Neural Network in Computer Graphics

“Reveal the order of the world where top-down meets bottom-up”

Liqian Ma
Megvii (Face++) Researcher
maliqian@megvii.com
Nov 2017

Raise your hand and ask,
whenever you have questions...

Outline

- Graphics overview
- NN in graphics
  - NN for rendering
  - NN for 3D modeling
  - NN for visual media retouching
- Example: NN 3D face
- Rendering for CV applications

Graphics - rendering

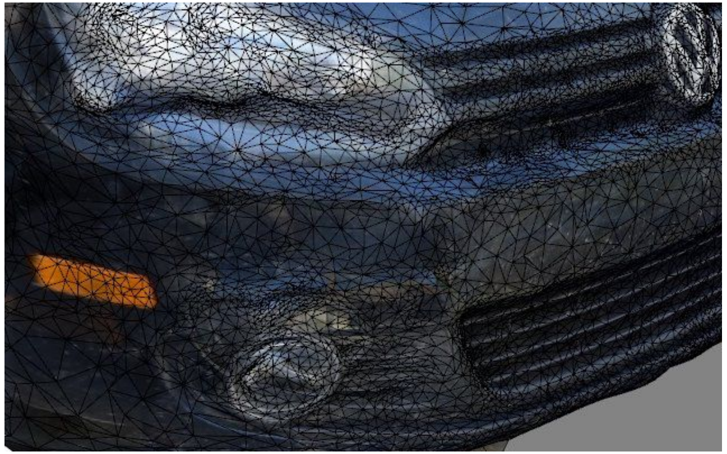
- Mission: reveal the order of lights/optics that formulates this colorful, full-shading world

- Keyword: Ray tracing, photon mapping, real-time rendering, environment lighting, Phong, reflection, refraction, Bidirectional Reflectance Distribution Function, physics simulation, VR, AR, …

Graphics – 3D modeling

- Mission: play with a low rank presentation for describing real world 3D

- Keyword: mesh, geometry, voxel, triangulation, point cloud, 3D reconstruction, shape from motion, Stanford bunny, Earth mover's distance, Laplacian deformation, hair modeling, registration, edge collapse, …



Graphics – visual media retouching

- Mission: enjoy the tricks on visuals to fake human eye/brain out

- Keyword: camera, image signal processing, perception, artifact, segmentation, propagation, diffusing, composition, matting, blending, stylization, super-resolution, deblur, computational photography, …

Challenges in Graphics

- Rendering with tons of complex detailed formulation challenges frontend/backend hardware.
- 3D modeling: human brain may not be able to identify the low-dimensionity
- Visual media retouching: a lot of effects can not be well formulated by human brain

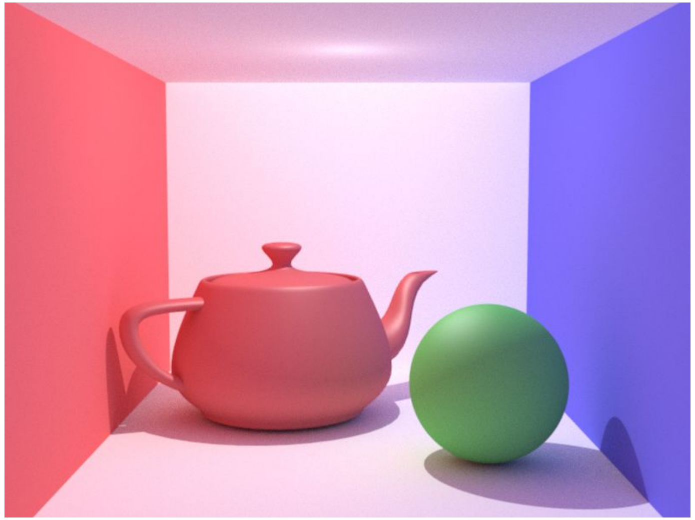
Neutral Network is comming

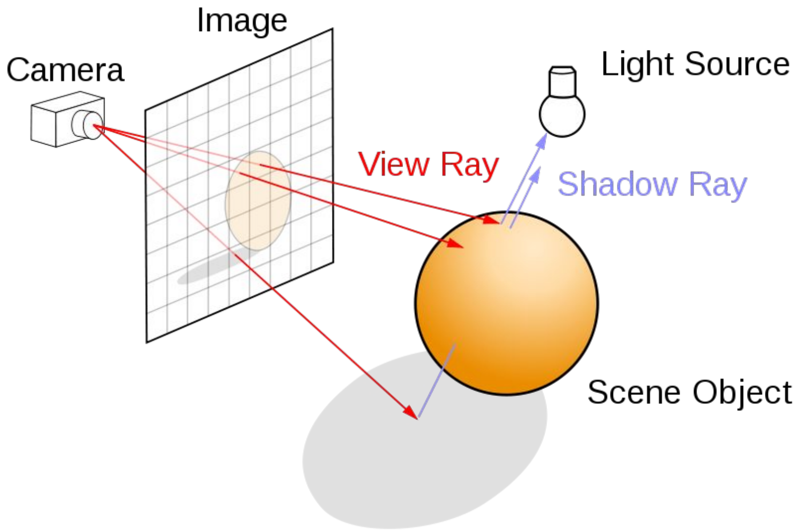
Neural Network for graphics

- Neural Network (NN) can
  - Faster, better and more robust than human-written equations
  - Handle data with very high dimensions
  - Explore the low-rank characteristics of a problem and formulate it, which is difficult to formulize by human brain

NN Rendering - Monte Carlo ray tracing

- An offline rendering technique, simulate large amount of ray using Monte Carlo sampling, and calculate the radiance of each ray, accumulate for each pixel.





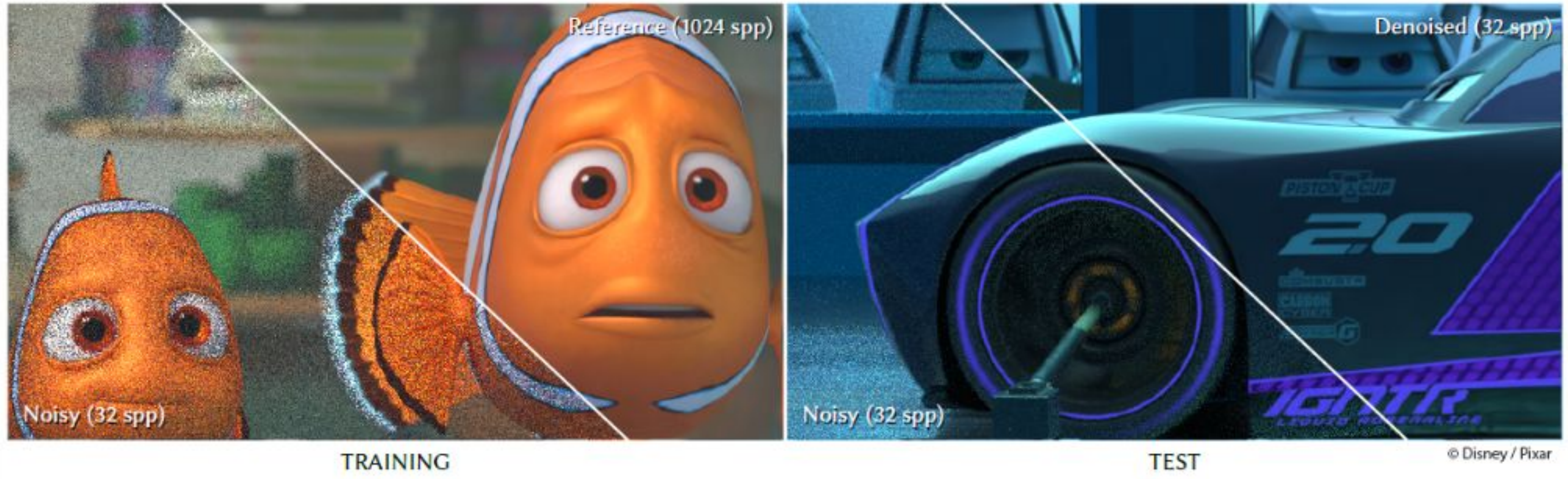
Face++ 旷视

NN Rendering - Monte Carlo ray tracing

- An offline rendering technique, simulate large amount of ray using Monte Carlo sampling, and calculate the radiance of each ray, accumulate for each pixel.

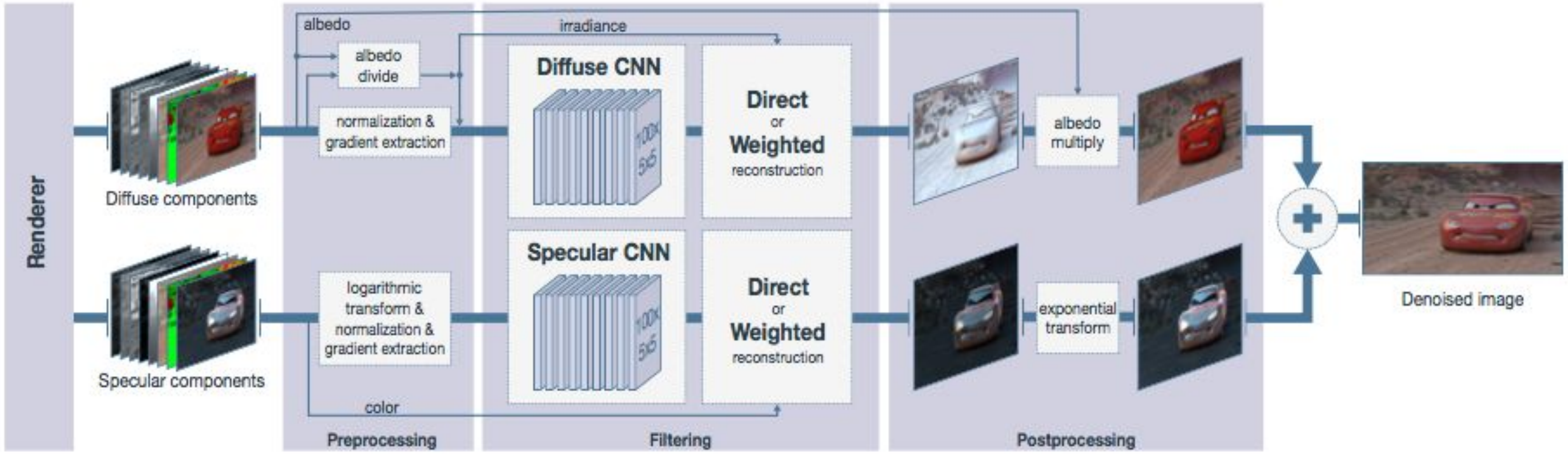
- weakness: having more ray would reduce image noise, at the expense of slower computation.

- [SIGGRAPH17] Kernel-Predicting Convolutional Networks for Denoising Monte Carlo Renderings



NN Rendering - Monte Carlo ray tracing

- [SIGGRAPH17] Kernel-Predicting Convolutional Networks for Denoising Monte Carlo Renderings
- Utilize CNN to predict de-noising kernels, thus enhance ray tracing rendering result.



NN Rendering – Volume rendering

- Volume rendering: simulate scattering effect, light would scatter at any point of the volume.
- Traditional: complex integral over scattering function and ray, quite slow:

scattering function

radiance of other direction

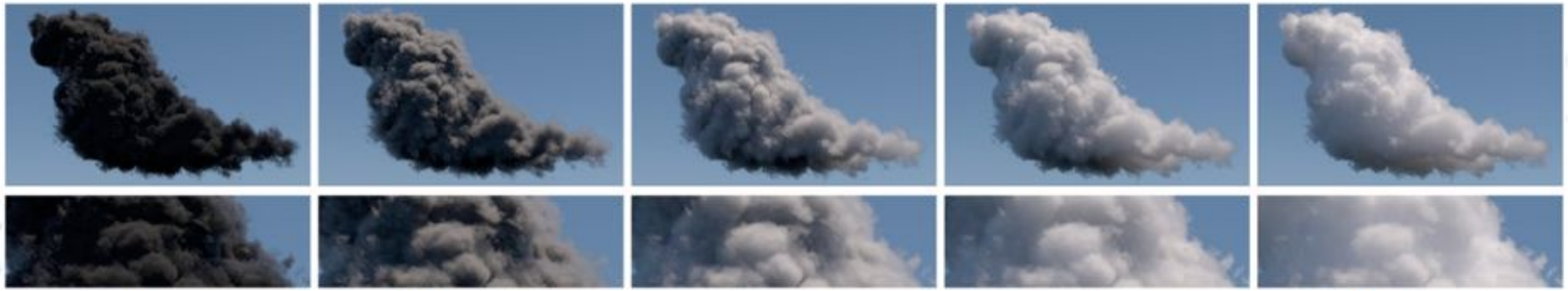
$$(\omega \cdot \nabla)L(\mathbf{x}, \omega) = -\,\mu_t(\mathbf{x})L(\mathbf{x}, \omega) + \mu_s(\mathbf{x}) \int_{S^2} p(\omega \cdot \widehat{\omega})L(\mathbf{x}, \widehat{\omega})\, \mathrm{d}\widehat{\omega},$$

radiance increasement along specific direction

radiance decay

scatter coeff

radiance gathered from other direction



(a) 1 interaction Render time = 0.017 h
(b) 4 interactions Render time = 0.89 h
(c) 16 interactions Render time = 6.5 h
(d) 64 interactions Render time = 16 h
(e) All interactions Render time = 34 h

NN Rendering – Volume rendering

- Deep Scattering: Rendering Atmospheric Clouds Radiance-Predicting Neural Networks (2017)
- Use CNN to pre-train the complex integral.
- 200x faster.

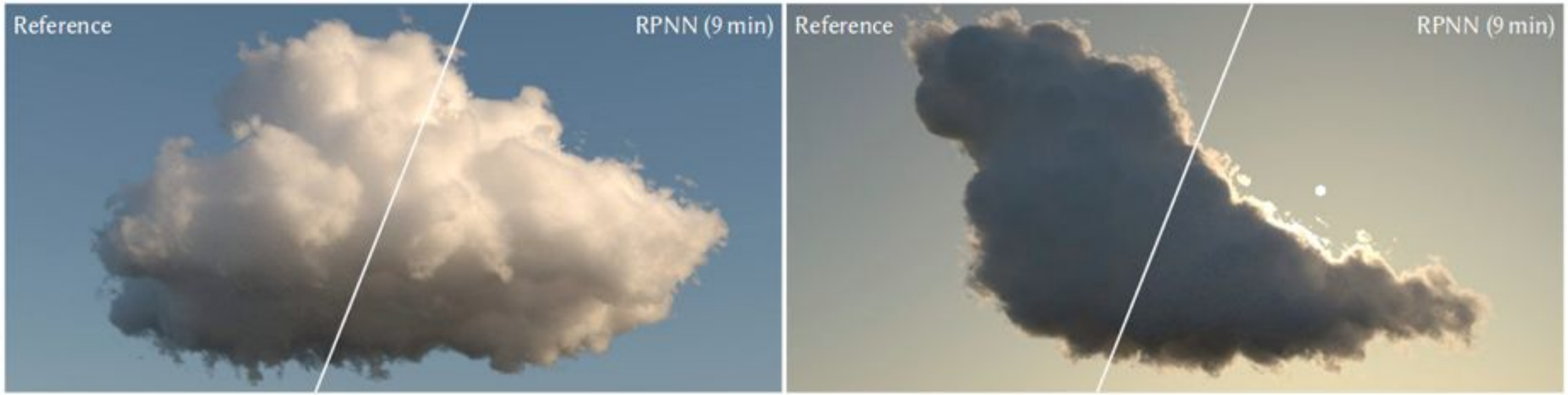
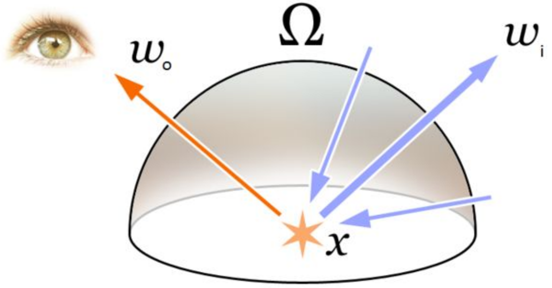


Fig. 1. We synthesize multi-scattered illumination in clouds using deep radiance-predicting neural networks (RPNN). We combine Monte Carlo integration with data-driven radiance predictions, accurately reproducing edge-darkening effects (left), silverlining (right), and the whiteness of the inner part of the cloud.

NN rendering – NN shading

- Real-time rendering
    - design various fast approximations of the famous Rendering Equation:



$$L_o(\mathbf{x}, \omega_o, \lambda, t) = L_e(\mathbf{x}, \omega_o, \lambda, t) + \int_\Omega f_r(\mathbf{x}, \omega_i, \omega_o, \lambda, t)\, L_i(\mathbf{x}, \omega_i, \lambda, t)\, (\omega_i \cdot \mathbf{n})\, \mathrm{d}\,\omega_i$$

outgoing radiance | emission | integral over all incoming direction | BRDF | incoming radiance | cosine

x: the location in space
ω_o: the direction of the outgoing light
λ: a particular wavelength of light
t: time

NN rendering – NN shading

- Deep shading: Convolutional Neural Networks for Screen-space shading (2016)

- Use nn to directly learn complex BRDF shading equations: normal map/position map/reflectance map -> color

- Shading calculation can be 10x+ faster.

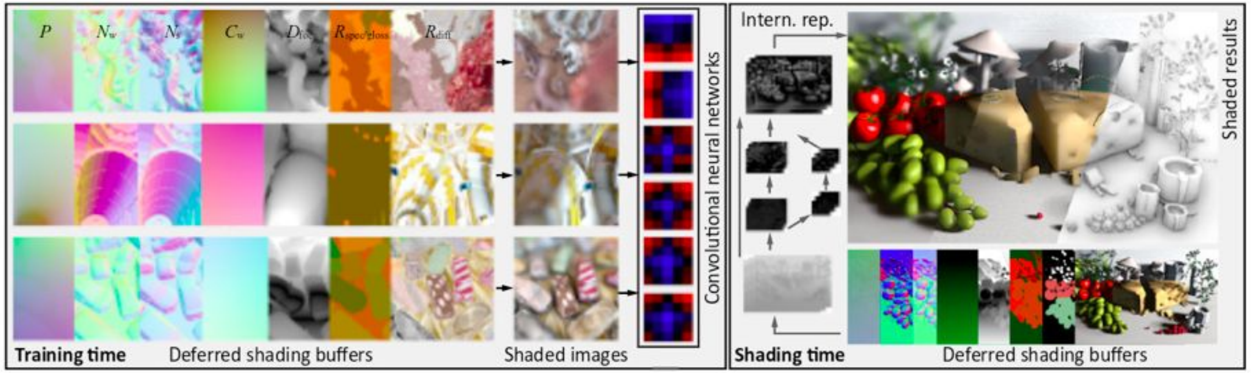


Figure 1: *In training* (left), *our approach learns a mapping from attributes in deferred shading buffers, e. g., positions, normals, reflectance, to RGB colors using a convolutional neural network (CNN). At run-time* (right), *the CNN is used to produce effects such as depth-of-field, sub-surface scattering or ambient occlusion at interactive rates (768×512 , px 1 ms rasterizing attributes, 21 /21/ 17 ms network execution).*

NN rendering – takeaway

- NN training to accelerate rendering is a trend, with visually equal rendering quality, rendering can be 10~1000x faster.
- If you want, all training data can be gathered virtually, no need to collect real data.
- Currently NN is only capable to handle domain-specific render tasks.

NN 3D modeling – shape understanding

- Find a fix-sized presentation of arbitrary 3D mesh(point cloud, octree, voxel..) , throw it to NN

- tricks are: presentation should be x-invariant

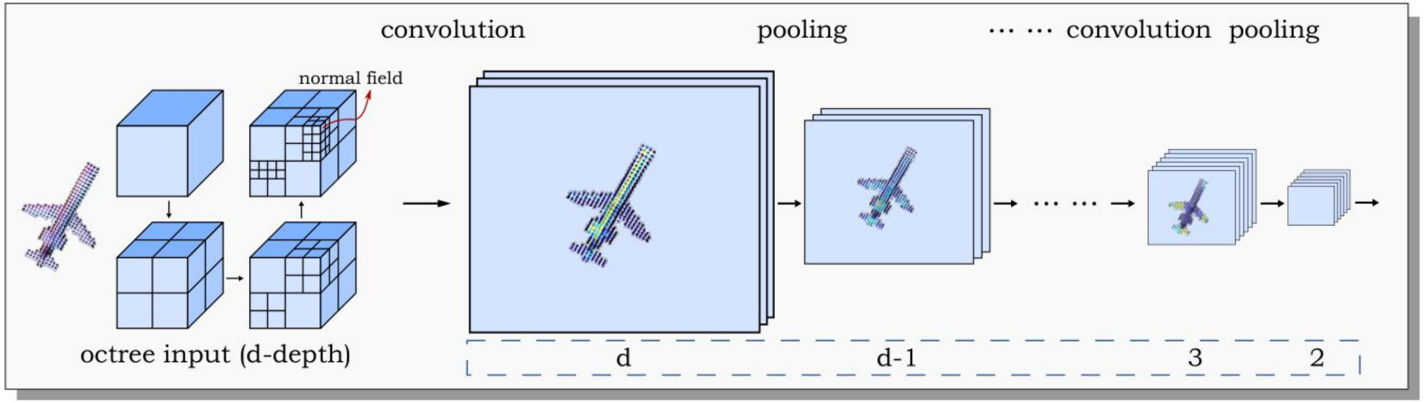


Fig. 1. An illustration of our octree-based convolutional neural network (O-CNN). Our method represents the input shape with an octree and feeds the averaged normal vectors stored in the finest leaf octants to the CNN as input. All the CNN operations are efficiently executed on the GPU and the resulting features are stored in the octree structure. Numbers inside the blue dashed square denote the depth of the octants involved in computation.

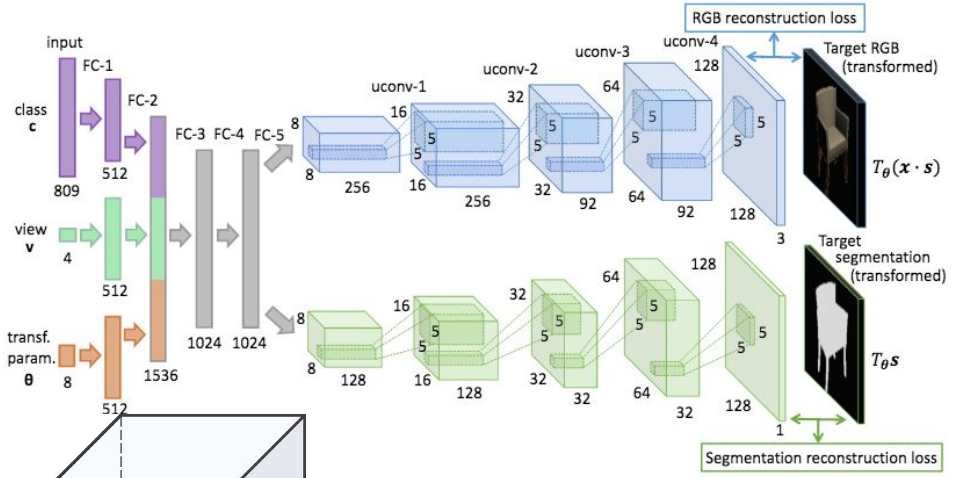
NN 3D modeling – shape understanding

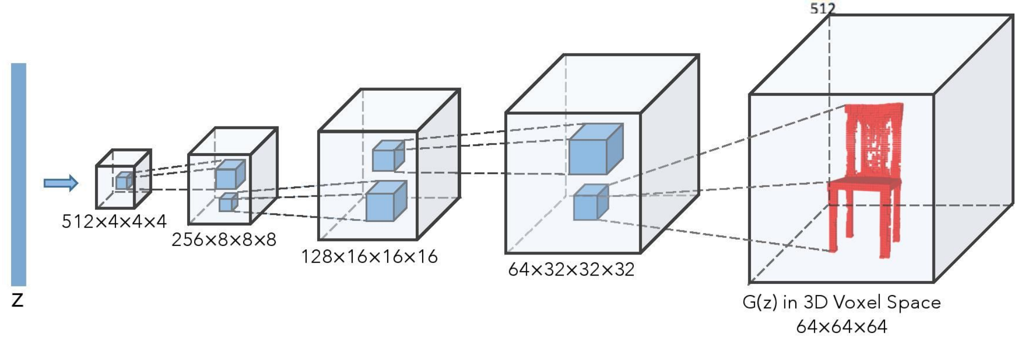
- 3D ShapeNets: A Deep Representation for Volumetric Shapes (2015)
- VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition (2015)
- DeepPano: Deep Panoramic Representation for 3-D Shape Recognition (2015)
- FusionNet: 3D Object Classification Using Multiple Data Representations (2016)
- OctNet: Learning Deep 3D Representations at High Resolutions (2017)
- O-CNN: Octree-based Convolutional Neural Networks for 3D Shape Analysis (2017)
- Orientation-boosted voxel nets for 3D object recognition (2017)
- PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation (2017)

NN 3D modeling – shape synthesis

Learning to Generate Chairs, Tables and Cars with Convolutional Networks (2014)

- 3d-conv, 3d-deconv,
- can also use GAN
- Training 3D NN is difficult.





3D GAN: Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling (2016)

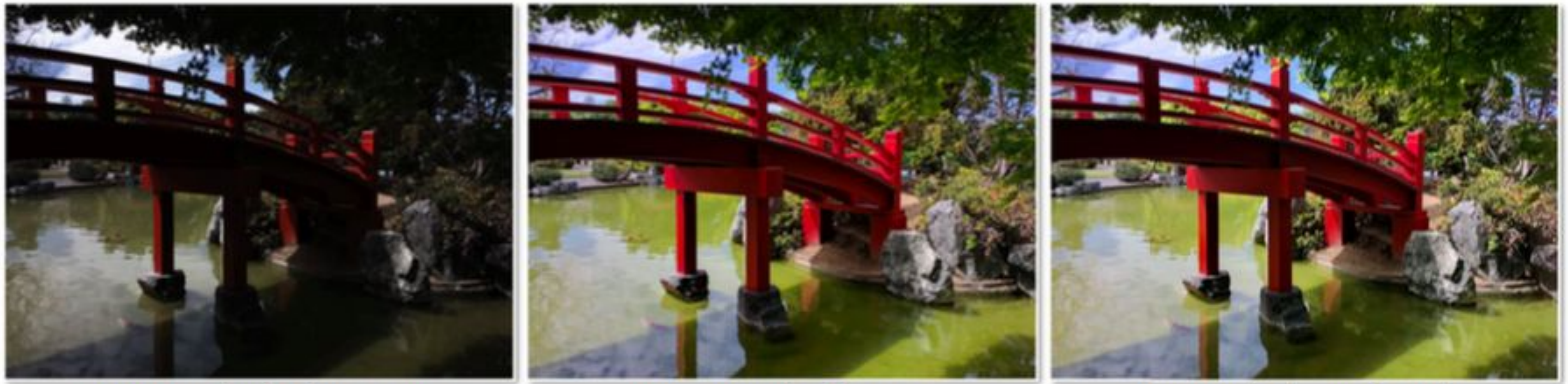
NN 3D modeling - takeaway

- From 2d to 3d, data are exponentially harder to handle, even for nn.
- The design of mesh presentation is the trick but the key to success.
- High-resolution 3D problems are still a very hard problem, nn helps little.

NN visuals retouching – tone mapping

- Tone-mapping: compute a per-pixel bilateral coefficient for adjusting, aiming to get HDR from LDR

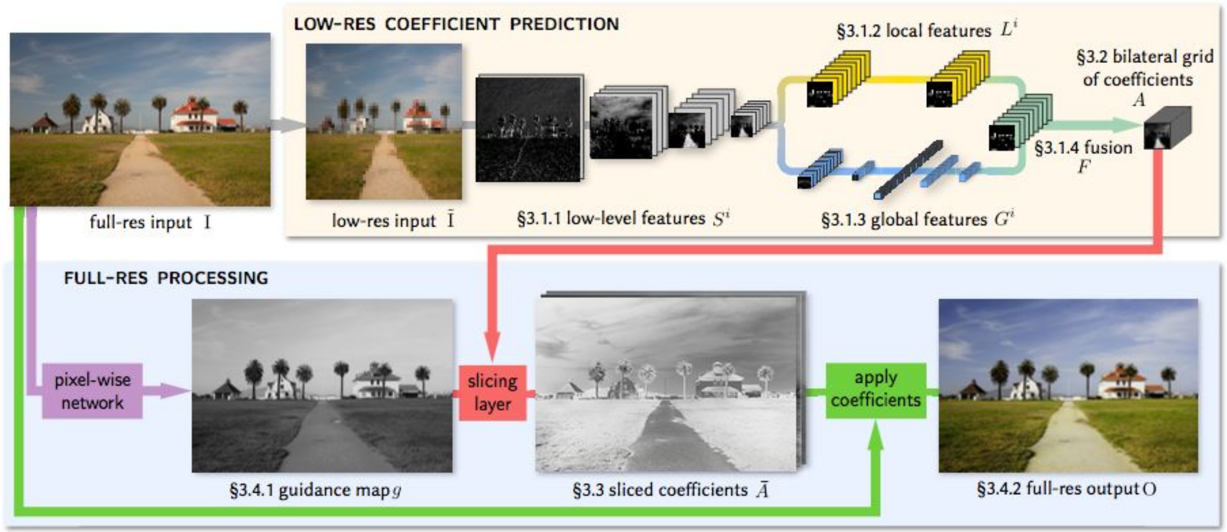
- Deep Bilateral Learning for Real-Time Image Enhancement (2017)



12 megapixel 16-bit linear input (tone-mapped for visualization) | tone-mapped with HDR+ **400 – 600 ms** | processed with our algorithm **61 ms**, PSNR = **28.4 dB**

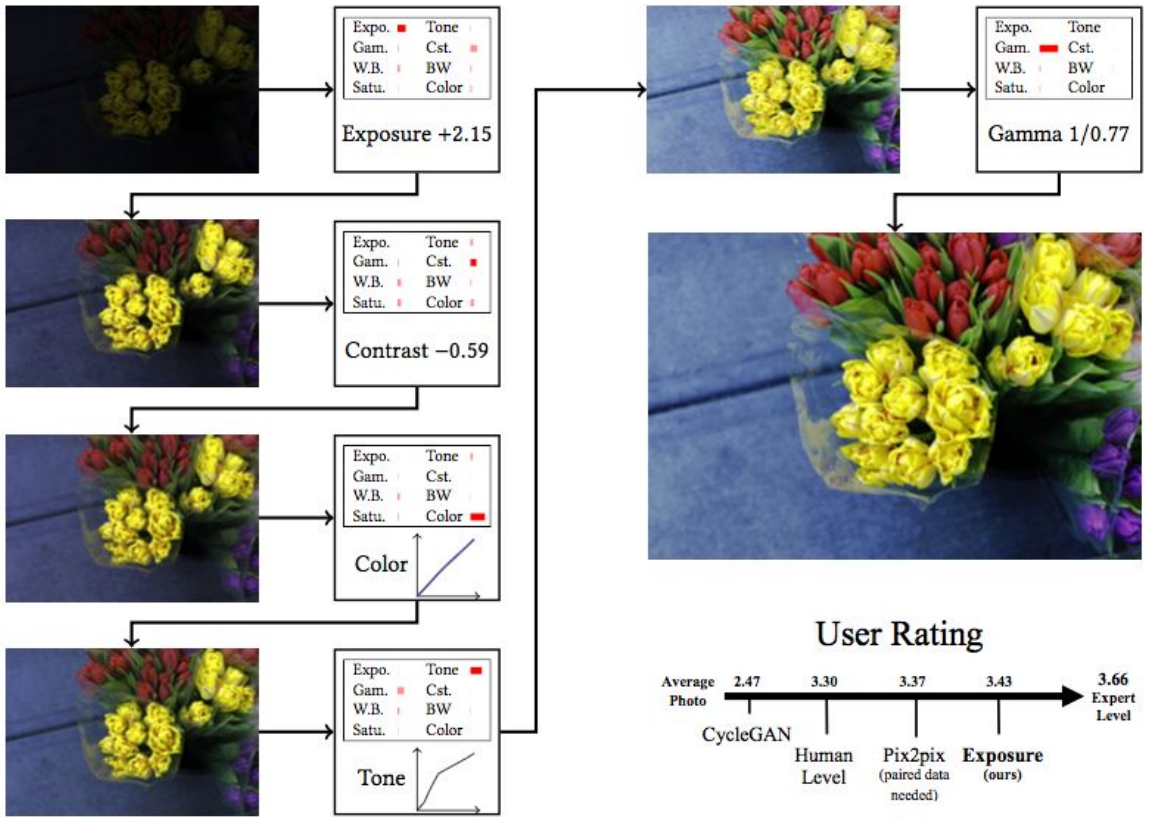
NN visuals retouching – tone mapping

- Low resolution stream: extract as much info as possible
- High resolution stream: pixel-wise computing
- This paper provides a strategy that can handle high-resolution images relatively fast.


LOW-RES COEFFICIENT PREDICTION
§3.1.2 local features L^i
§3.2 bilateral grid of coefficients A
§3.1.4 fusion F
full-res input I
low-res input $\tilde{I}$
§3.1.1 low-level features S^i
§3.1.3 global features G^i
FULL-RES PROCESSING
pixel-wise network
slicing layer
apply coefficients
§3.4.1 guidance map g
§3.3 sliced coefficients $\bar{A}$
§3.4.2 full-res output O

NN visuals retouching – automatic enhancement

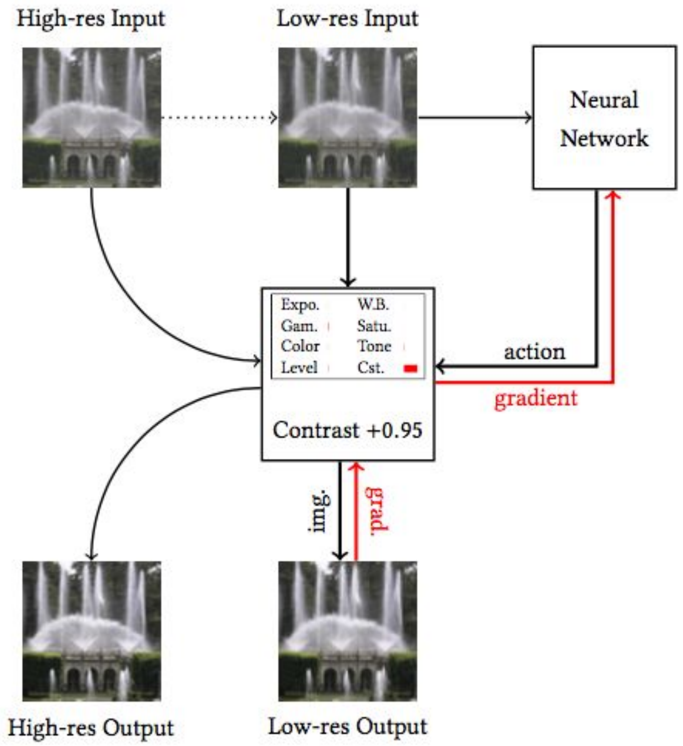
- Exposure: A white-box Photo Post-processing Framework (2017
- Automatically propose editing history for arbitrary image, to improve image quality



NN visuals retouching – automatic enhancement

- Data: a set of images with good degree of impression, manually adjusted

- Use a discriminator to measure whether the auto-generated images are close to the pre-adjusted set of images.



NN visuals retouching - takeaway

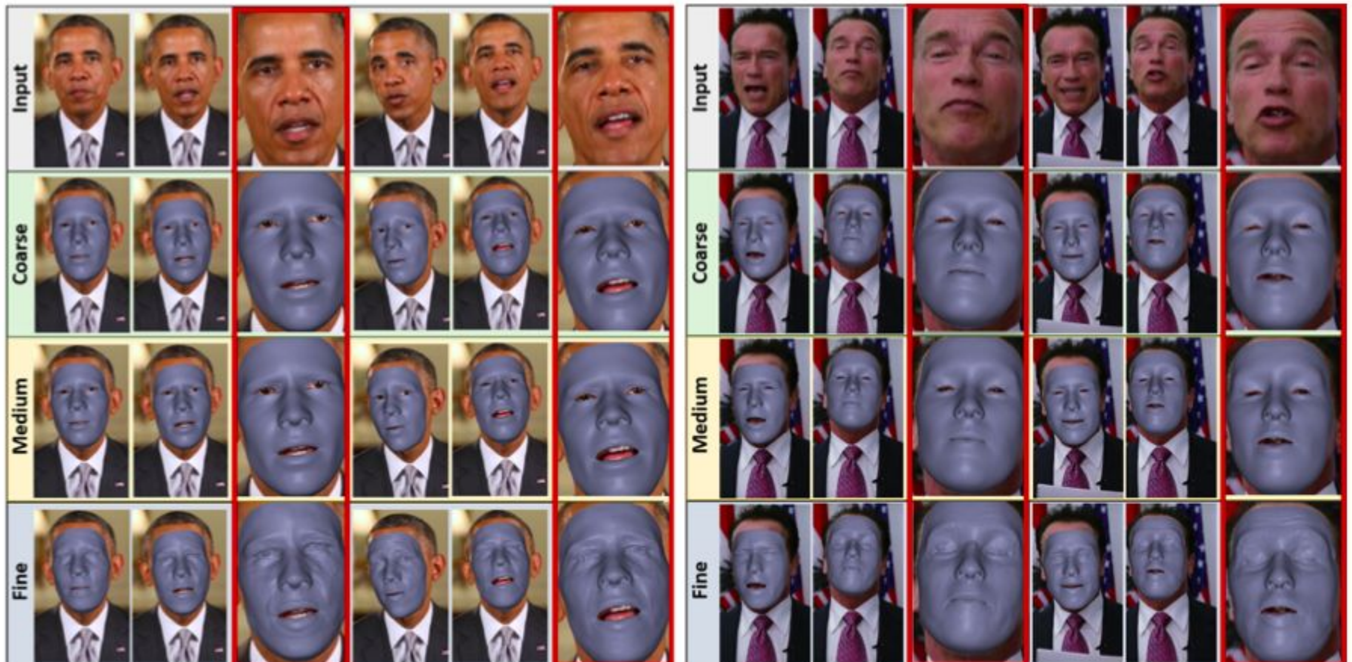
- FCN and GAN like networks are popular for visuals retouching, and works great.
- End-to-end networks here are still not suitable for mobile real-time applications currently, try to find an intermediate presentation.

NN 3D Face

“Towards face recognition by (x, y, z).”

NN 3D Face

- NN 2D Face : detect face region and recognize faceid from 2D visuals
- NN 3D Face : detect face 3D shape and recognize faceid from 3D shape

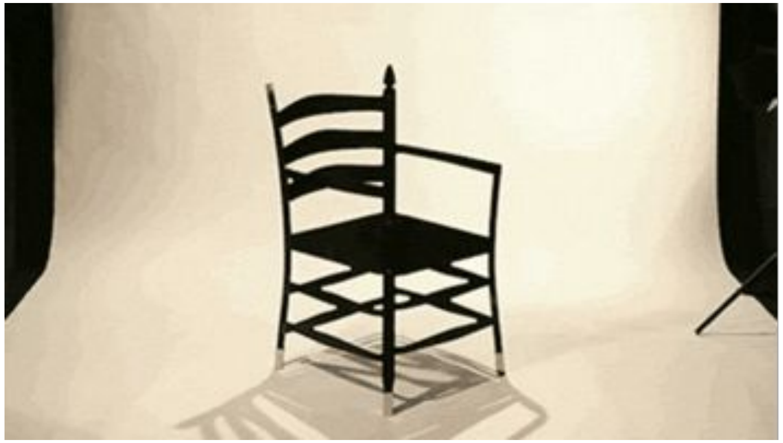
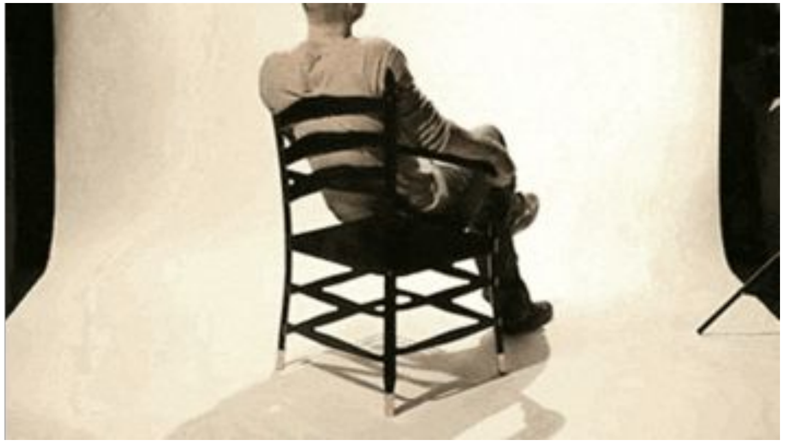


NN 3D Face

- Data source: monocular RGB/RGBD stills/sequences
- Given a face RGB/RGBD still/sequence, reconstruct for each frame:
  - Inner/outer camera matrix
  - Face 3D pose
  - Face shape
  - Face expression
  - Face albedo
  - lighting
- This kind of problem is also called intrinsic image or inverse rendering.

NN 3D Face

- General 3D reconstruction problem from visuals is still very hard due to depth ambiguity.

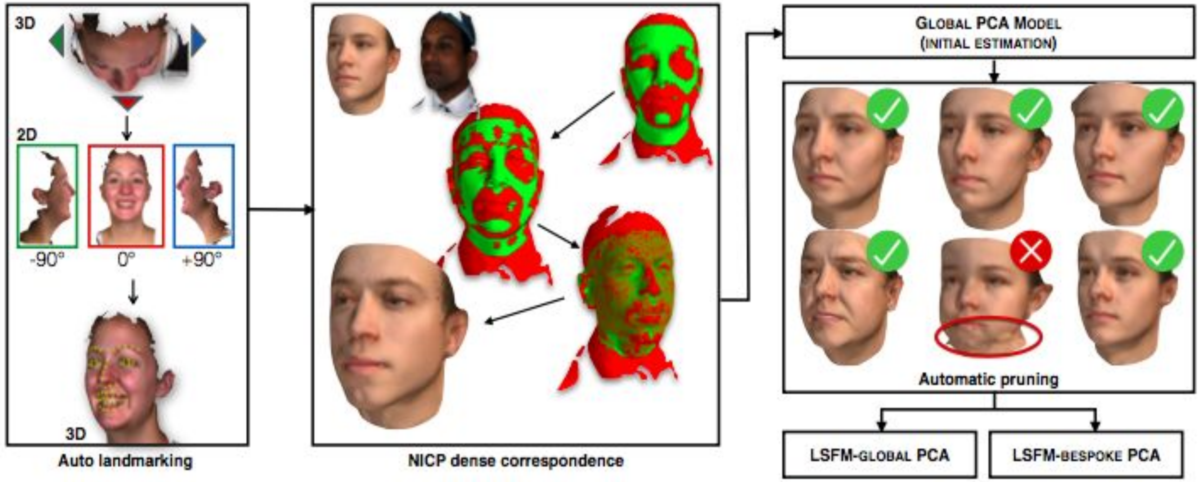



NN 3D Face – Good news

- Face has strong priors
  - Face shape and expression have a relatively small rank
  - Face albedo also lies in a relatively small subspace
  - Face detection and landmarking can be done in 2D
  - Face BRDF have strong priors
- Shading of environment lighting often changes softly across face
  - good starting point of optimization.
  - RGB reveals shape details(wickles..) and lighting (think about rendering equation).
- Face shape is the same across all frames in a face track
- For pre-calibrated cameras, inner matrix are known.

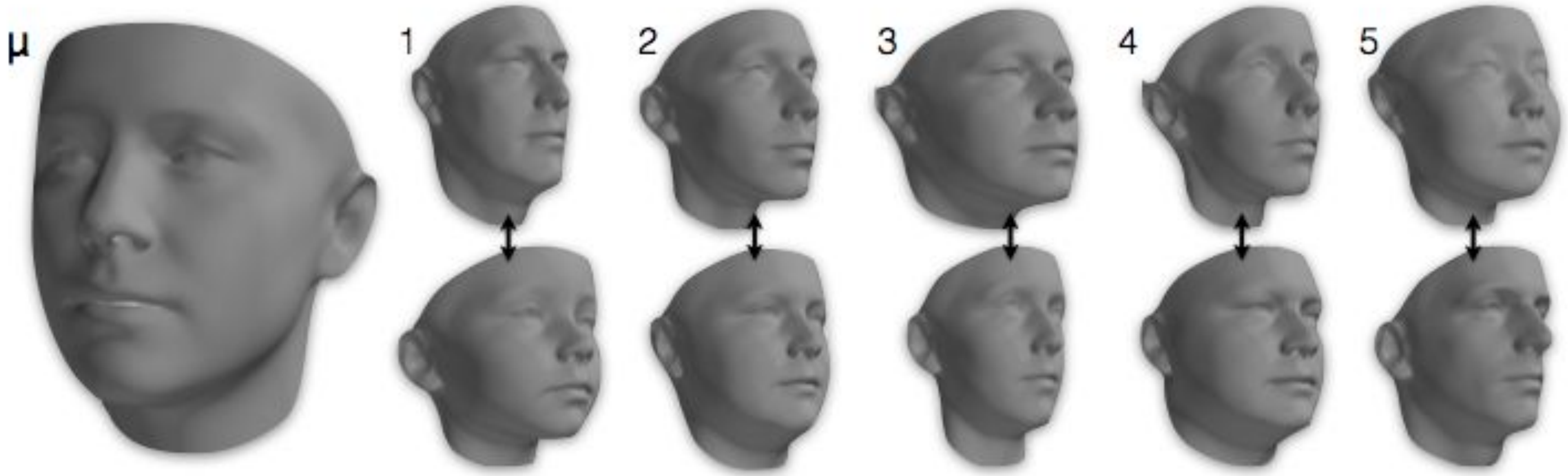
3D Face priors – shape & albedo

- A 3D Morphable Model learnt from 10,000 faces (2016)
- step 1: Capture multi-view face image using camera array
- step 2: 3D reconstruction
- step 3: compute NICP dense correspondence, map point cloud to a template face
- step 4: PCA



3D Face priors - shape & albedo

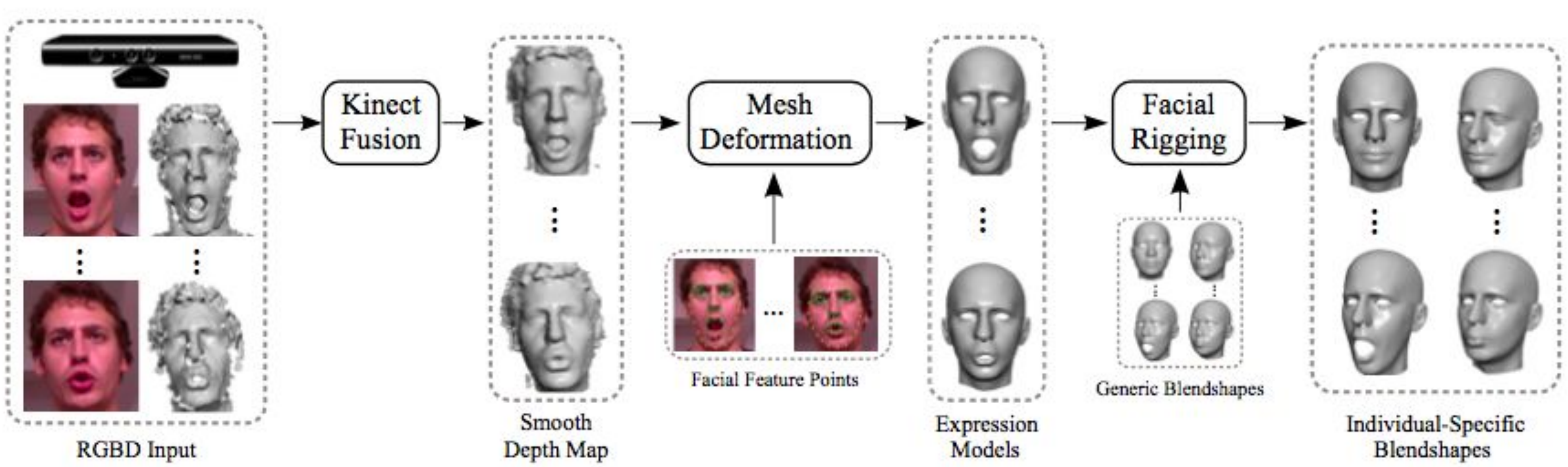
- Mean face and first 5 principal components



Mean Face

3D Face priors - expression

- Example-based facial rigging (2010)
- FaceWarehouse: a 3D Facial Expression Database for Visual Computing (2012)



3D Face prior - takeaway

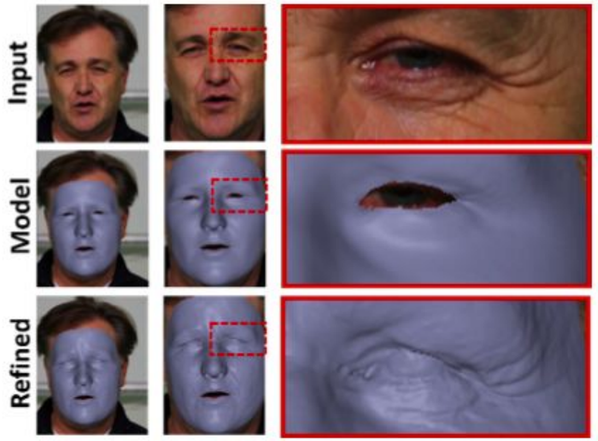
- Commonly, coarse shape of arbitrary face can be formulated as:

$$\mathbf{S} = \overline{\mathbf{S}} + \mathbf{A}_{id}\boldsymbol{\alpha}_{id} + \mathbf{A}_{exp}\boldsymbol{\alpha}_{exp},$$

- Term 1: Mean shape
- Term 2: eig_vec_identity * pca_coeff_identity
- Term 3: eig_vec_expression * pca_coeff_expression

Optimization-based 3D Face Fitting

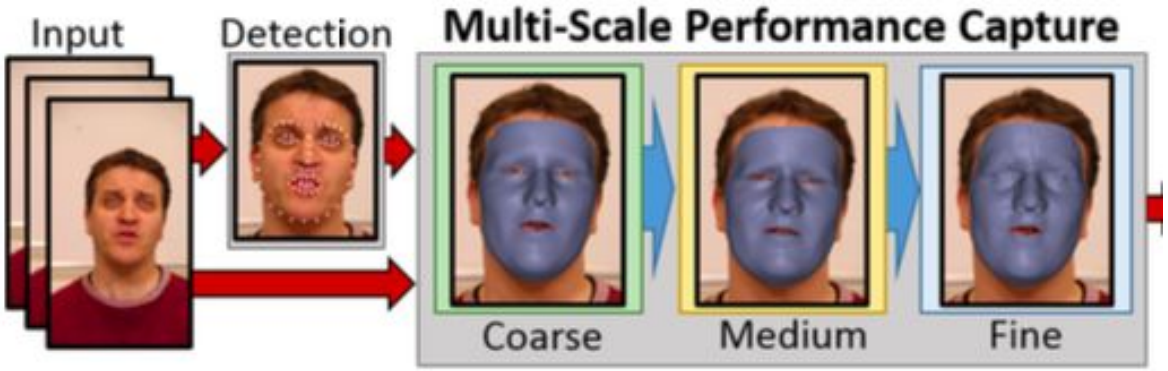
- 3D Face fitting can be solved by traditional gradient descent method.
- Params to solve:
    - Inner/outer camera matrix ~ 7 dims
    - Face 3D pose ~ 3 dims
    - Face shape ~ 60 dims
    - Face expression ~ 46 dims
    - Face albedo ~ 60 dims
    - Lighting ~ 30 dims
- Objective function
    - Consistency term: L2 per-pixel distance between reconstructed image and input image
    - Landmark term: L2 distance between facial landmarks of reconstructed mesh and ground truth
    - Regulation term: L2 norm of face shape/expression/albedo coeffs
- Further higher-accuracy shape(out of PCA subspace) can be solved by additional refinement by shape from shading.
- During each iteration of gd, the image are re-rendered using current params, and partial derivatives are calculated.
- Unluckily, though rendering step is differentiable, it can not be easily written using symbolic(z-buffer makes it hard), this problem can not be solved use NN efficiently.
- The result params can be treated as ground truth label for the following NN fit methods.



Shape from shading example

3D Face – three stages of face shape fitting

- Coarse level: shape lies in the subspace of the prior PCA model (optimization or NN)
- Medium level: optimize parametric deformation field to face mesh (optimization), optional
- Fine level: get per-pixel normal/depth from shading, thus reconstruct face mesh (optimization or NN)



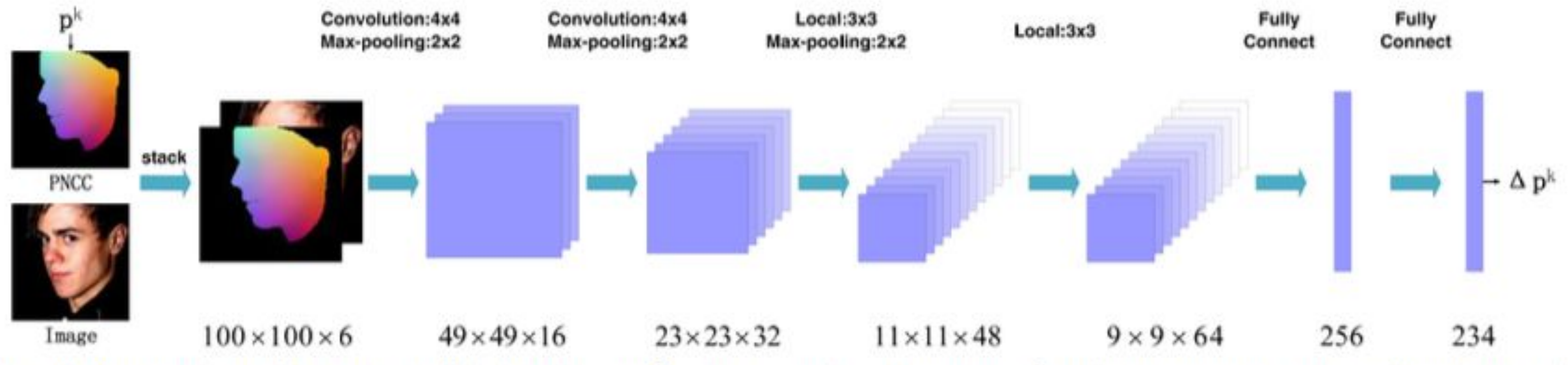
Reconstruction of Personalized 3D Face Rigs from Monocular Video (2016)

3D Face - Coarse Net

- Input an image, use CNN to learn camera parameter + 3D face shape/exp parameter(formulated using prior) directly
- It is very hard to converge, since abstract params have no direct relation with the input image

3D Face - Coarse Net

- Face Alignment Across Large Poses: A 3D Solution (2015), for RGB stills
- Explicitly add image presentation of current recovered parameters to the input, named PNCC
- In cascade manner, each level of the cascade learn a parameter update.
- Loss function is tricky: average mesh vertex 2d distance between predicted params and ground truth params

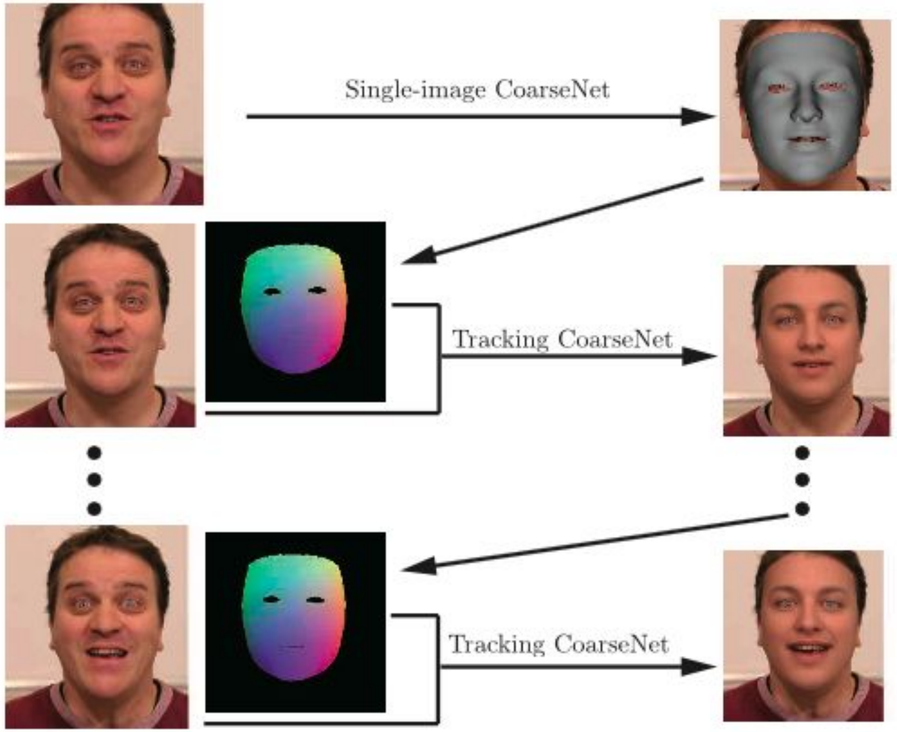


3D Face – Coarse Net

- 3DFaceNet: Real-time Dense Face Reconstruction via Synthesizing Photo-realistic Face Images (2017) , for RGB sequences

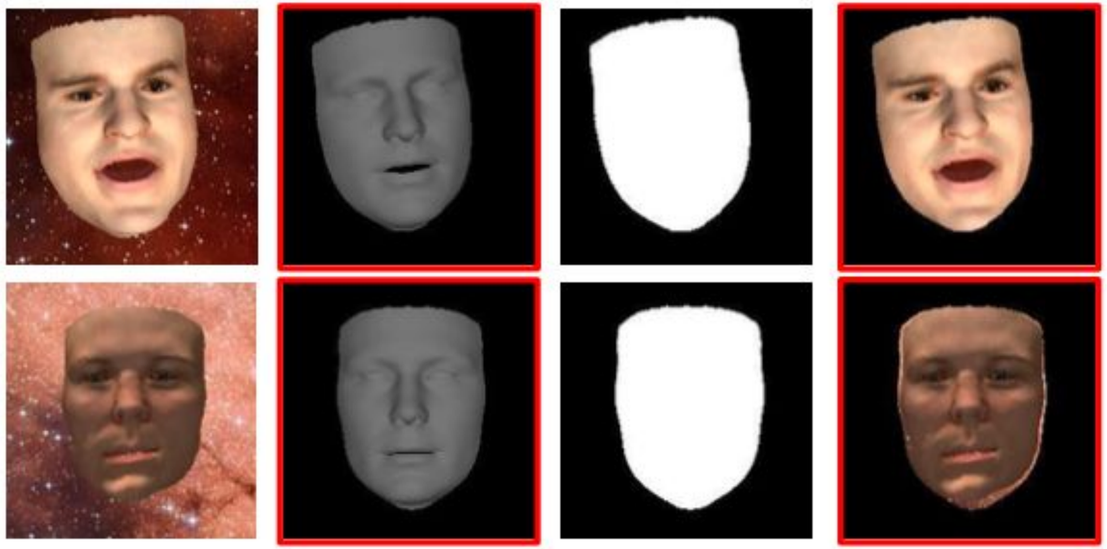
- Try to utilize temporal correlation: PNCC of predicted params of previous frame are transferred to the next frame as the input.

- More robust for image streams when inference



3D Face – Coarse Net

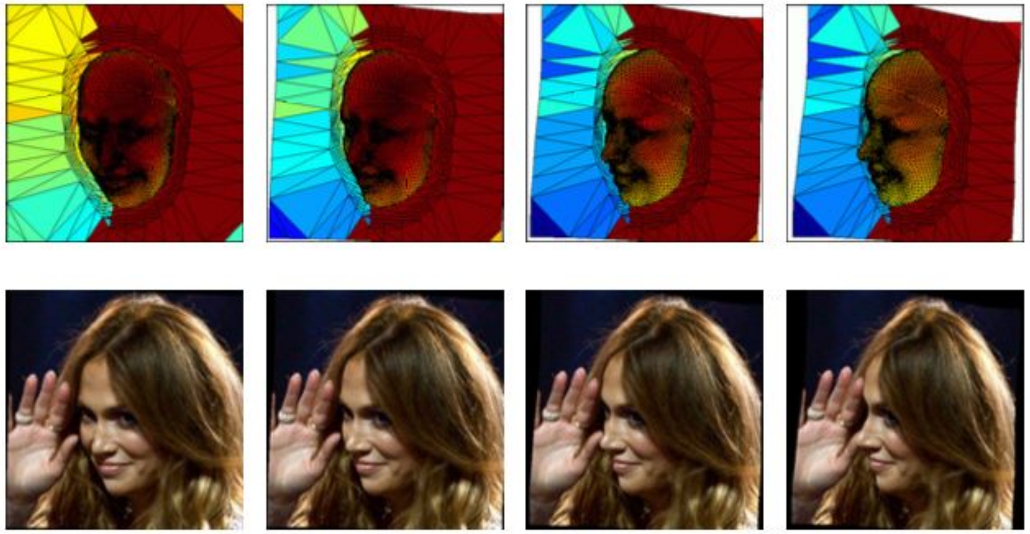
- Training data: from opt-based face fit, 3D face dataset, or mainly, from rendering
- Direct render from prior, with random coeffs, random render settings



3D Face Reconstruction by Learning from Synthetic Data (2016)

3D Face – Coarse Net

- Training data: 3D augmentation
- inverse rendering + re-render + 2D image warp, to synthesis various 3D pos examples

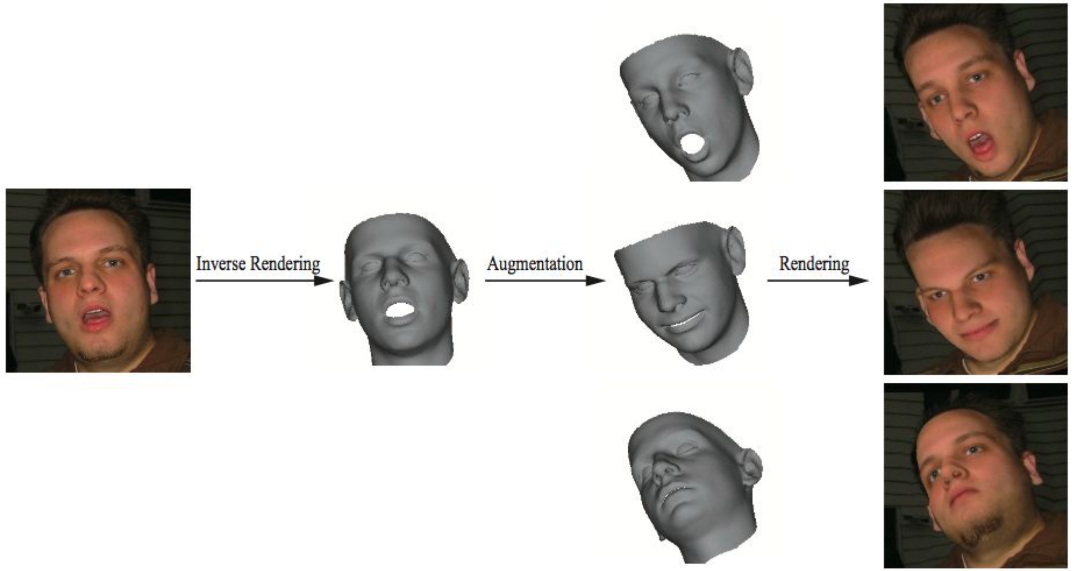


Face Alignment Across Large Poses: A 3D Solution (2015)

3D Face – Coarse Net

- Training data: 3D augmentation

- inverse rendering + modify prior model coeffs + re-render, to synthesis various 3D pos and expression examples

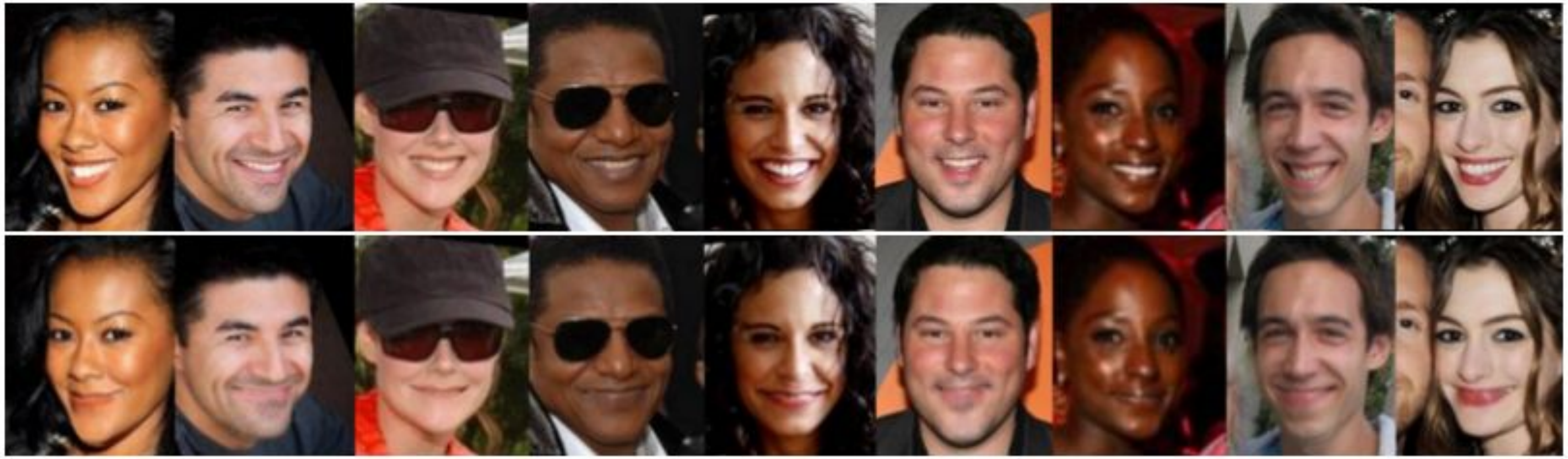


3DFaceNet: Real-time Dense Face Reconstruction via Synthesizing Photo-realistic Face Images (2017)

3D Face – Coarse Net

- Training data: 3D augmentation

- inverse rendering + modify prior model coeffs + re-render, to synthesis various 3D pos and expression examples



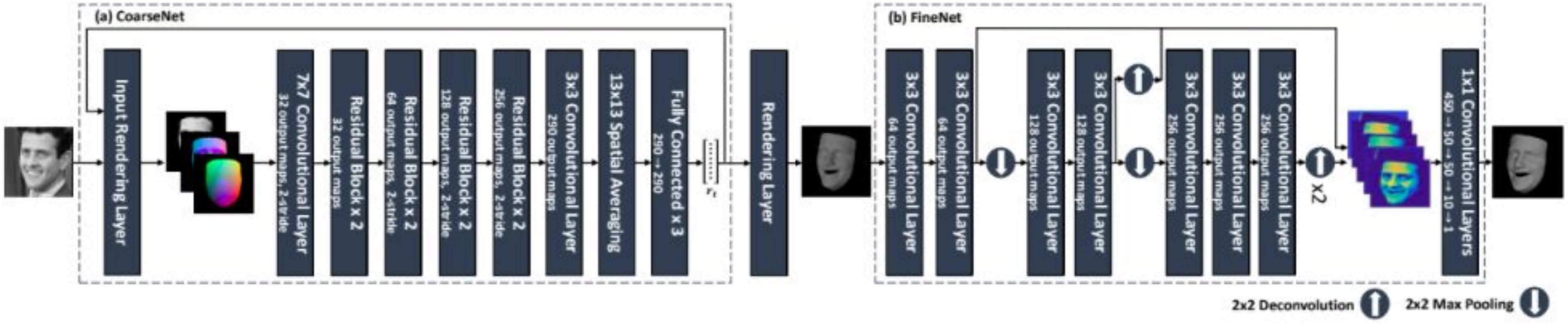
Do We Really Need to Collect Millions of Faces for Effective Face Recognition? (2016)

3D Face – Fine Net

- Get per-pixel normal/depth from shading, which is commonly called normal map, depth map, displacement map(bump map)
- Reconstruct face mesh by integral over normal/depth

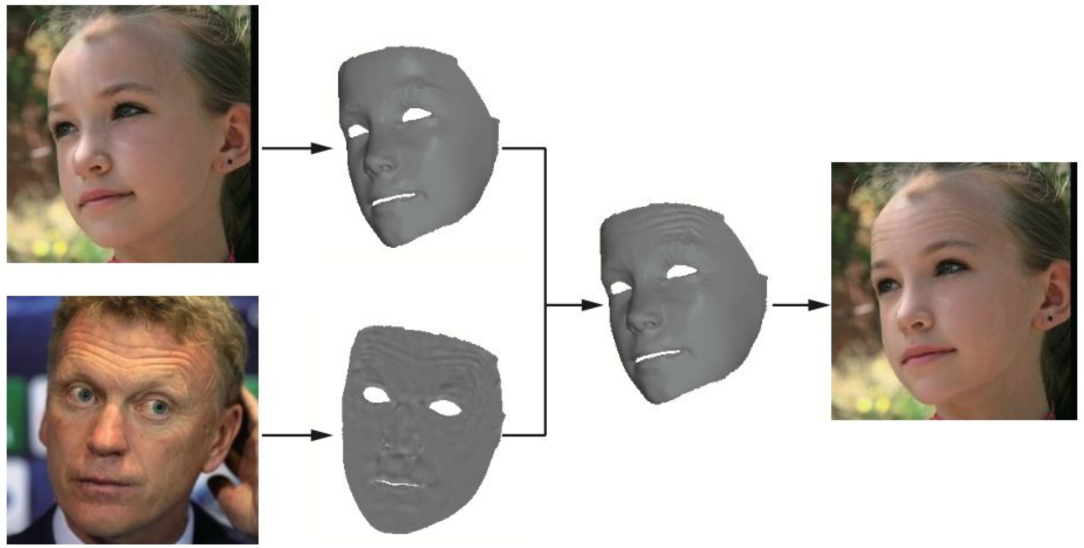
3D Face – Fine Net

- Learning Detailed Face Reconstruction from a Single Image (2017)
- End-2-End coarse net + fine net, though rendering layer is involved(training and inference are slow).
- Fine net is CNN, loss is L2 difference between resulting depth map and opt-based method depth map(using shape from shading).


(a) CoarseNet
Input Rendering Layer
7x7 Convolutional Layer 32 output maps, 2-stride
Residual Block x 2 32 output maps
Residual Block x 2 64 output maps, 2-stride
Residual Block x 2 128 output maps, 2-stride
Residual Block x 2 256 output maps, 2-stride
3x3 Convolutional Layer 290 output maps
13x13 Spatial Averaging
Fully Connected x 3 290 → 290
Rendering Layer
(b) FineNet
3x3 Convolutional Layer 64 output maps
3x3 Convolutional Layer 64 output maps
3x3 Convolutional Layer 128 output maps
3x3 Convolutional Layer 128 output maps
3x3 Convolutional Layer 256 output maps
3x3 Convolutional Layer 256 output maps
3x3 Convolutional Layer 256 output maps
x2
1x1 Convolutional Layers 450 → 50 → 50 → 10 → 1
2x2 Deconvolution
2x2 Max Pooling

3D Face – Fine Net

- Training data: 3D augmentation
- inverse rendering + transfer displacement map, to synthesis more images with shape details

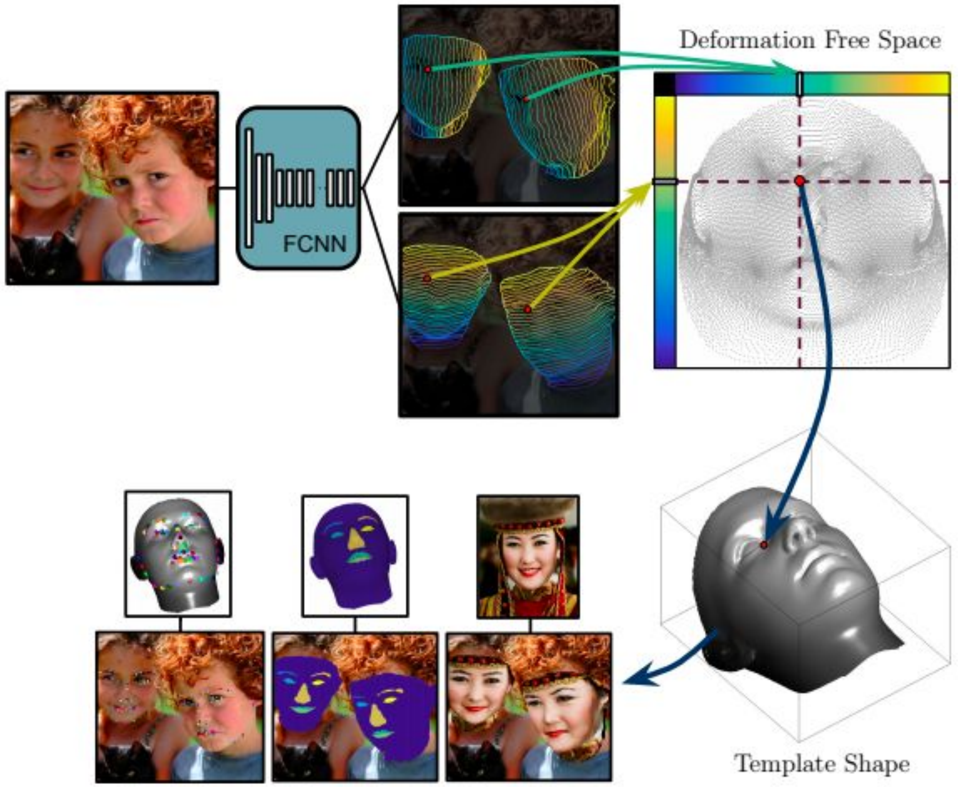


3DFaceNet: Real-time Dense Face Reconstruction via Synthesizing Photo-realistic Face Images (2017)

- Some workaround from face prior

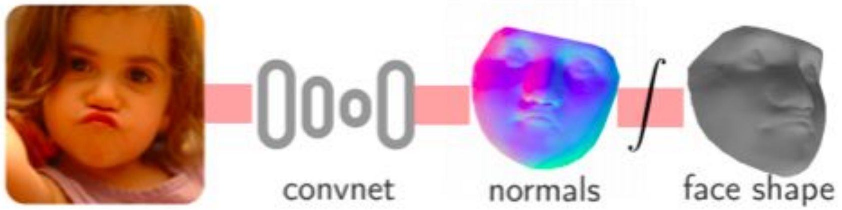
3D Face – without prior

- DenseReg: Fully Convolutional Dense Shape Regression In-the-Wild(2017)
- Input single image, output uv code map
- uv code is the correspondence of a template face, which is capable for face alignment



3D Face – without prior

- Face Normals "in-the-wild" using Fully Convolutional Networks (2017)
- Input single image, output per-pixel normal map
- Face shape can be computed from the integral of normal map.



3D Face - takeaway

- Face prior makes 3D Face fit from monocular RGB stills/sequences doable. Robustness: sequences > stills, face prior > face prior workarounds

- Depth info can highly reduce ambiguity, even with coarse depth info.

- Synthetic data plays a very important role, 3D data generation and 3D face augmentation is tricky but successful.

- Coarse Net is relatively easy to implement, fast to run, has not enough precision, but able to handle some real-time applications on mobiles.

- Fine Net is hard to train, with higher precision, but currently less applicable.

- We are on the way towards 3D Face Recognition. It would be a long way but more and more researchers are joining.

Rendering for Computer Vision

“How complicated a virtual analytic world should be, to reveal the true order of real world, from Computer Vision perspective?”

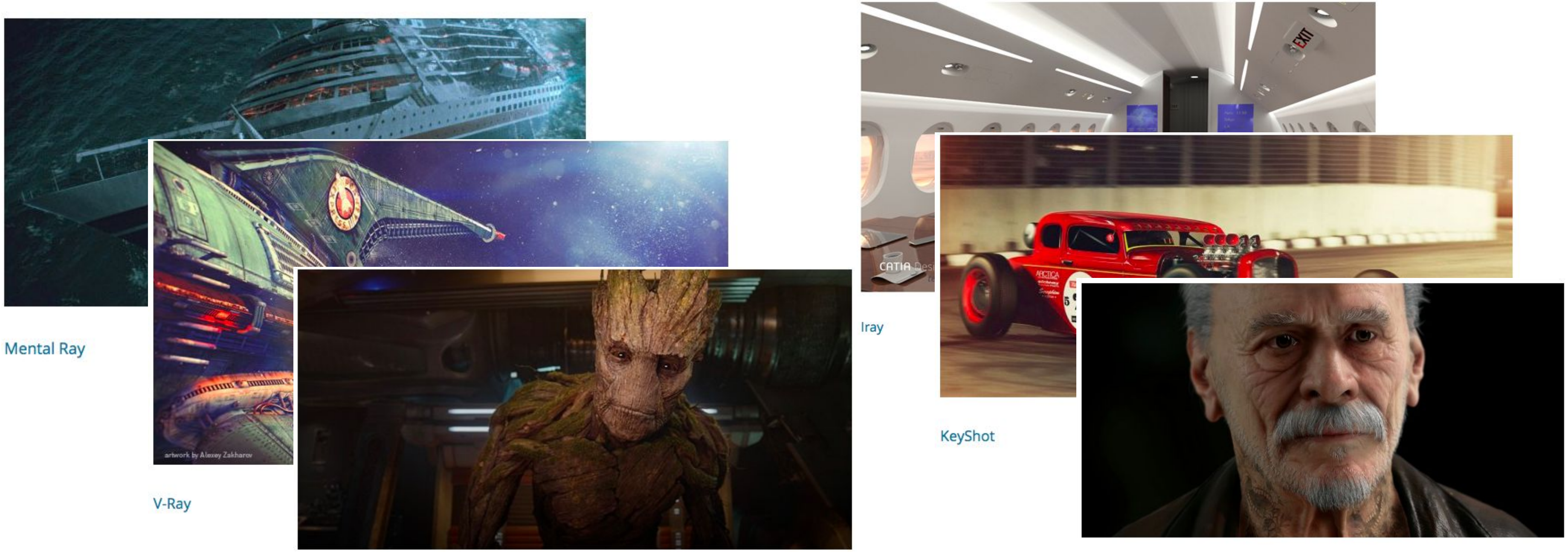
"Will NN be the judge?”

Rendering for CV

- Data is the most important part for CV tasks.
- Real data and labeling is expensive and hard to get sometimes.
- Synthetic data plays a more and more important role in CV, even for traditional tasks.

Render for CV

- Generations of GPU and powerful Rendering Engines are waiting there, but not enough effort to explore how much they would boost the performance of CV tasks by render synthesis data.



Mental Ray

V-Ray

RenderMan

Iray

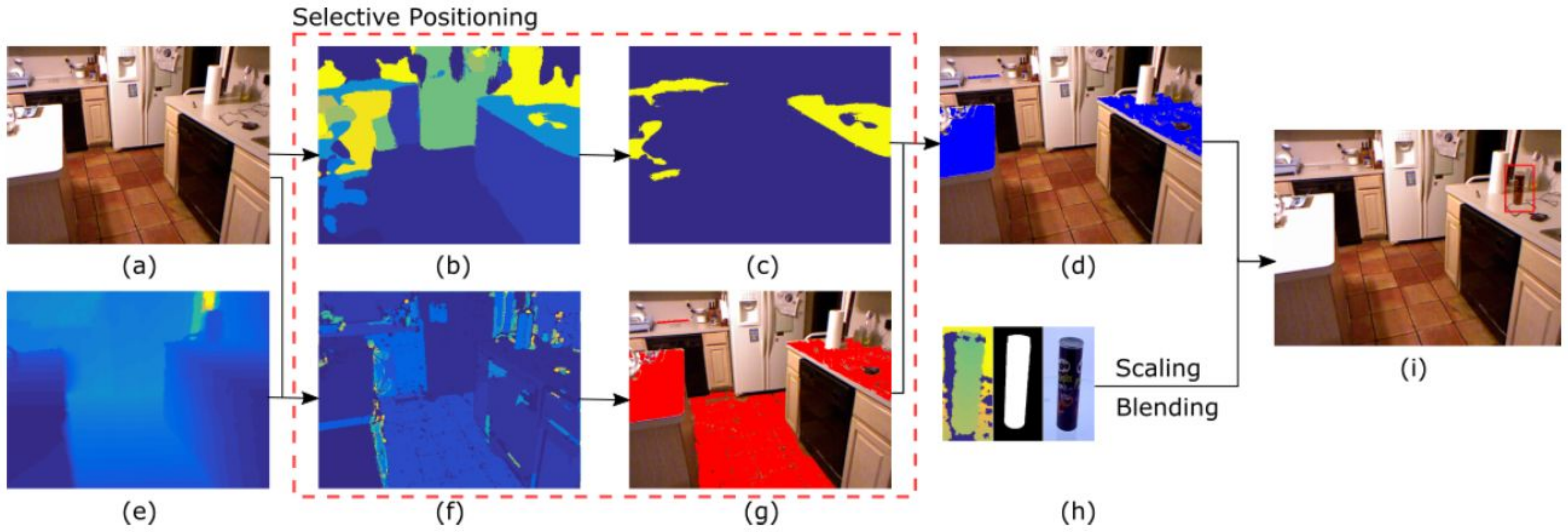
KeyShot

Toolbag

Render for CV

- Synthesizing Training Data for Object Detection in Indoor Scenes, (2017)

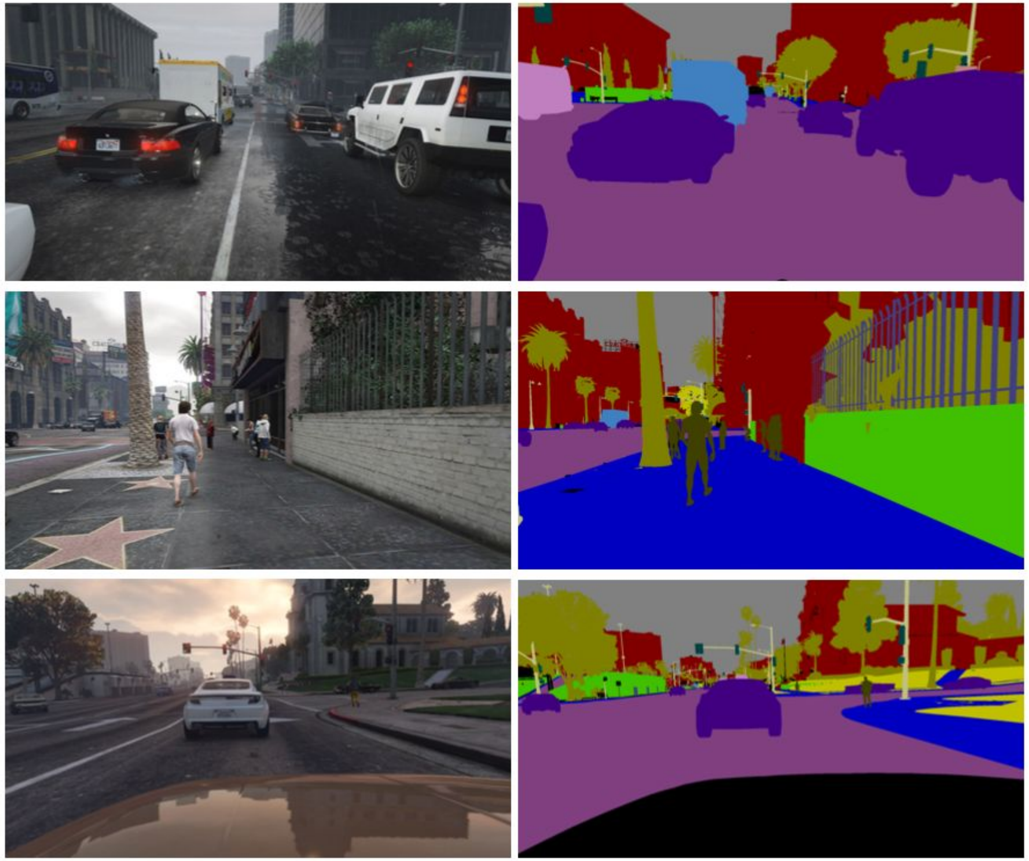
- Composite additional objects in real in-door images for object detection task.



Render for CV

- Playing for Data: Ground Truth from Computer Games (2016)

- Semantic segmentation ground truth data can be directly got from games.

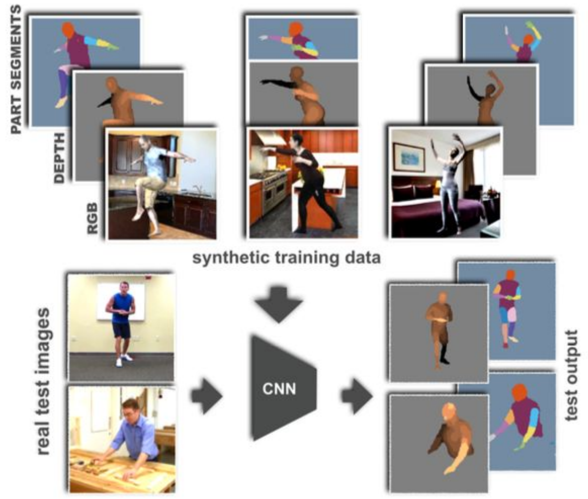


Render for CV

- LCrowdV: Generating Labeled Videos for Simulation-based Crowd Behavior Learning (2016)
- Synthesis crowed images, no need to label ground truth.

Render for CV

- Learning from Synthetic Humans (2017)
- Synthesis fake human to do human part segmentation, depth estimation



Render for CV – takeaway

- Theoretically rendering makes training data endless. But additional data become less useful as dataset size becomes larger. For traditional cv tasks, which already have large data size, improvement is little.

- Overall, generalization power limits the use of synthetic data. But in more and more cases, researchers find training on well-tuned synthetic data do not affect generalization power, currently.

- New research direction: since rendering layer is not easy-symbolic-lize but differentiable, adversarial manner should be applied to improve generalization power. For example, to tune parameters of rendering settings.

Backup after this slide

NN 3D Face

- Data source: monocular RGB/RGBD stills/sequences
- General 3D reconstruction problem from visuals is still very hard due to depth ambiguity.
    - But it’s easier for a specific senario, like portraits!

NN 3D Face - formulation

- Given a face RGB/RGBD still/sequence, reconstruct for each frame:
  - Inner/outer camera matrix
  - Face 3D pose
  - Face shape
  - Face expression
  - Face albedo
  - lighting
- This kind of problem is also called intrinsic image or inverse rendering.
- Highly ill-posed problem