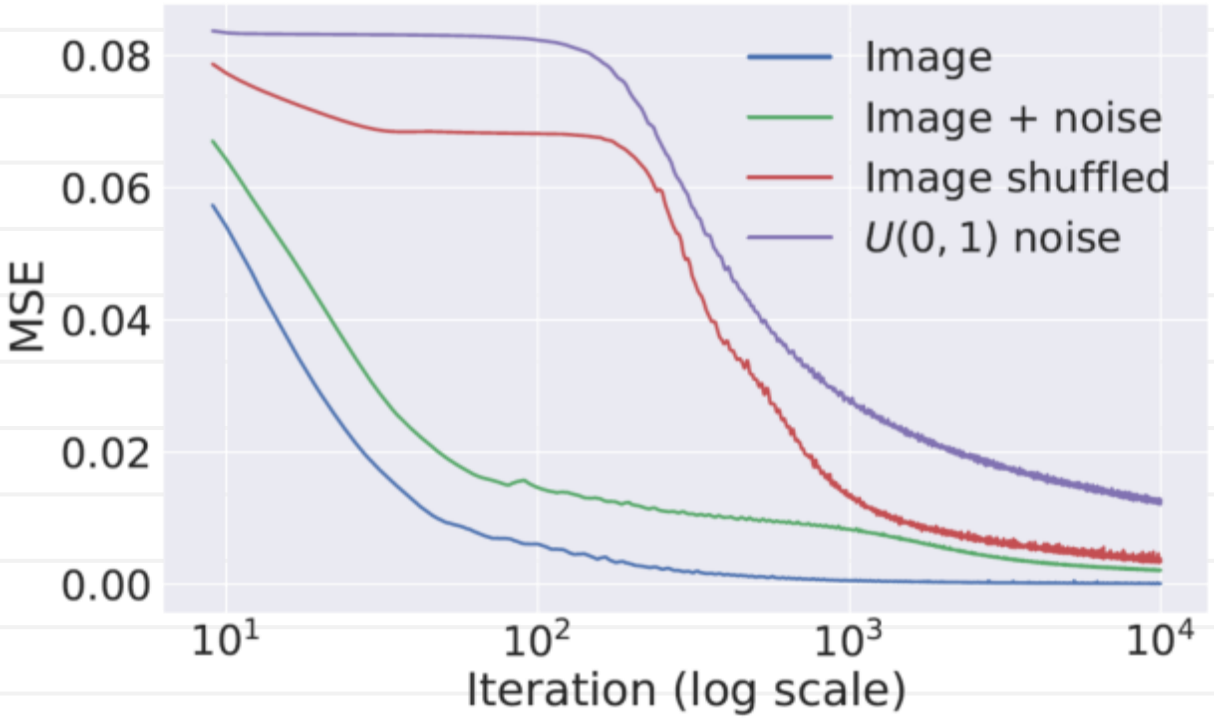
- Stationarity of natural images (*the appearance of object parts are invariant/covariant w.r.t. 2D translation*)
- Abundance of supervision

Deep Image Prior: toy example

fixed random map

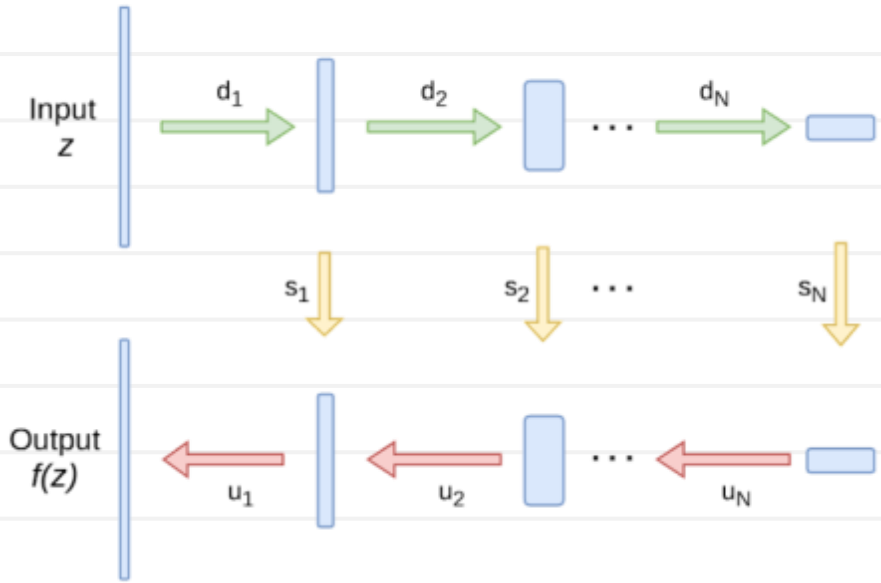
$$\min_{\theta} E(f_\theta(z); x_0)$$

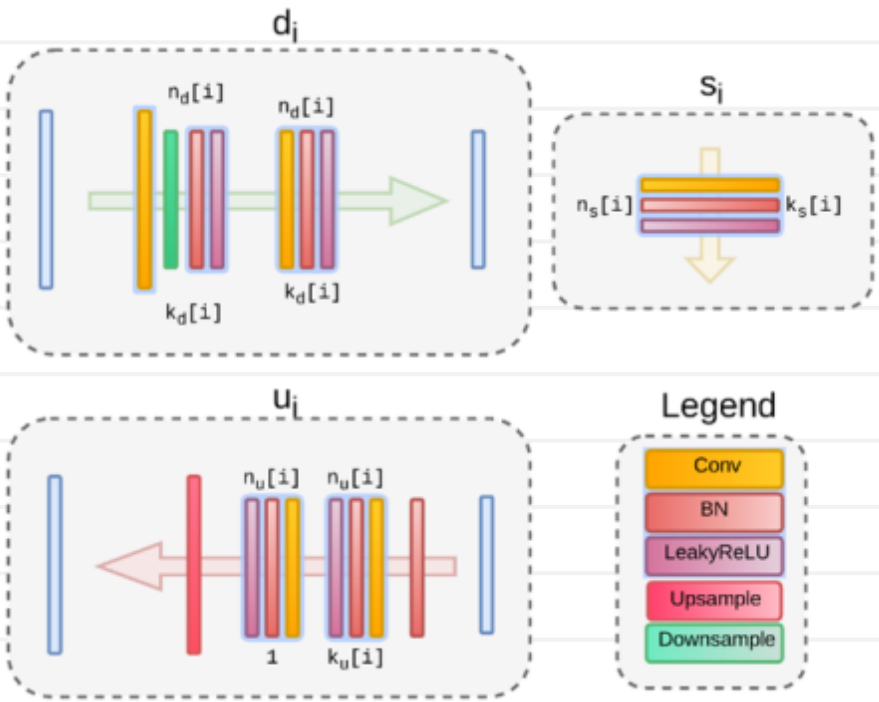
$$E(x; x_0) = \|x - x_0\|^2$$



[Ulyanov et al. CVPR18]

Architecture





Architecture (U-Net type)

Building blocks

[Ulyanov et al. CVPR18]

Image restoration: standard approach

- x – Recovered image
- x_0 – Corrupted image (observed)

Regularization / Prior term

$$\arg\min_x E(x; x_0) + R(x)$$

- **Denoising:** $E(x; x_0) = \|x - x_0\|^2$
- **Inpainting:** $E(x; x_0) = \|(x - x_0) \odot m\|^2$
- **Super-Res.:** $E(x; x_0) = \|d(x) - x_0\|^2$
- **Feature inv.:** $E(x; x_0) = \|\Phi(x) - \Phi(x_0)\|^2$

Image restoration with *Deep Image Prior*

"Classical" MAP approach to inverse problems:

$$\arg\min_x E(x; x_0) + R(x)$$

Consider all images obtained from a random signal *z* via a convolutional network with a certain architecture :

$$g(\theta) \equiv f_\theta(z)$$

Fixed input

Convolutional network with parameters θ

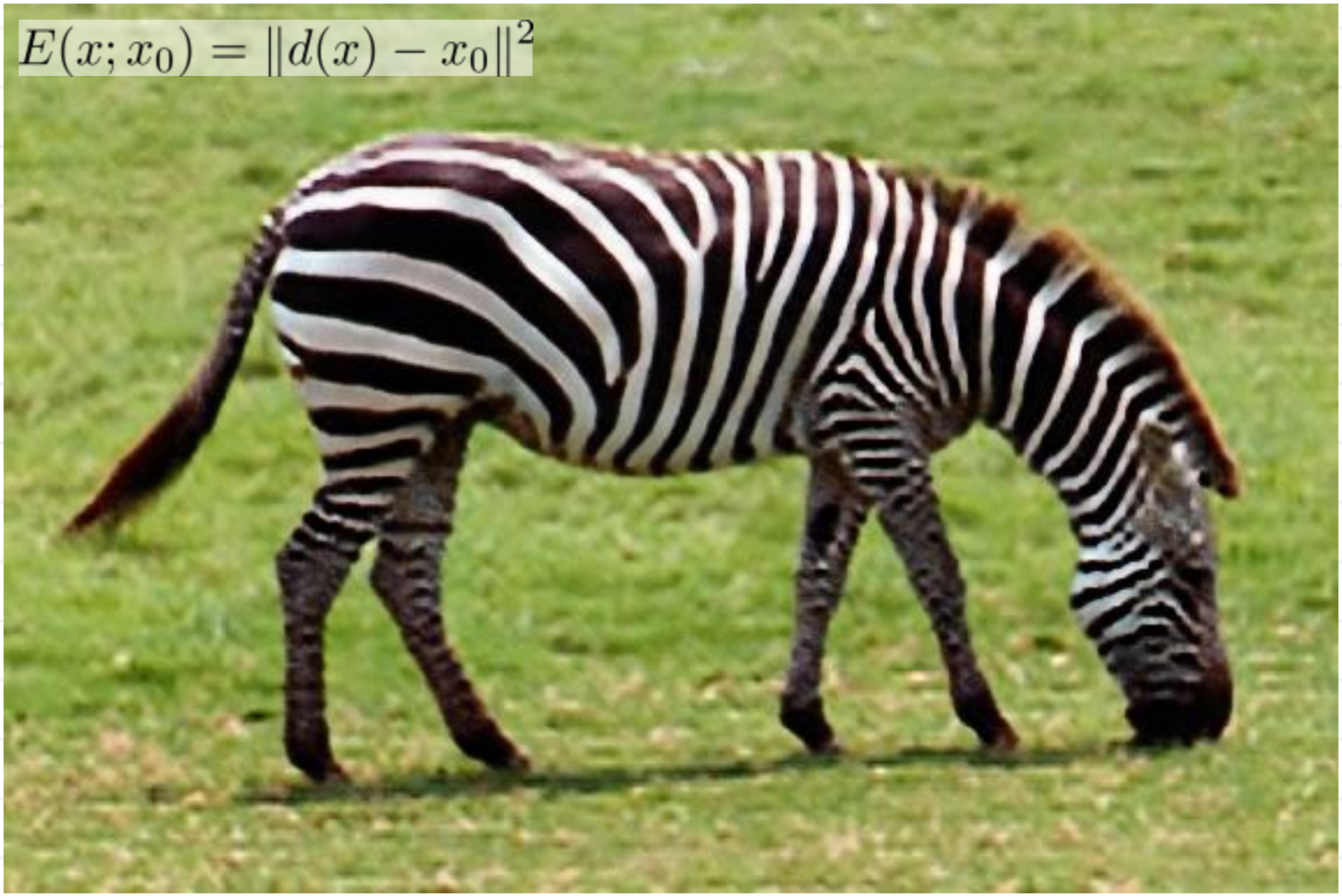
Perform the reconstruction by solving (*optionally: use fixed number of iterations*):

$$\min_\theta E(f_\theta(z); x_0)$$

"Find the most likely image that can be generated by a ConvNet from z"

[Ulyanov et al. CVPR18]

Deep superresolution without training



[Ulyanov et al. CVPR18]

Deep superresolution without training

[Ulyanov et al. CVPR18]

Image restoration

$$E(x; x_0) = \|(x - x_0) \odot m\|^2$$

Corrupted **Deep image prior**

[Ulyanov et al. CVPR18]

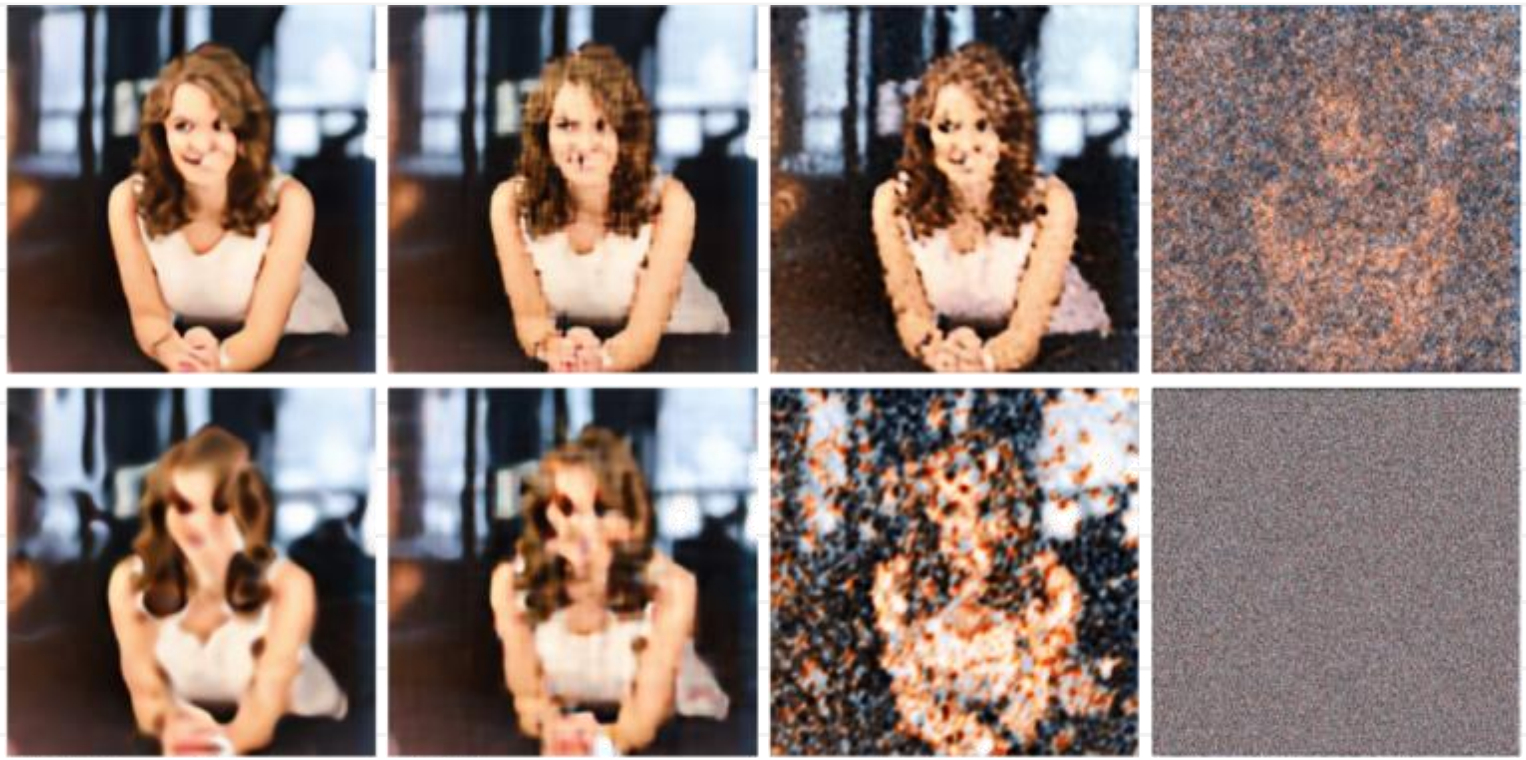
Deep inpainting without training

[Ulyanov et al. CVPR18]

Comparing architectures

98% occluded

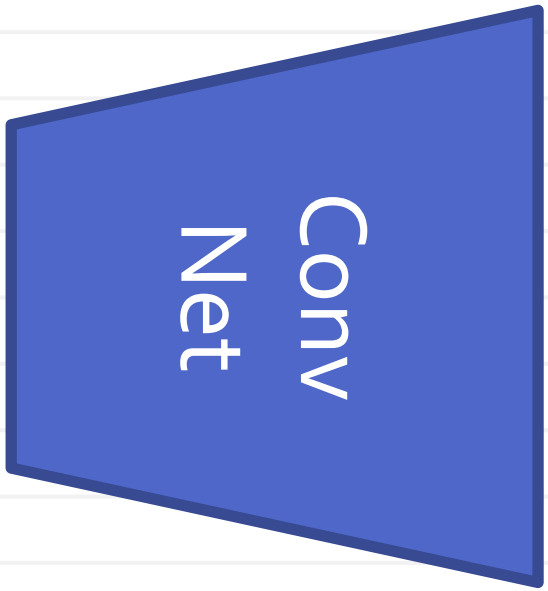
99.5% occluded



(d) Hourglass bilinear upsampling | (e) Hourglass nearest upsampling | (f) U-Net | (g) ResNet

[Ulyanov et al. CVPR18]

The pitfall of the pixel-wise losses

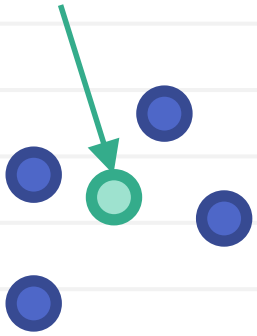


$$\hat{\theta} = \arg\min_{\theta} \sum_{i} \| f(x_i; \theta) - y_i \|^2$$

Averaging over jittered versions = blurriness

The pitfall of the pixel-wise losses

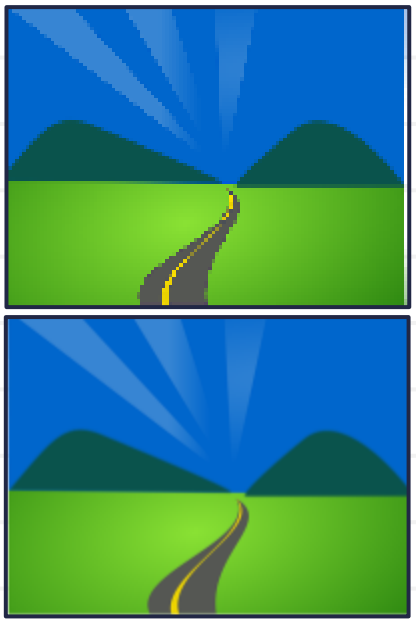
Best prediction under L2-loss

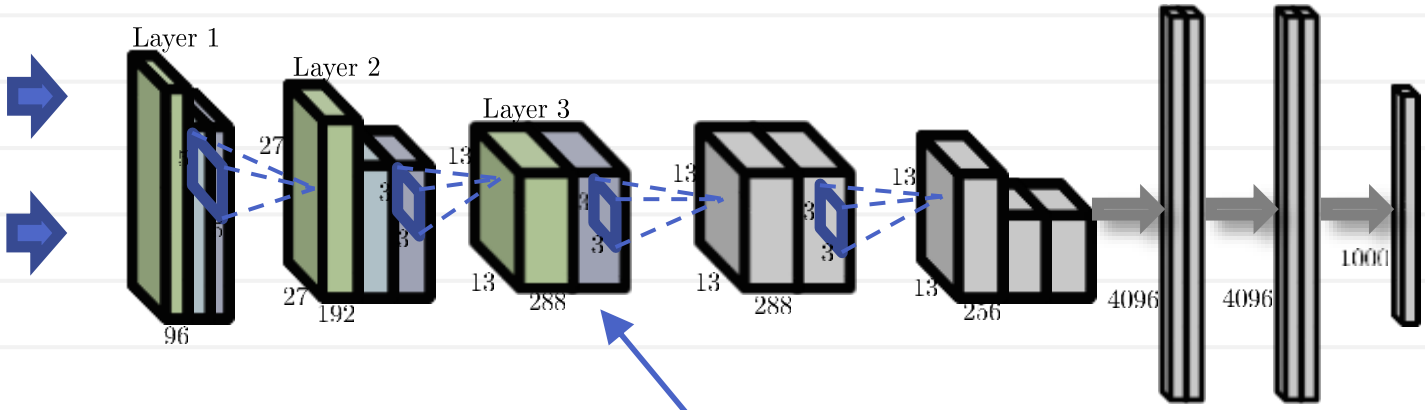


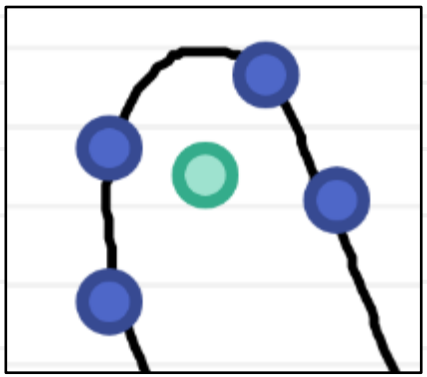
High-res image manifold

Low-res image manifold

Perceptual loss functions



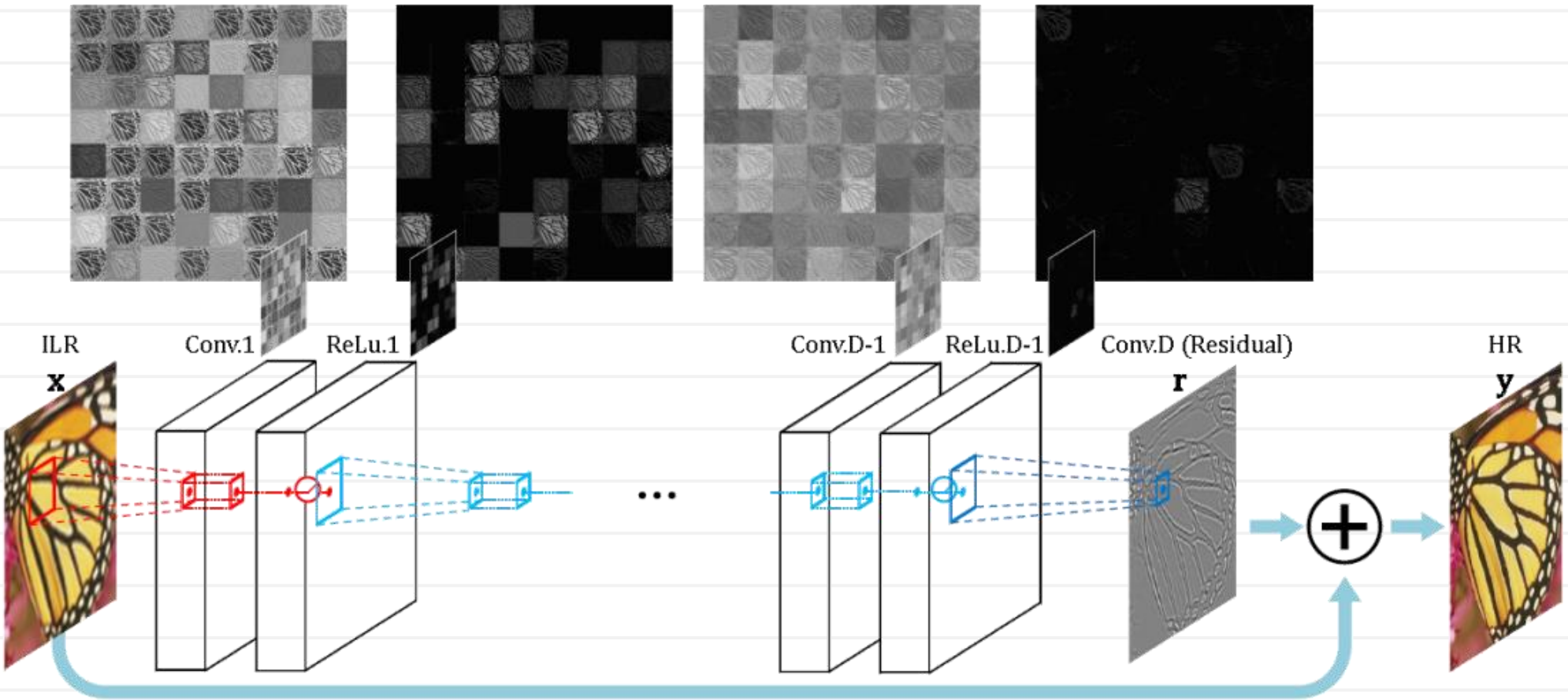




$$\mathcal{L}_C(\mathbf{x};\mathbf{y}) = \sum_{l \in L_C} \sum_{i=1}^{N_l} \|F_i^l(\mathbf{x}) - F_i^l(\mathbf{y})\|_2^2$$

- Averaged image is far from all blue
- Blue points are similar to each other

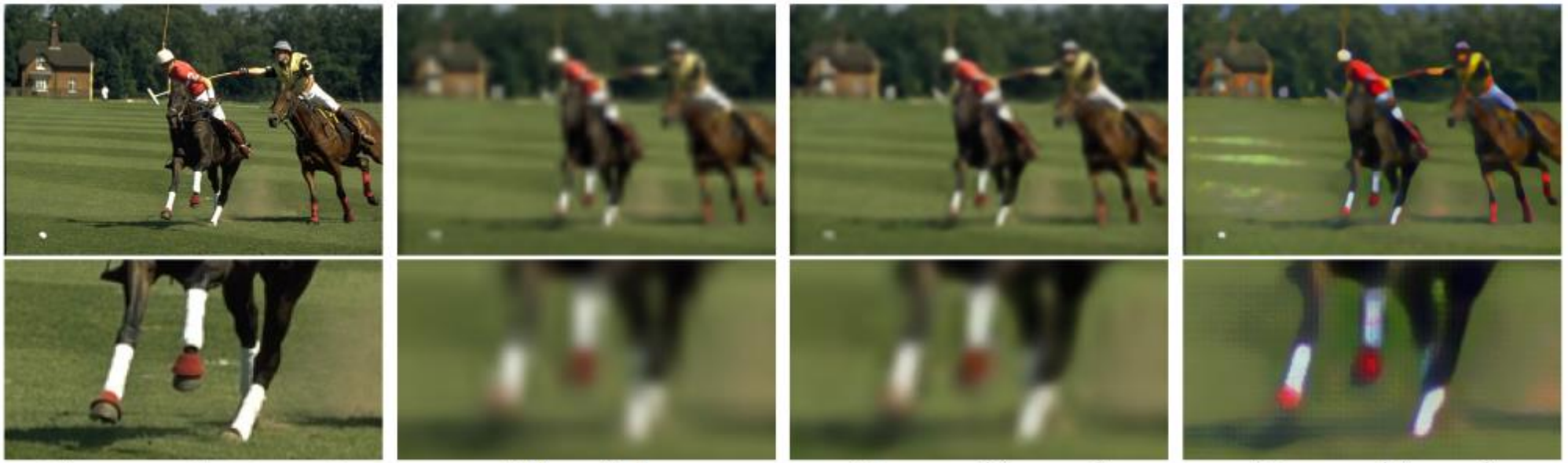
Recap: superresolution using ConvNets



- 19 layers with 3x3 filters (VGG inspired)

[Kim et al. CVPR16]

Superresolution with perceptual losses



Ground Truth	Bicubic	Ours (ℓ_{pixel})	Ours (ℓ_{feat})
This image	22.75 / 0.5946	23.42 / 0.6168	21.90 / 0.6083
Set5 mean	23.80 / 0.6455	24.77 / 0.6864	23.26 / 0.7058
Set14 mean	22.37 / 0.5518	23.02 / 0.5787	21.64 / 0.5837
BSD100 mean	22.11 / 0.5322	22.54 / 0.5526	21.35 / 0.5474

[Johnson et al. ECCV2016]

Superresolution with perceptual losses

Ground Truth	Bicubic	Ours (ℓ_{pixel})	SRCNN [13]	Ours (ℓ_{feat})
This image	31.78 / 0.8577	31.47 / 0.8573	32.99 / 0.8784	29.24 / 0.7841
Set5 mean	28.43 / 0.8114	28.40 / 0.8205	30.48 / 0.8628	27.09 / 0.7680

[Johnson et al. ECCV2016]

High-res generation with perceptual losses

[Chen and Koltun ICCV17]

High-res generation with perceptual losses

[Chen and Koltun ICCV17]

High-res generation with perceptual losses

- Cascade of refinement modules starting from 4x8 upto 1024x2048
- Simple variant outputs 1 variant and uses perceptual loss:

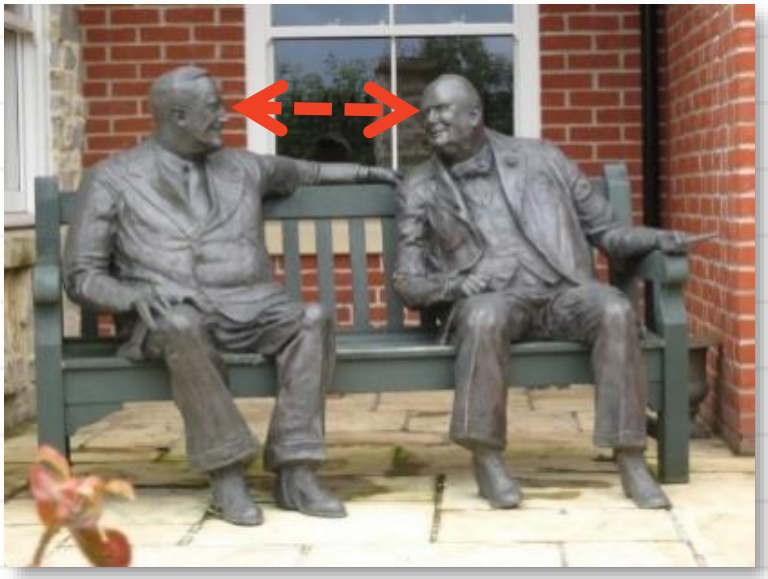
$$\mathcal{L}_{I,L}(\theta) = \sum_l \lambda_l \|\Phi_l(I) - \Phi_l(g(L;\theta))\|_1$$

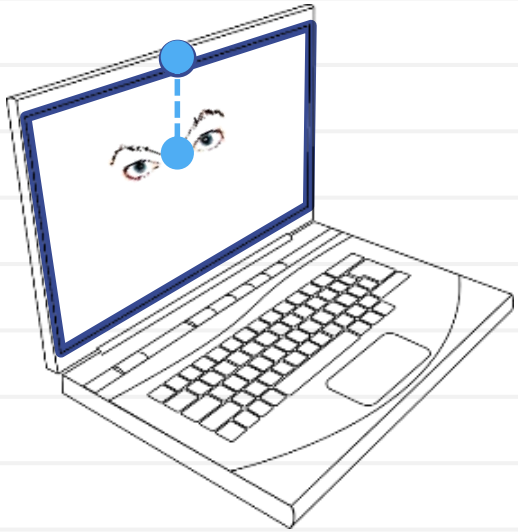
- Multiple-choice learning improves results:

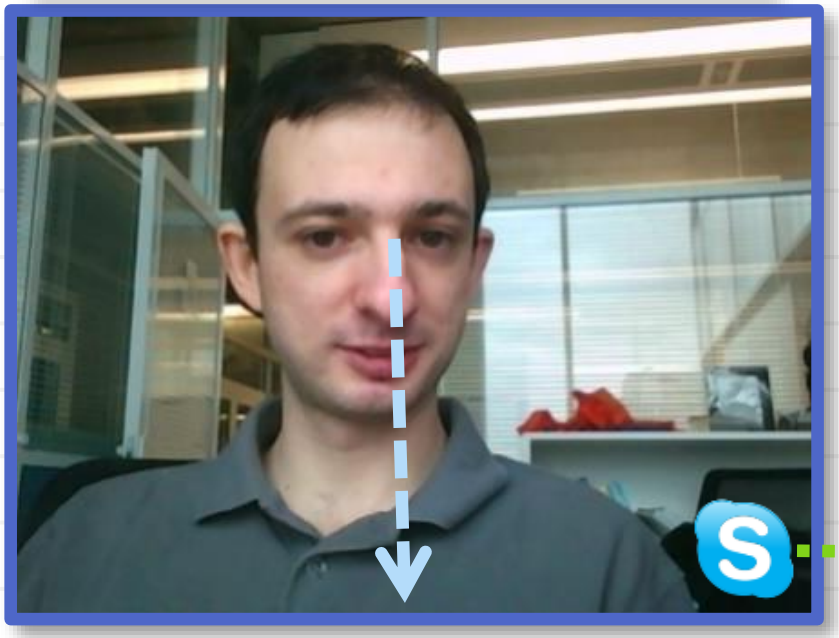
$$\sum_{p=1}^{c} \min_u \sum_l \lambda_l \sum_j \left\| L_p^l \odot \left(\Phi_l^j(I) - \Phi_l^j(g_u(L;\theta))\right)\right\|_1$$

[Chen and Koltun ICCV17]

Problem: lack of eye contact





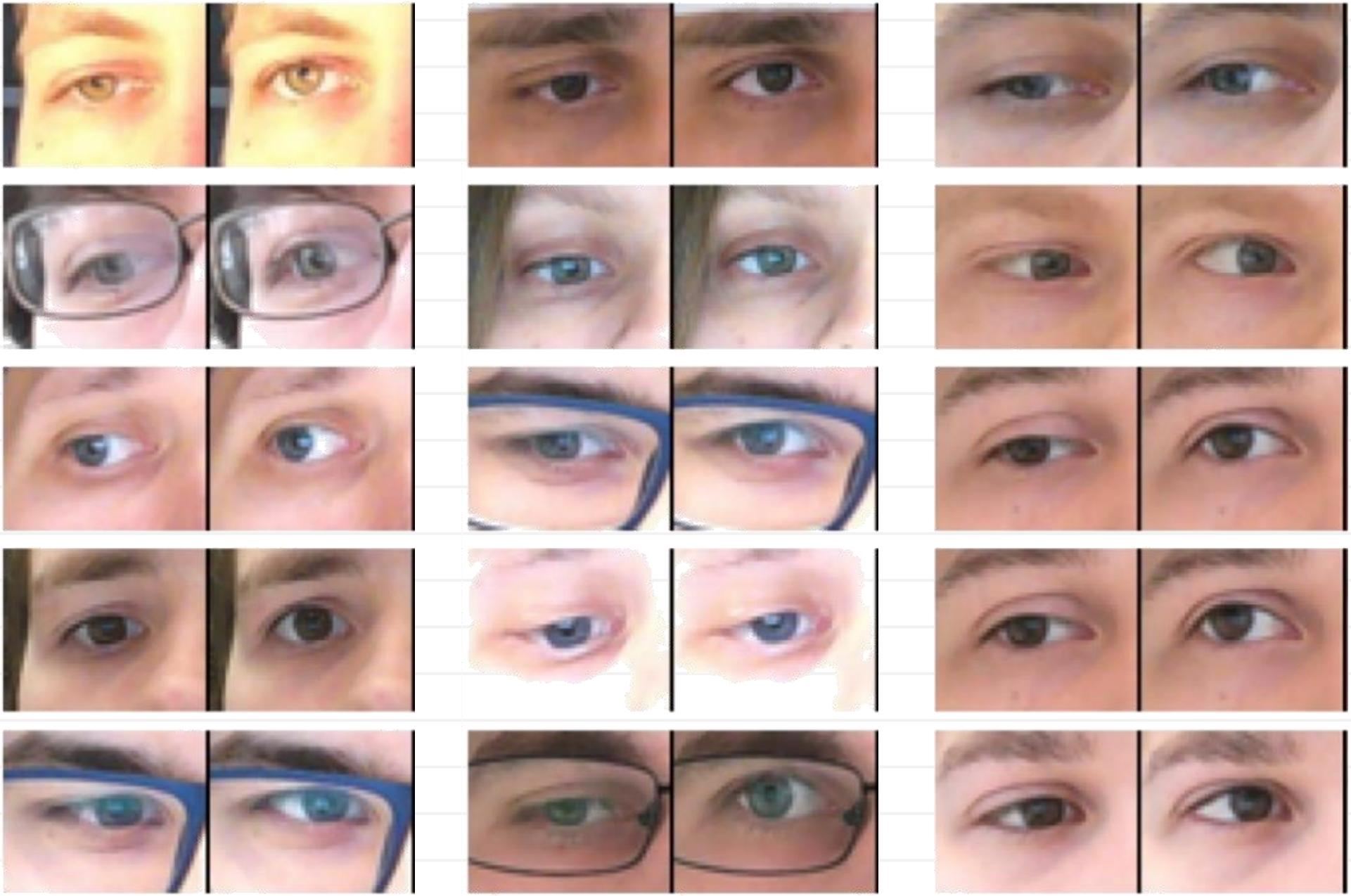


Supervised learning is hard

Goal: capture appearance changes from gaze redirection.

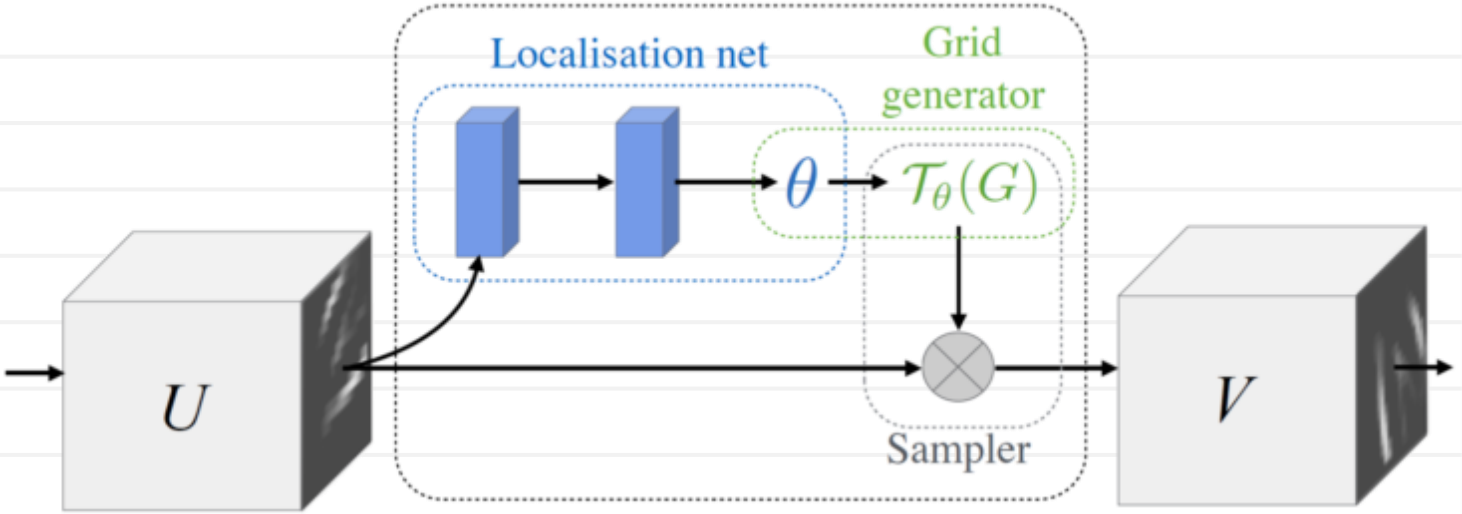
[Ganin et al. 2016]

Harnessing training examples

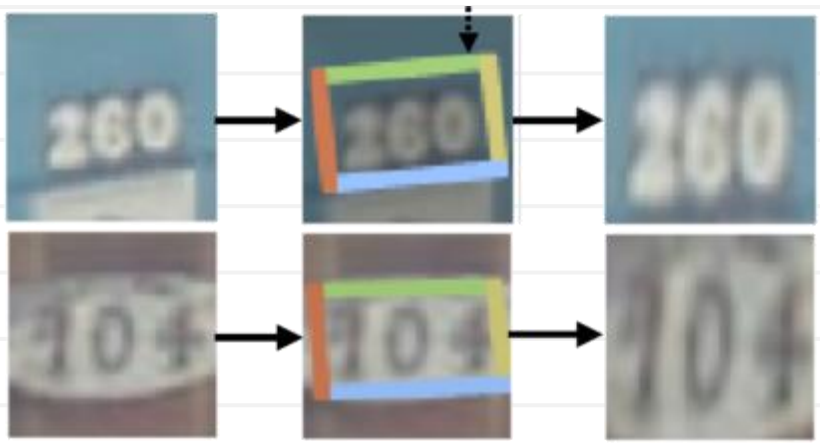


Spatial transformer networks

[Jadegberg et al. NIPS 2015]

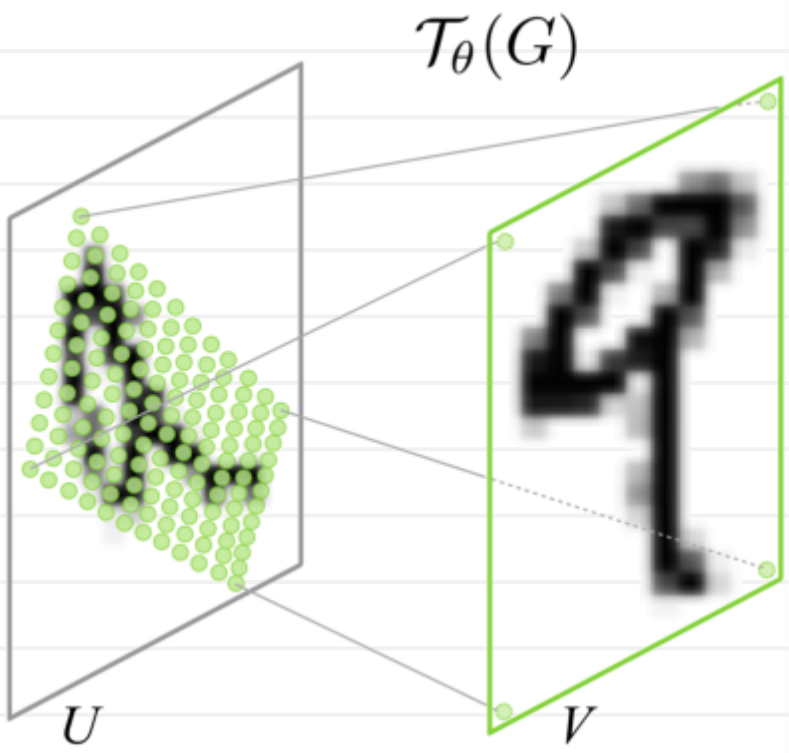


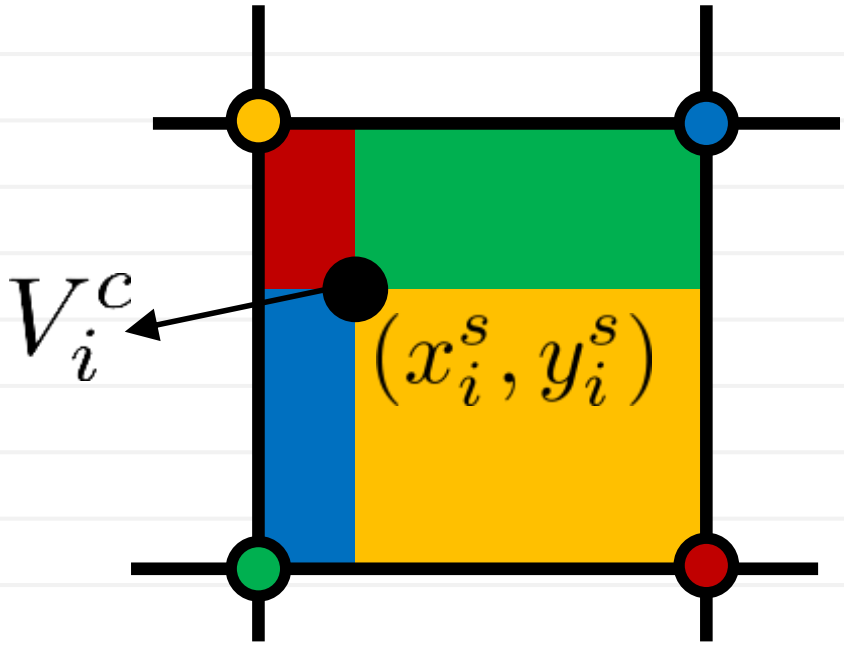
Spatial transformer in action:



Bilinear sampling layer

- Was introduced as part of *Spatial transformer*
- Very useful in generative networks





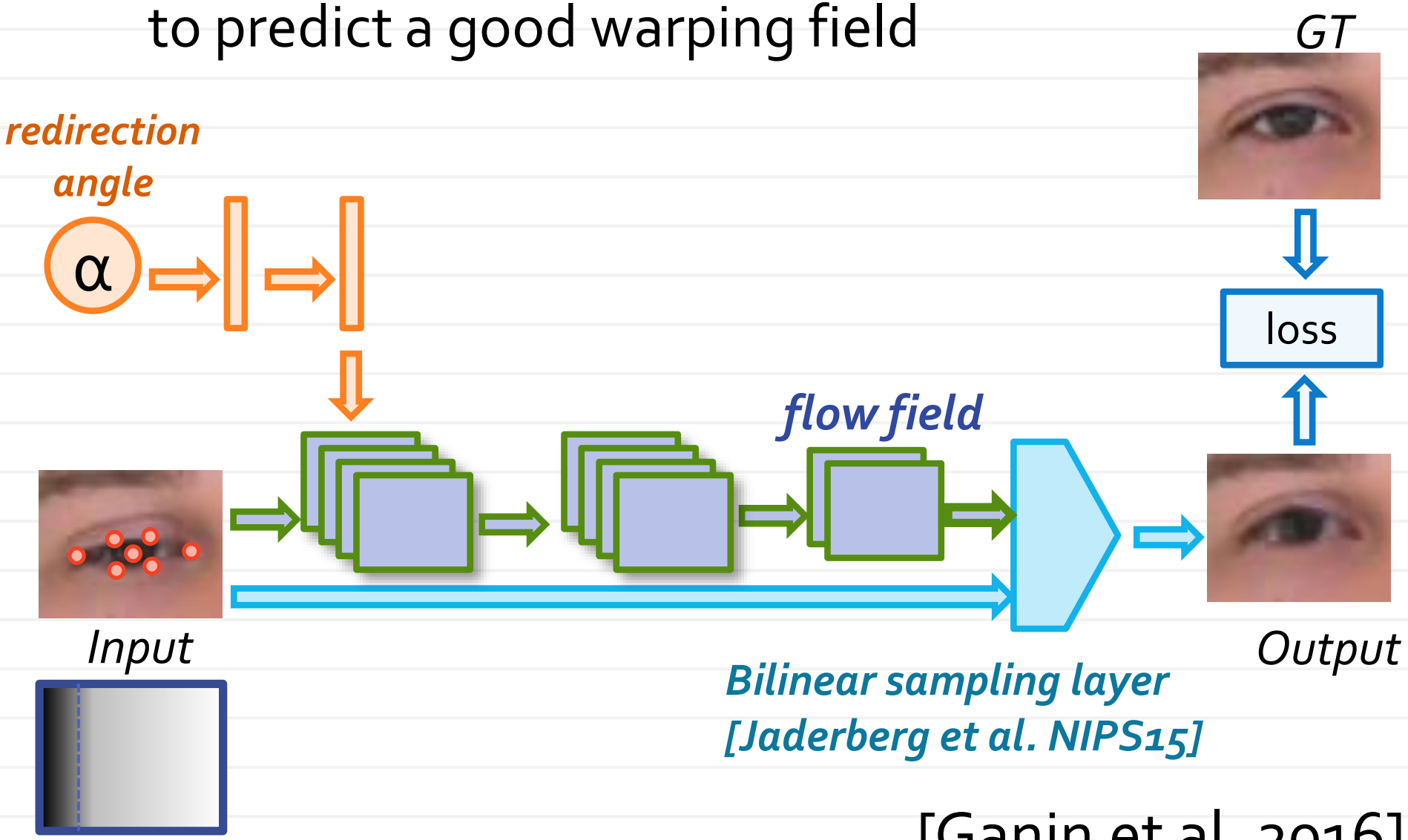
Piecewise-differentiable!

$$V_i^c = \sum_n^H \sum_m^W U_{nm}^c \max(0, 1 - |x_i^s - m|) \max(0, 1 - |y_i^s - n|)$$

[Jadegberg et al. NIPS 2015]

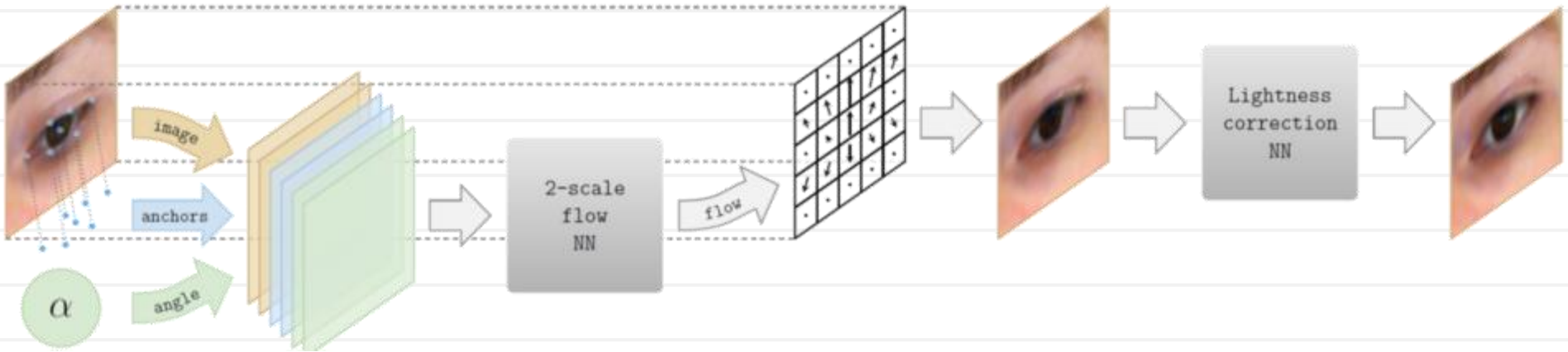
Deep warping

- High-level idea: make a network to predict a good warping field



[Ganin et al. 2016]

Model overview



All three layers can be trained "end-to-end"

[Ganin et al. 2016]

Example results

15° up

15° down

[Ganin et al. 2016]

Results

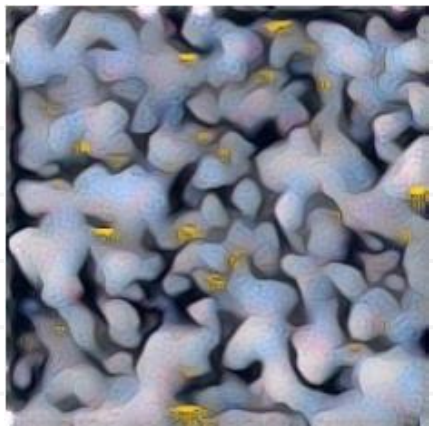
Try yourself:

@MeduzaEyeBot on Telegram *(send any photo with large face, tick "Compress image" when sending)*

[Ganin et al. 2016]

Texture synthesis

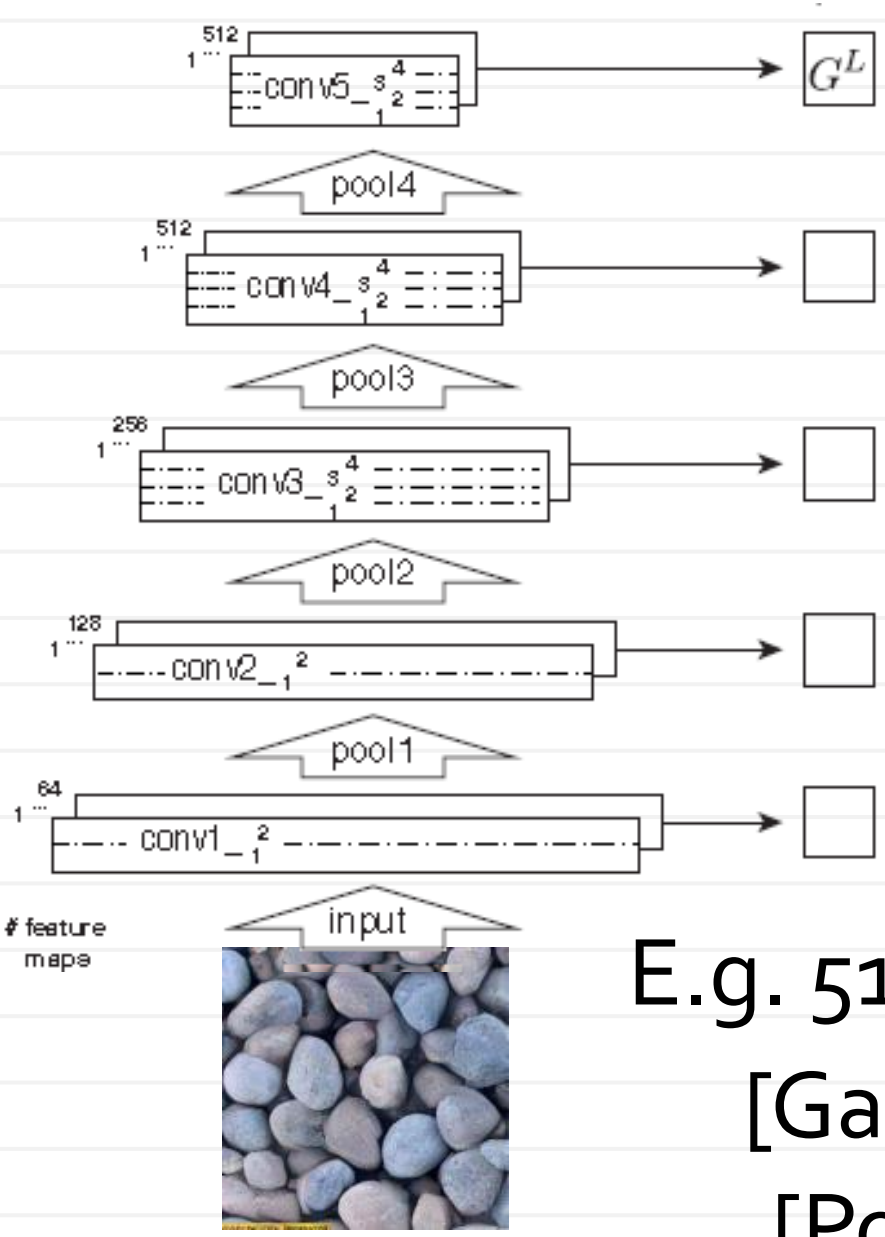
- How can we measure the similarity of *textures (i.e. spatially-stationary natural visual patterns)*?

similar different

- Pixel-wise measures are useless
- Histograms are not too useful
- Long history of research

What describes a texture?



Gram matrix:

$$G^l$$

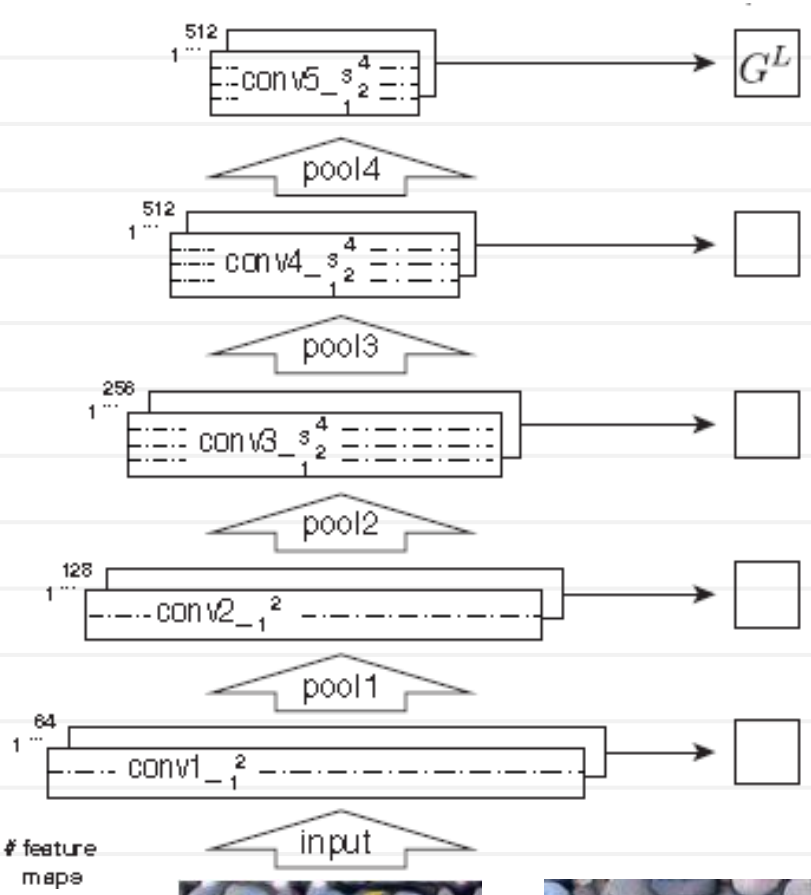
$$G^l_{ij} = \sum_k F^l_{i,k} \cdot F^l_{j,k}$$

E.g. 512x512-dim descriptor

[Gatys, Ecker, Bethge 2015]

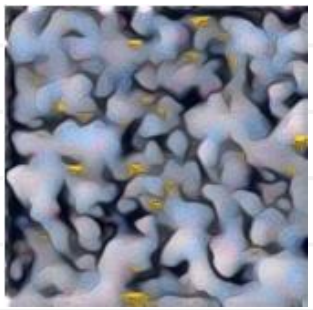
[Portilla Simoncelli 2000]

Pre-image texture synthesis



Texture loss:

$$\mathcal{L}(x) = \sum_l \|G^l(x) - G^l(x^*)\|^2$$



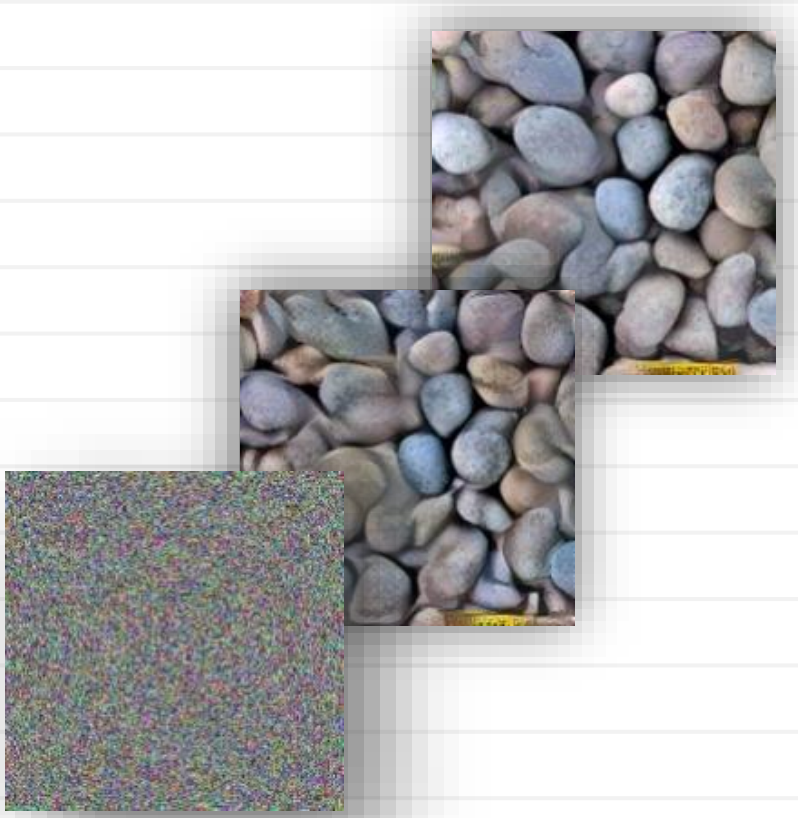
x

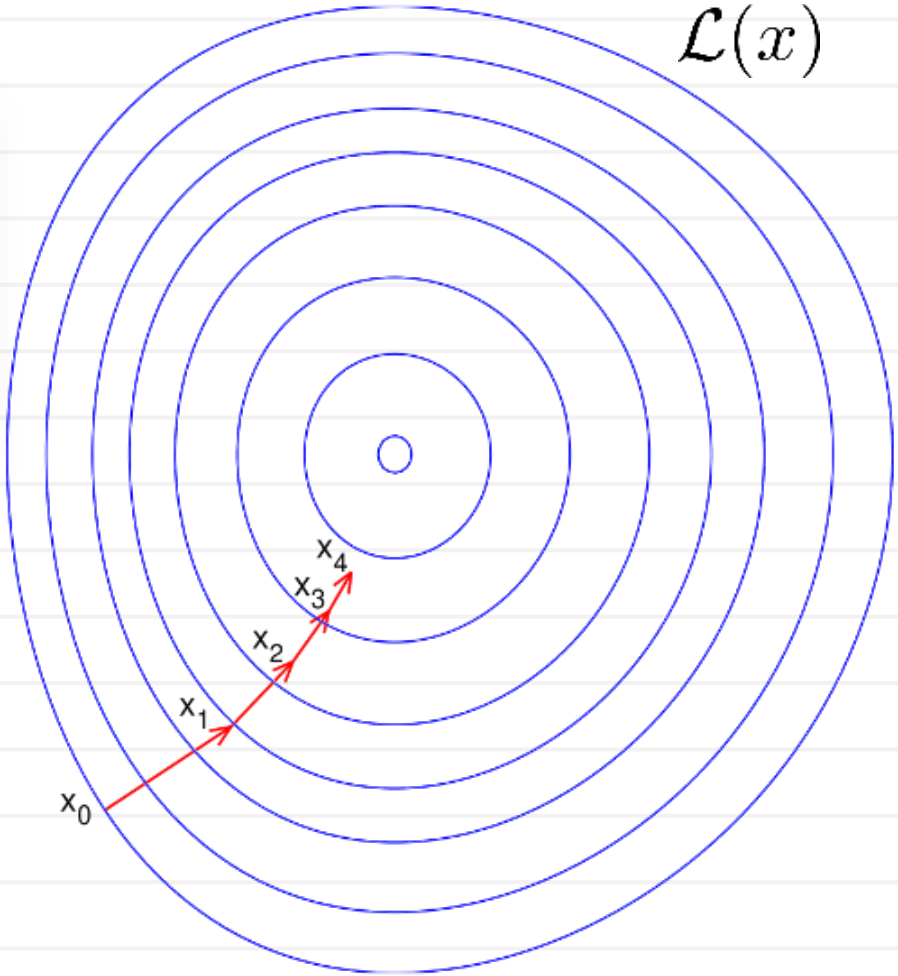
x^*

[Gatys, Ecker, Bethge 2015]

Image generation by optimization

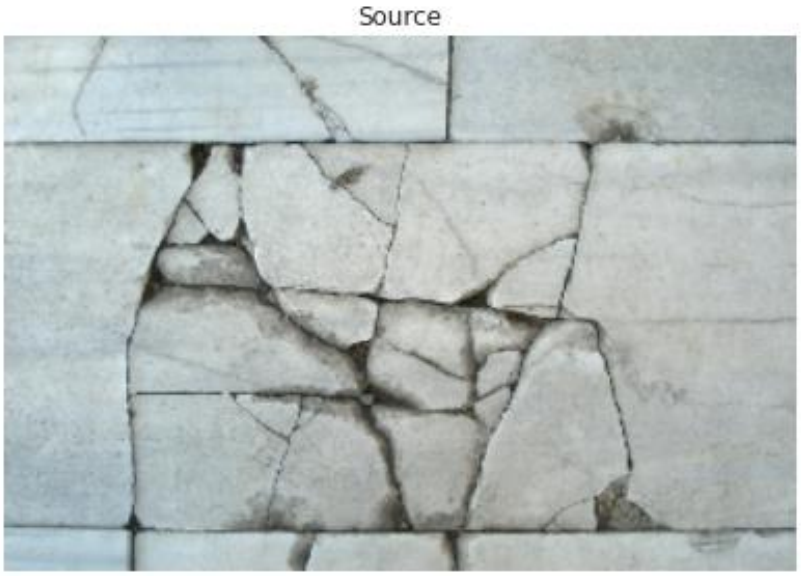
$$x^* = \arg\min_x \mathcal{L}(x)$$





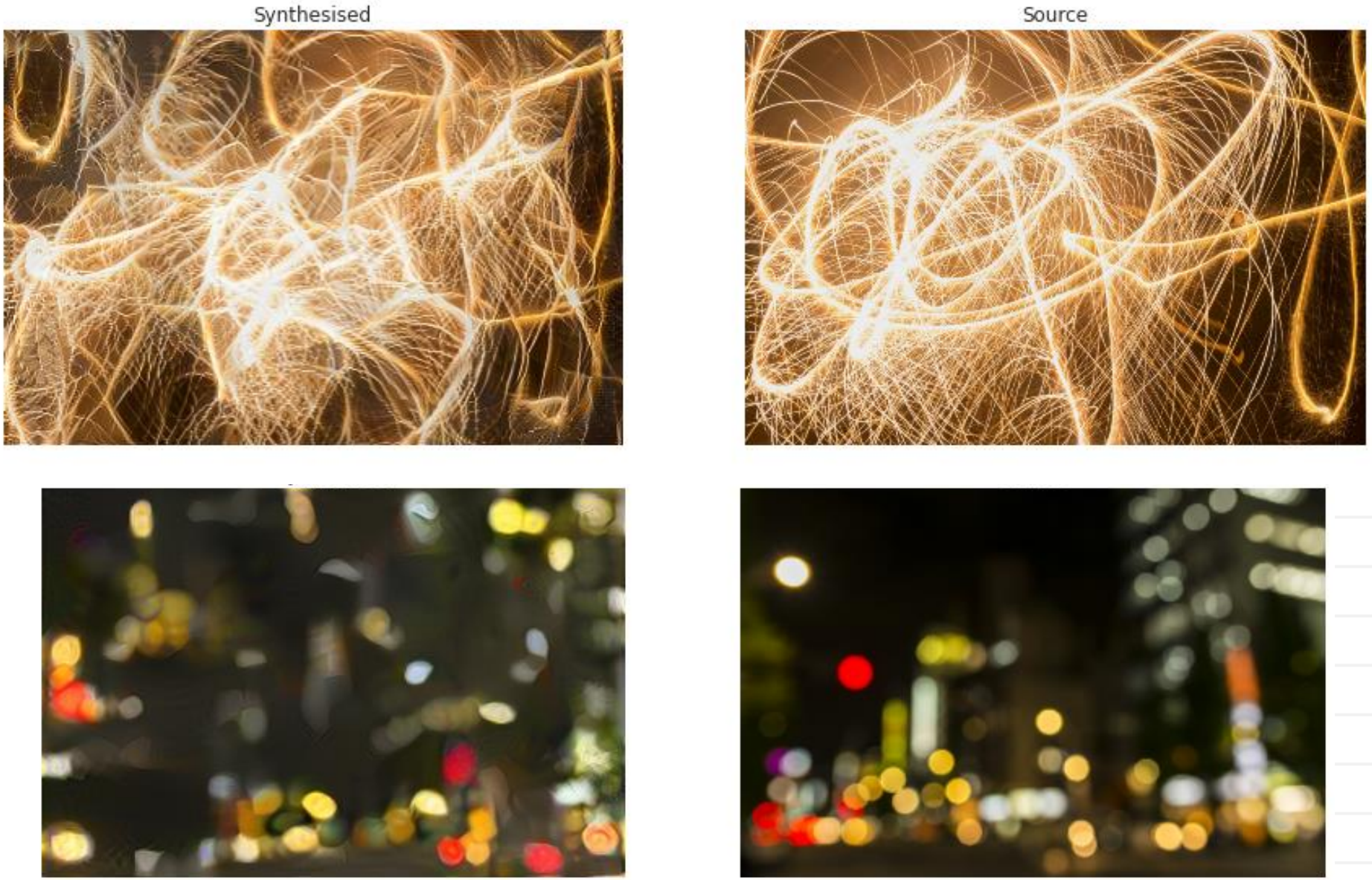
[Gatys, Ecker, Bethge 2015]

Pre-image texture synthesis



[Gatys, Ecker, Bethge 2015]

Pre-image texture synthesis



[Gatys, Ecker, Bethge 2015]

Pre-image texture synthesis

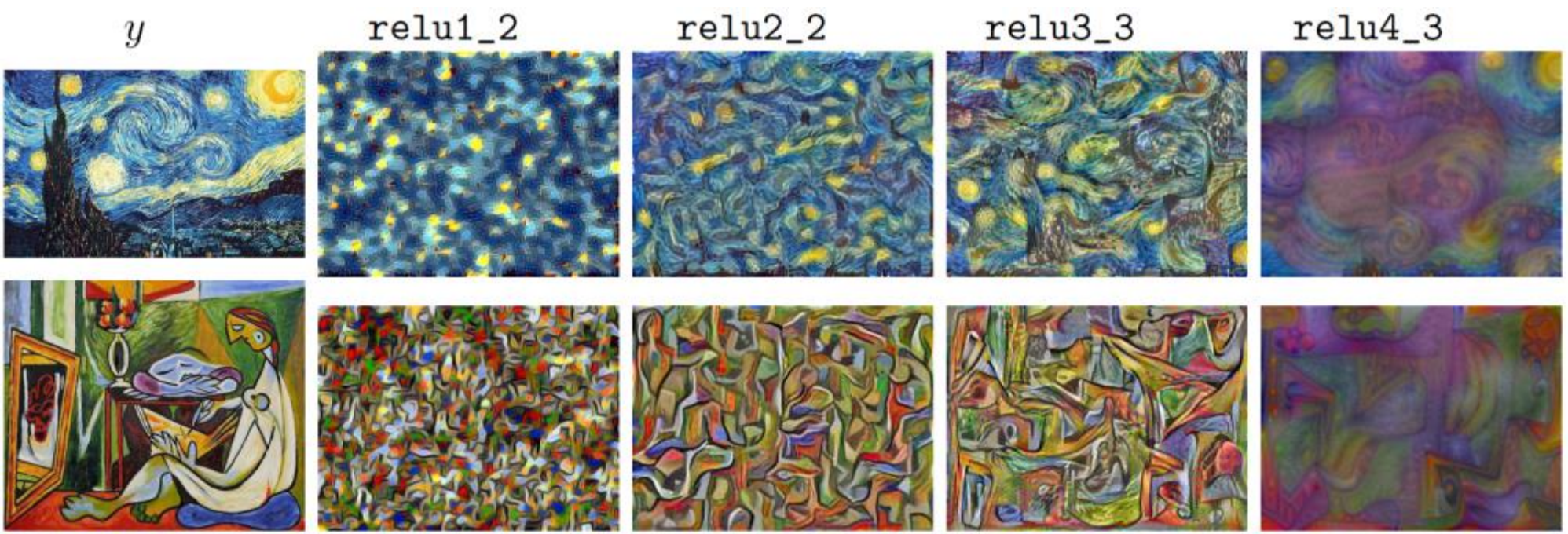
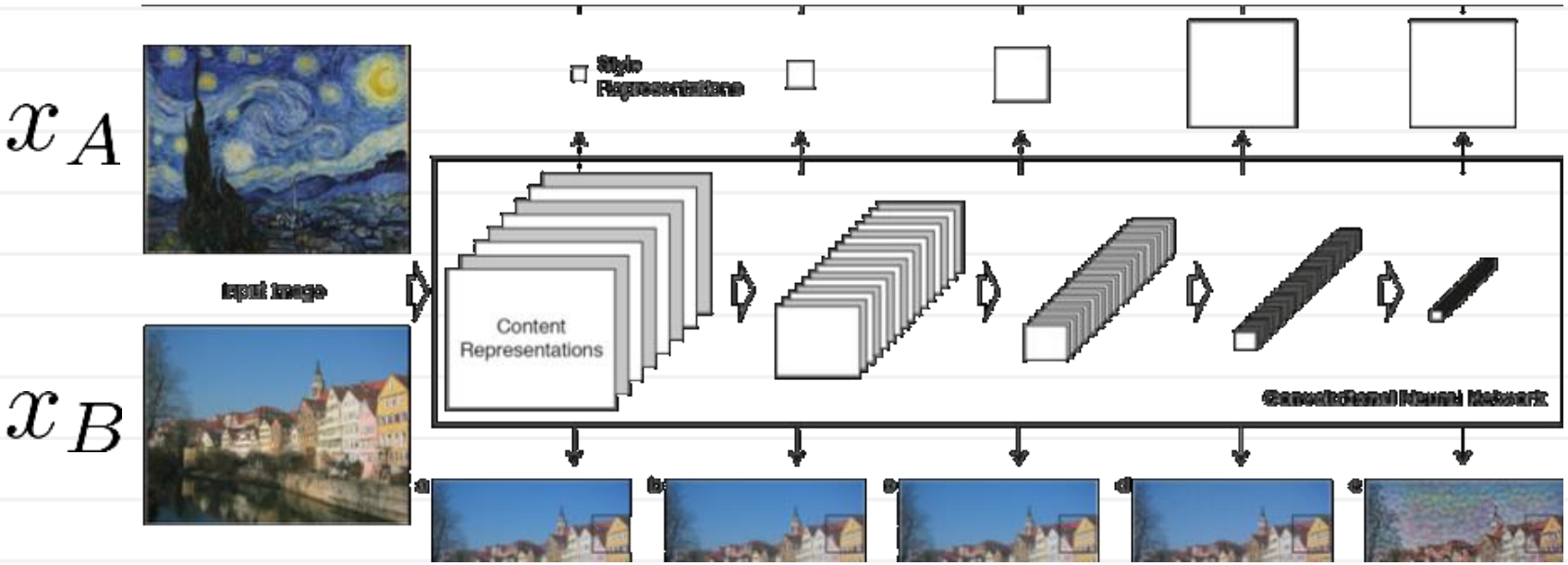


Image from [Johnson et al. ECCV16]

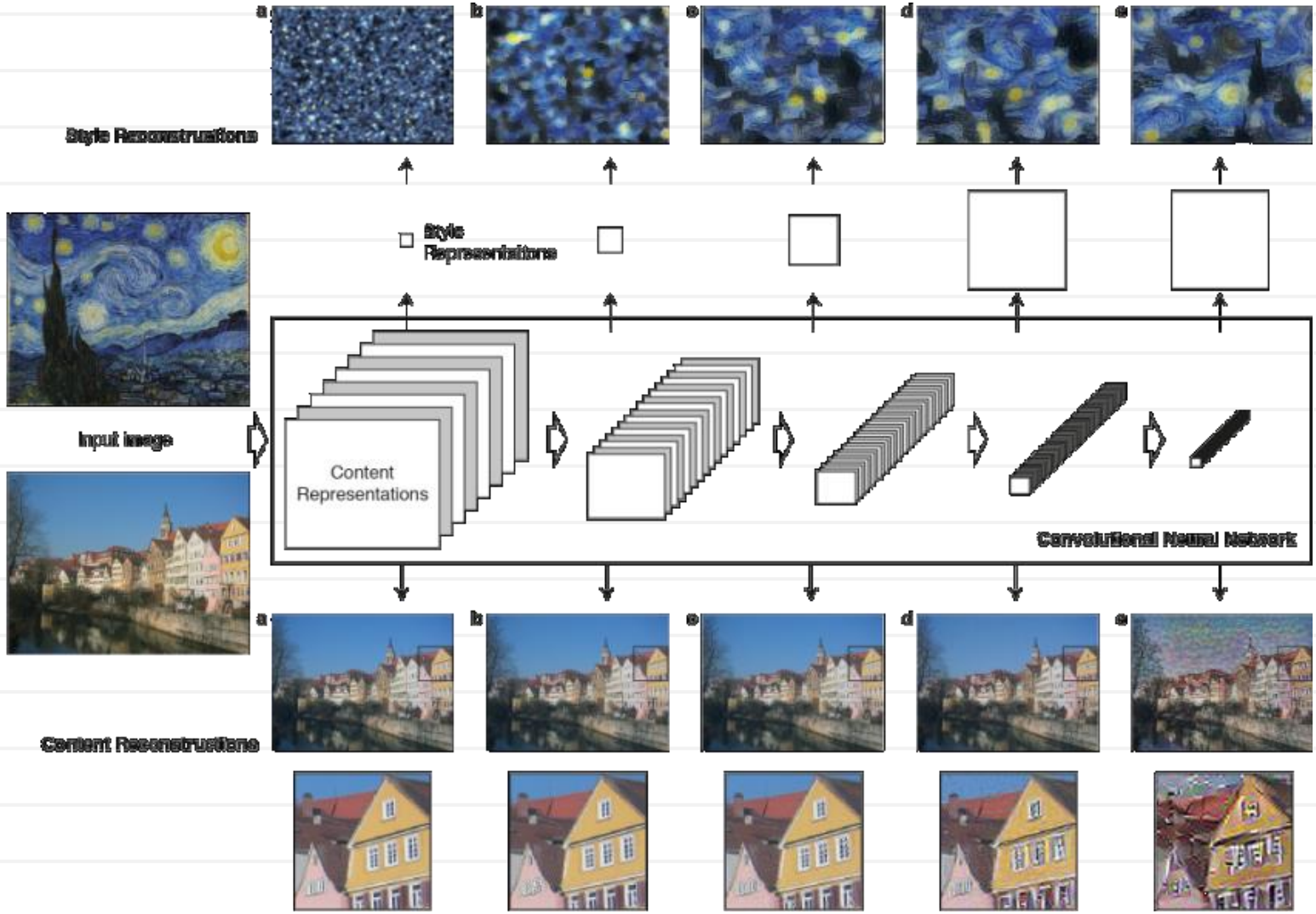
Stylization with texture loss



$$\mathcal{L}(x) = \lambda \sum_{l \in T} \|G^l(x) - G^l(x_A)\|^2 + \sum_{l \in C} \|F^l(x) - F^l(x_B)\|^2$$

[Gatys et al. 2015]

Stylization with texture loss

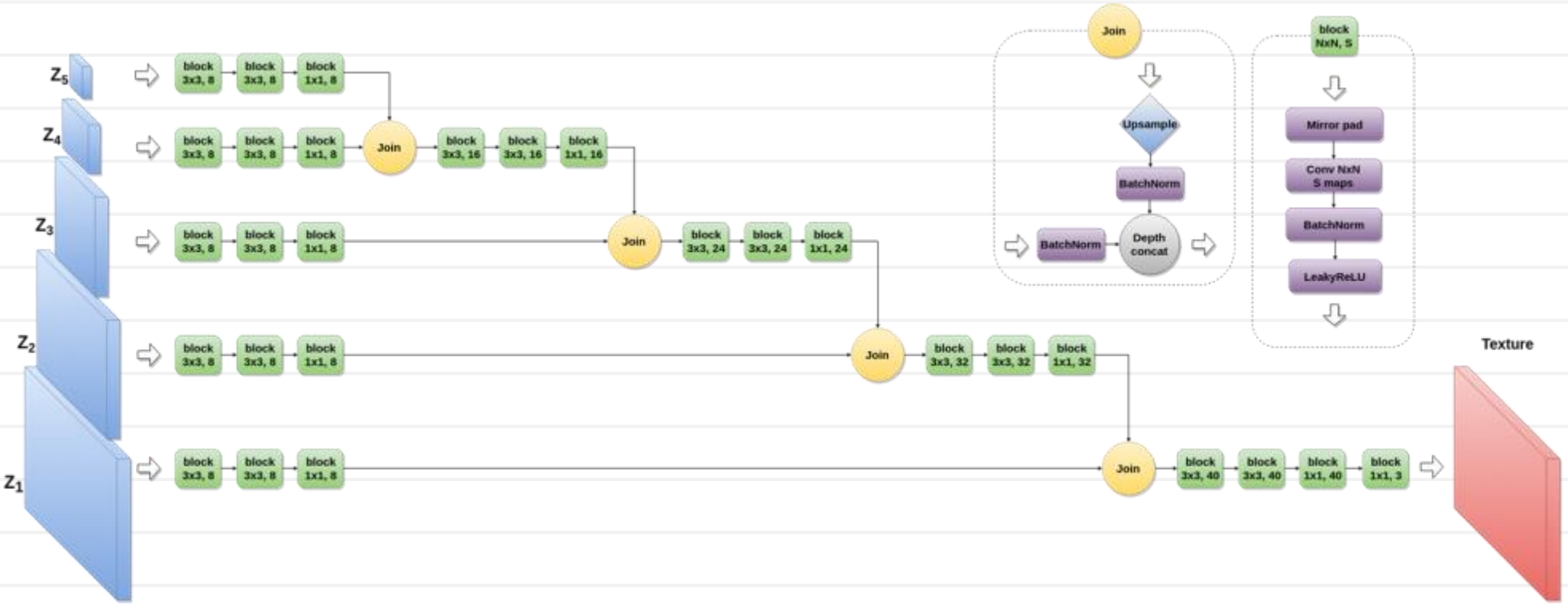


[Gatys et al. 2015]

Neural algorithm for artistic style

[Gatys et al. 2016]

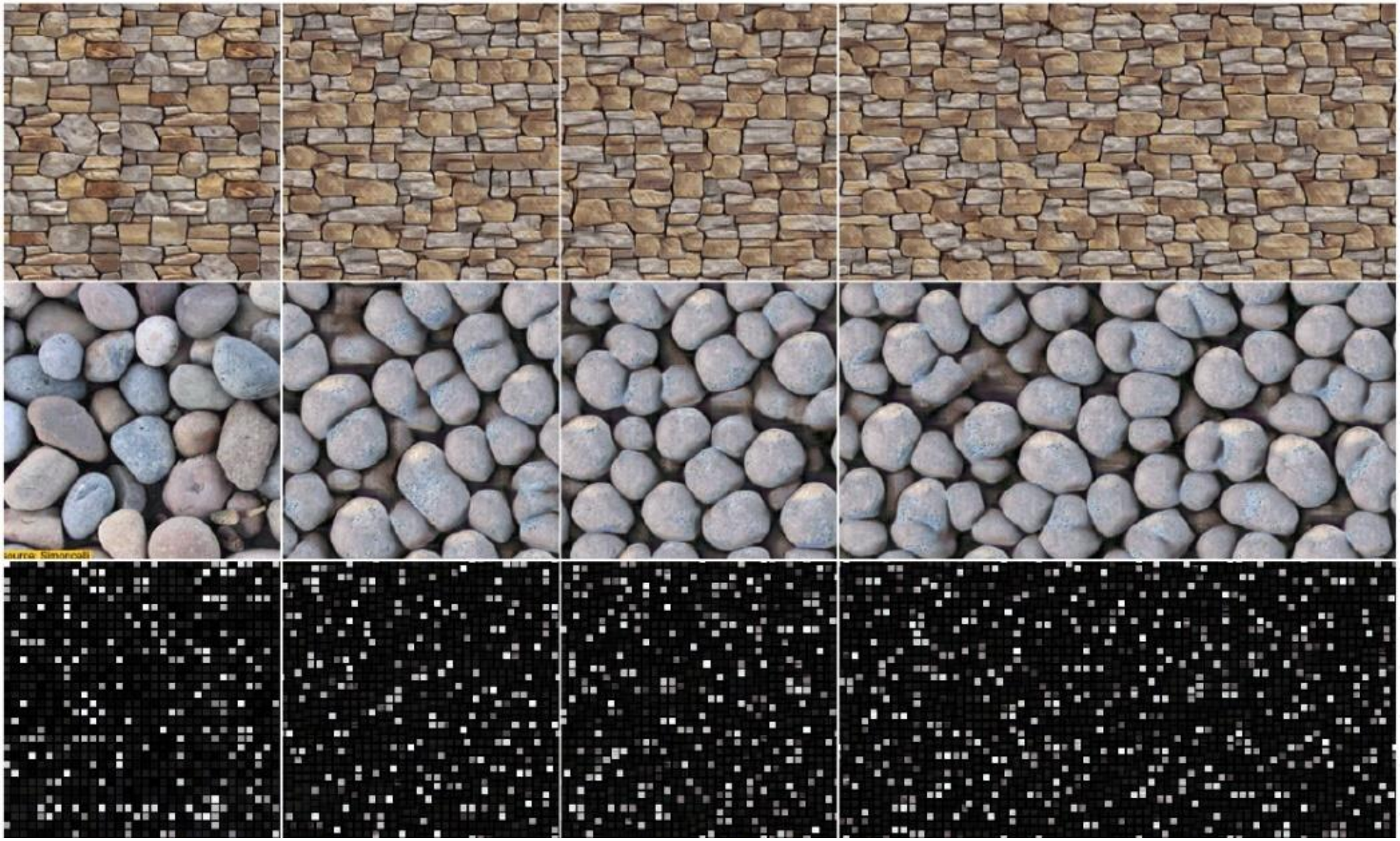
Generator networks details



- Multi-scale approach simplifies learning
- Looks scary, but still feedforward and fully convolutional

[Ulyanov et al. ICML15]

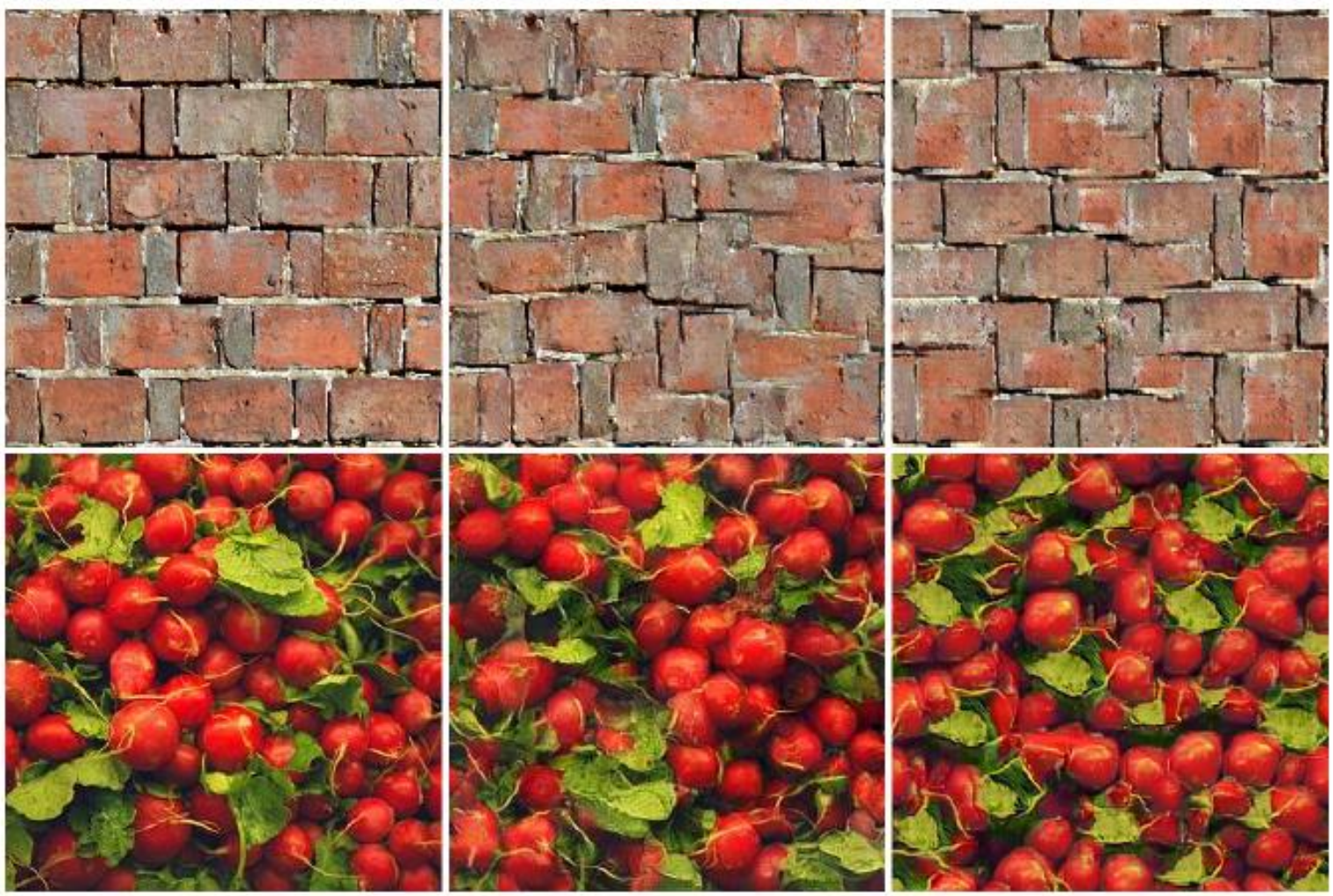
Feed-forward texture synthesis



sample

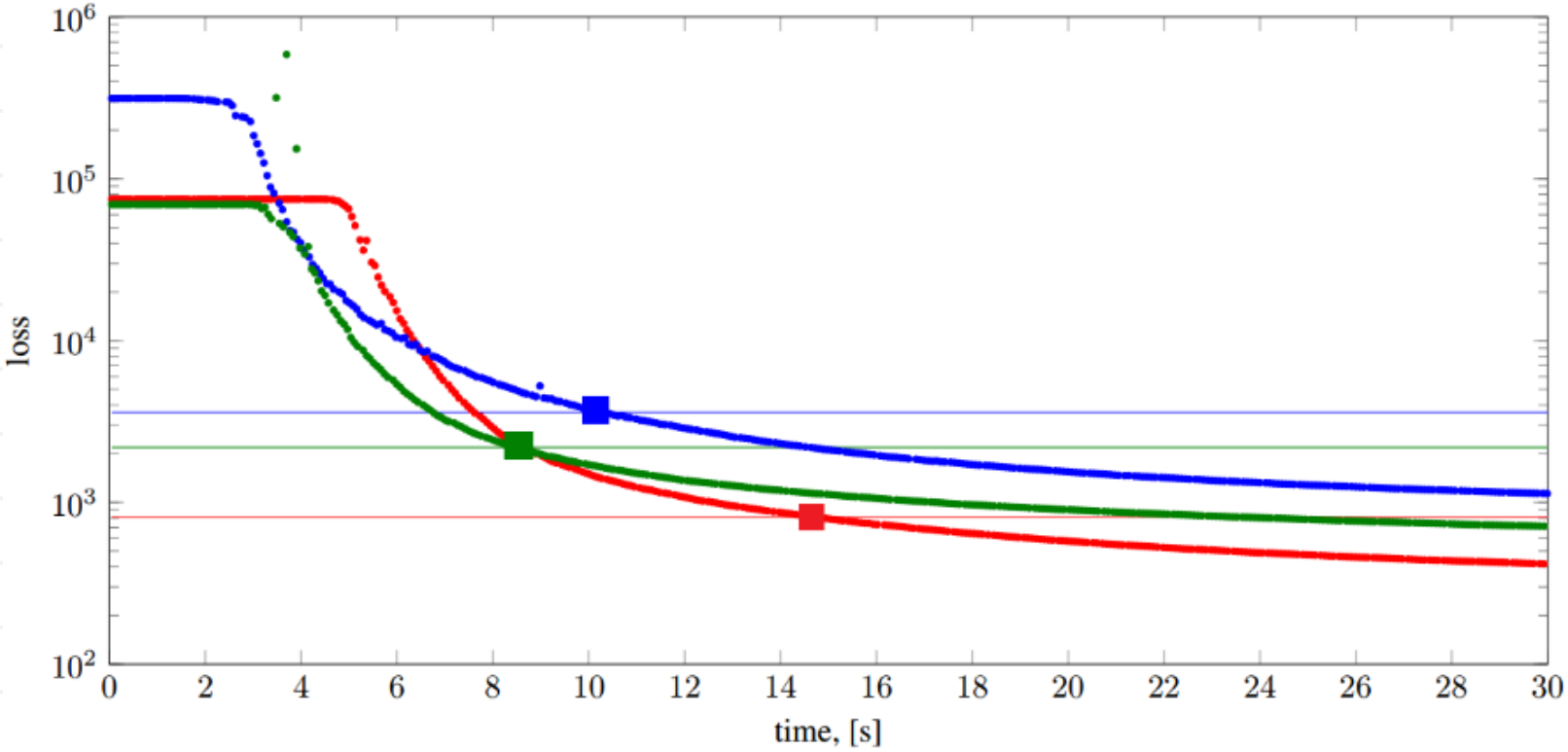
Results

Pre-image vs. feed-forward



sample Pre-image Feed-forward

Feed-forward ConvNet as a "pseudooptimizer"



- Horizontal lines: average sample loss
  - **0.06 second** to generate sample
- Dotted lines: *optimization-based* loss as a function of time
  - **10 seconds** to achieve the same loss values

Feed-forward stylization

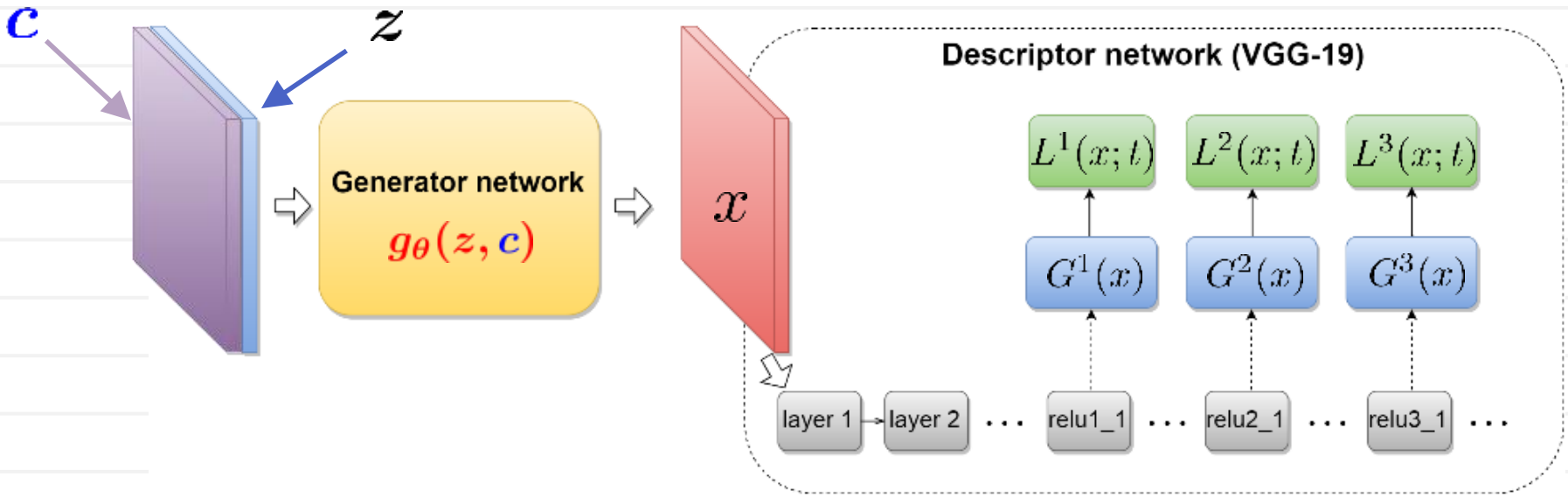


Image generation:

$$x = g_\theta(\boldsymbol{z}, \boldsymbol{c}), \quad \boldsymbol{z} \sim U(0,1)$$

Optimization task:

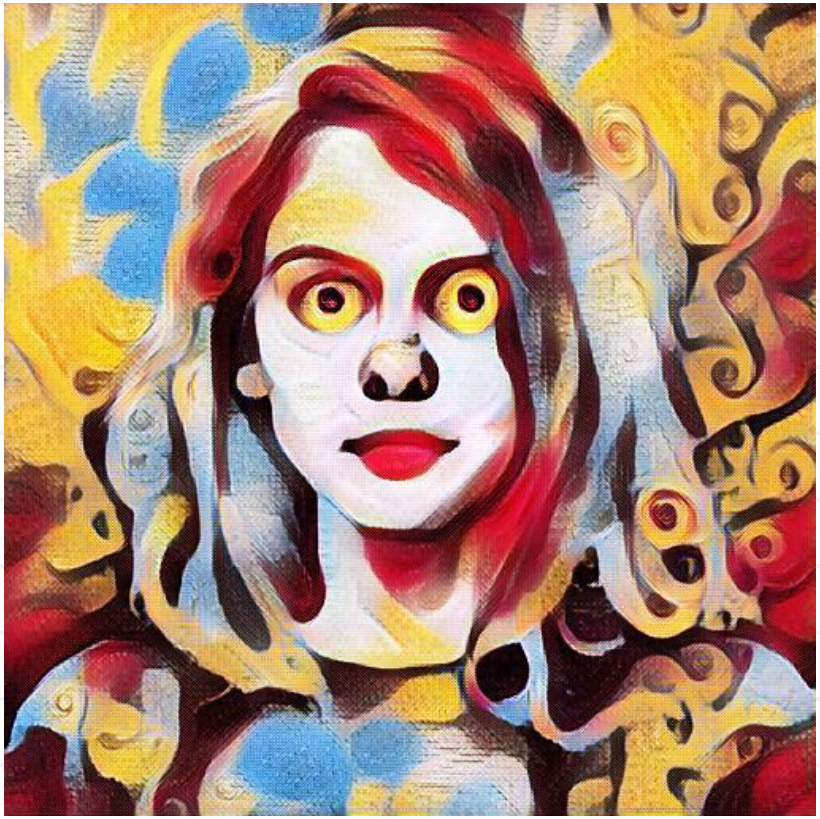
$$\min_\theta \mathbb{E}\mathcal{L}(g_\theta(\boldsymbol{z}, \boldsymbol{c}); c, t), \quad \boldsymbol{z} \sim U(0,1)$$

Feed-forward stylization

feedforward

optimization

Feed-forward stylization



feedforward

optimization

Feed-forward stylization

feedforward

optimization

Bibliography

J Kim, JK Lee, KM Lee, Accurate Image Super-Resolution Using Very Deep Convolutional Networks, CVPR 2016

Alexey Dosovitskiy, Jost Tobias Springenberg, Thomas Brox:
Learning to generate chairs with convolutional neural networks. CVPR 2015: 1538-1546

Dmitry Ulyanov, Andrea Vedaldi, Victor S. Lempitsky:
Deep Image Prior. CVPR 2018

Justin Johnson, Alexandre Alahi, Li Fei-Fei:
Perceptual Losses for Real-Time Style Transfer and Super-Resolution. ECCV (2) 2016: 694-711

Qifeng Chen, Vladlen Koltun:
Photographic Image Synthesis with Cascaded Refinement Networks. ICCV 2017: 1520-1529

Bibliography

Leon A. Gatys, Alexander S. Ecker, Matthias Bethge:
Texture Synthesis Using Convolutional Neural Networks. NIPS 2015: 262-270

Leon A. Gatys, Alexander S. Ecker, Matthias Bethge:
Image Style Transfer Using Convolutional Neural Networks. CVPR 2016: 2414-2423

Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, Victor S. Lempitsky:
Texture Networks: Feed-forward Synthesis of Textures and Stylized Images. CoRR abs/1603.03417 (2016)