Computer Vision

Lecture 12 – Diverse Topics in Computer Vision

Prof. Dr.-Ing. Andreas Geiger

Autonomous Vision Group

University of Tübingen / MPI-IS

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

ellis
European Laboratory for Learning and Intelligent Systems

In this class, we have learned a lot ..

- History of computer vision
- Primitives and transformations
- Geometric image formation
- Photometric image formation
- Camera calibration
- SIFT detection and matching
- Epipolar geometry
- Structure-from-Motion
- Bundle adjustment
- Image rectification
- Block matching
- Siamese networks
- Probabilistic graphical models
- Belief propagation
- Structured prediction
- Deep structured models
- Volumetric 3D reconstruction
- Optical flow
- Rendering equation
- Shape-from-Shading
- Photometric stereo
- Multi-view stereo
- Structured light
- Monocular depth
- Volumetric fusion
- Marching cubes
- Implicit neural representations
- Occupancy networks
- Differentiable rendering
- Novel view synthesis
- Neural radiance fields
- Generative radiance fields
- Image classification
- Semantic segmentation
- Object detection
- Average precision
- Instance segmentation
- Data annotation
- Self-supervised learning
- Self-supervised depth/flow
- Pretext tasks
- Contrastive learning

.. but there is a lot more:

- 3D feature learning
- Shape abstraction
- Neural rendering
- Image correspondence
- SLAM, VO & localization
- Morphable 3D models
- 3D detection
- Action recognition
- Adversarial attacks
- Face recognition
- Gaze estimation
- Human body models
- Event cameras
- Omnidirectional vision
- Deblurring, denoising, dehazing
- Superresolution
- Lightfields
- Neuromorphic sensors
- Datasets and benchmarks
- Network pruning
- Few-shot learning
- Open set recognition
- Image/video retrieval
- Image synthesis, GANs, VAEs
- Neural style transfer
- Explainable AI
- Ethics, privacy & fairness
- Deepfakes
- Video prediction
- Saliency prediction
- Bio/medical imaging
- Object tracking
- 6D Pose estimation
- Active learning
- Disentangled representations
- Scene graphs
- Domain adaptation
- Geometric deep learning
- Ego-centric vision
- Vision and language
- Robot vision
- Surveillance

Credits

- **Justin Johnson** — Convolutional Neural Networks + Neural Style Transfer
  `http://web.stanford.edu/class/cs20si/lectures/slides_06.pdf`
- **Leon Gatys** — Image Style Transfer Using Convolutional Neural Networks
  `https://www.youtube.com/watch?v=UFffxcCQMPQ`
- **Michael Black** — SMPL made Simple
  `https://smpl-made-simple.is.tue.mpg.de/`
  `https://www.youtube.com/watch?v=rzpiSYTrRU0`
- **Matthias Niessner** — Deepfakes Creation and Detection
  `https://www.youtube.com/watch?v=-Xv2IRs2-KA`

Agenda

12.1 Input Optimization

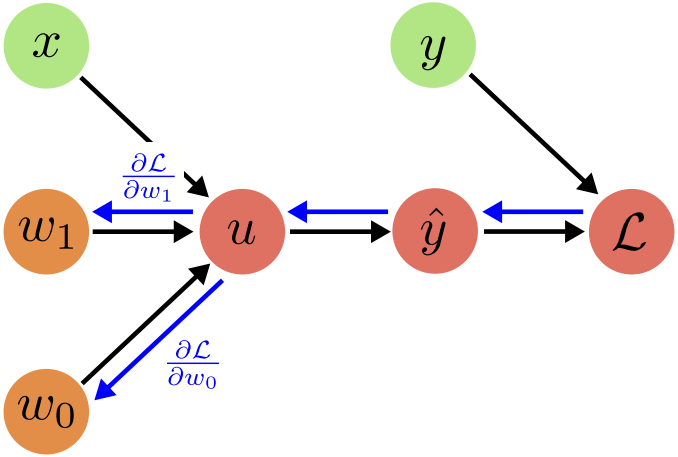
12.2 Compositional Models

12.3 Human Body Models

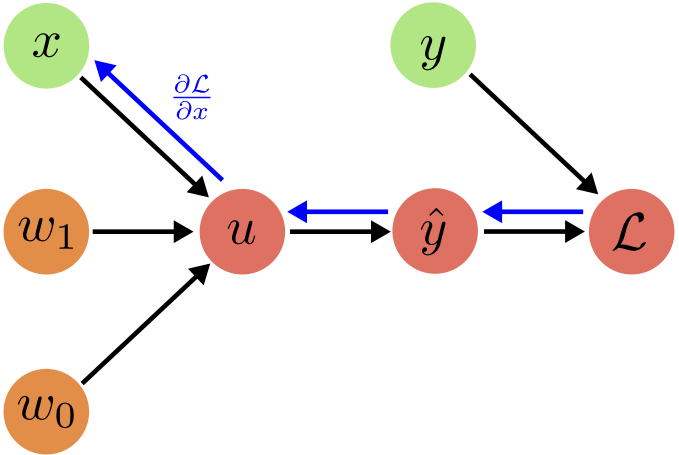
12.4 Deepfakes

12.1
Input Optimization

Weight Optimization vs. Input Optimization



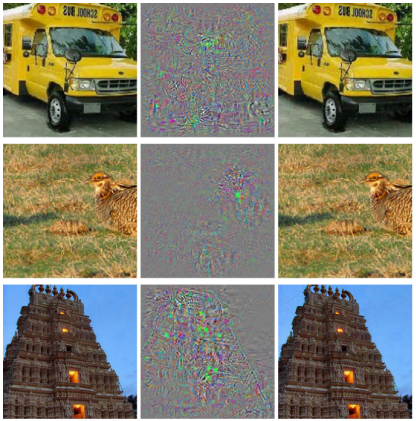
Weight Optimization vs. Input Optimization



- We will see two examples in this unit: **Adversarial attacks** and **style transfer**

Adversarial Attacks

Adversarial Attacks on Image Classification



L-BFGS Attack:

- Given classifier $f : \mathbb{R}^m \to \{1, \ldots, L\}$
- Find **adversarial example** for image x

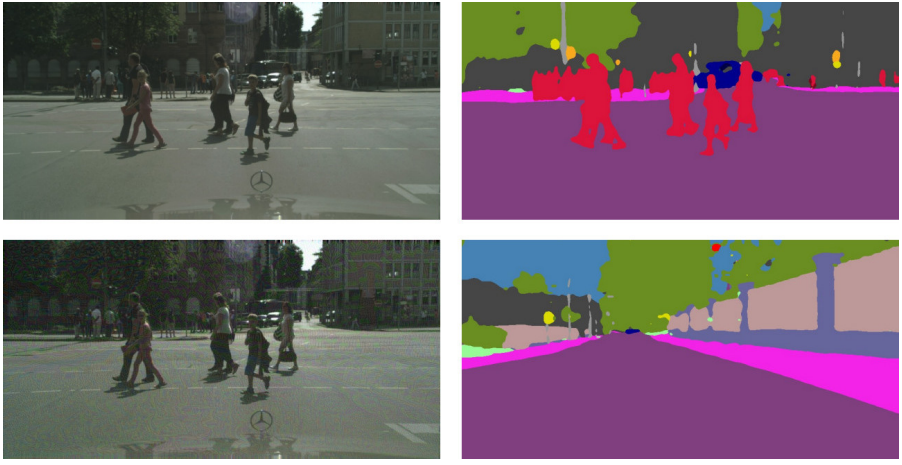
$$x + \underset{\Delta x}{\operatorname{argmin}} \{\|\Delta x\|_2 : f(x + \Delta x) = y_t\}$$

  where y_t denotes the target class

- All images in the right column classified as "ostrich", manipulations (Δx) are magnified

Szegedy, Zaremba, Sutskever, Bruna, Erhan, Goodfellow and Fergus: Intriguing properties of neural networks. ICLR, 2014

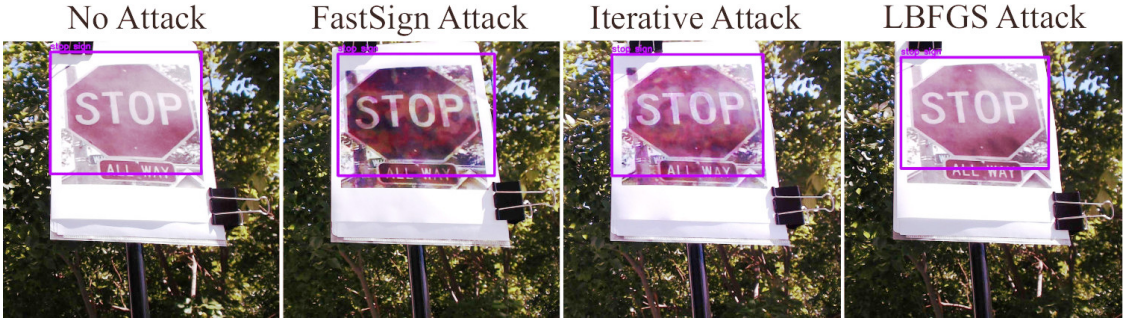
Adversarial Attacks on Semantic Segmentation



- Attack on **semantic segmentation** manipulates label map

Metzen, Kumar, Brox and Fischer: Universal Adversarial Perturbations Against Semantic Image Segmentation. ICCV, 2017.

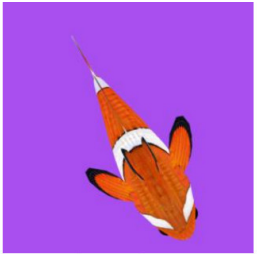
Physical Adversarial Attacks



"Adversarial perturbation methods applied to **stop sign detection** only work in carefully chosen situations, and our preliminary experiment shows that we might **not need to worry** about it in many real circumstances, specifically with autonomous vehicles."

Lu, Sibai, Fabry and Forsyth: NO Need to Worry about Adversarial Examples in Object Detection in Autonomous Vehicles. Arxiv, 2017.

Robust Adversarial Attacks

- Demonstrate existence of **robust adversarial examples** in the physical world
- Maximize **expectation over transformation** $\mathcal{T}$ (EOT):

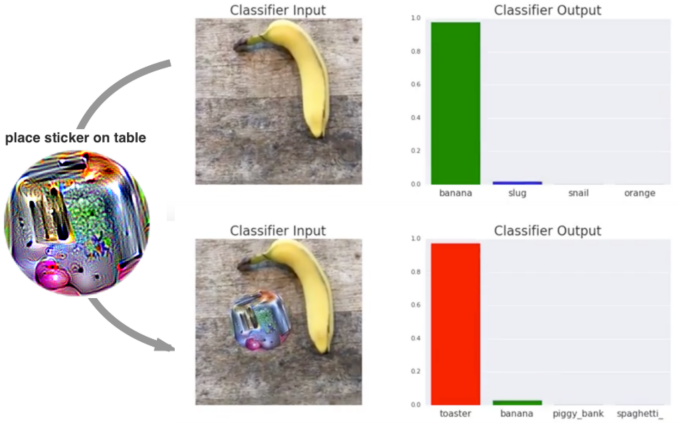
$$\underset{x'}{\operatorname{argmax}} \; \mathbb{E}_{t\sim\mathcal{T}} \left[\log P(y_t|t(x')) - \lambda\|(t(x') - t(x)\|_2\right]$$

- Larger distributions require larger perturbations

Robust Adversarial Attacks

- **Robust adversarial example** designed to mimic “graffiti”

Adversarial Patch Attacks



- **Patch attacks** use EOT idea, but additionally also optimize across **many images**
- Easy to apply in real-world settings (attaching patch to an object)

Brown, Mane, Roy, Abadi and Gilmer. Adversarial patch. Arxiv, 2017.

Attacking Optical Flow

Given an image pair from a video sequence ...

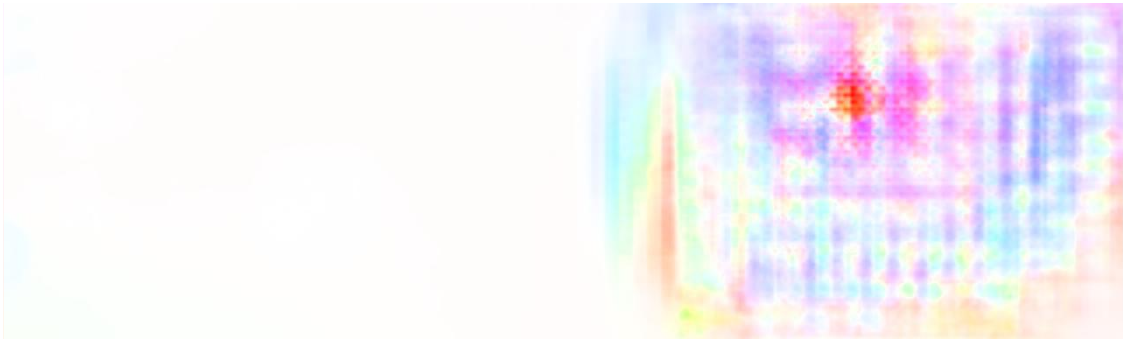
Attacking Optical Flow

... FlowNet2 predicts a smooth optical flow field.

Attacking Optical Flow

Can we obtain a small ($< 1\%$ area) attack patch $\hat{p}$ ...

Attacking Optical Flow



... that successfully attacks the optical flow network?

Attacking Optical Flow

Let $F(I, I')$ denote an **optical flow network** and $\mathcal{I}$ a dataset of frame pairs $\{(I, I')\}$.
Let $A(I, p, t, l)$ denote **image** I with **patch** p transformed by t inserted at location l.
Let $\mathcal{T}$ denote a distribution over affine 2D transformations.
Let $\mathcal{L}$ denote a (uniform) distribution over the image domain.

Goal: Find a patch $\hat{p}$ such that

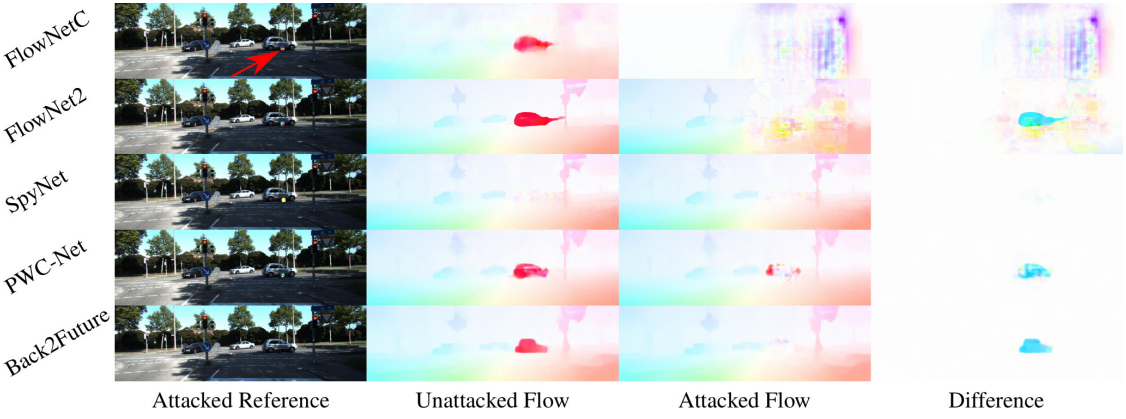
$$\hat{p} = \underset{p}{\operatorname{argmin}}\, \mathbb{E}_{(I,I')\sim\mathcal{I},\, t\sim\mathcal{T},\, l\sim\mathcal{L}} \left[\frac{(u, v) \cdot (\tilde{u}, \tilde{v})}{\|(u, v)\|_2 \cdot \|(\tilde{u}, \tilde{v})\|_2} \right]$$

with $\quad (u, v) = F(I, I')$

$$(\tilde{u}, \tilde{v}) = F(A(I, p, t, l), A(I', p, t, l))$$

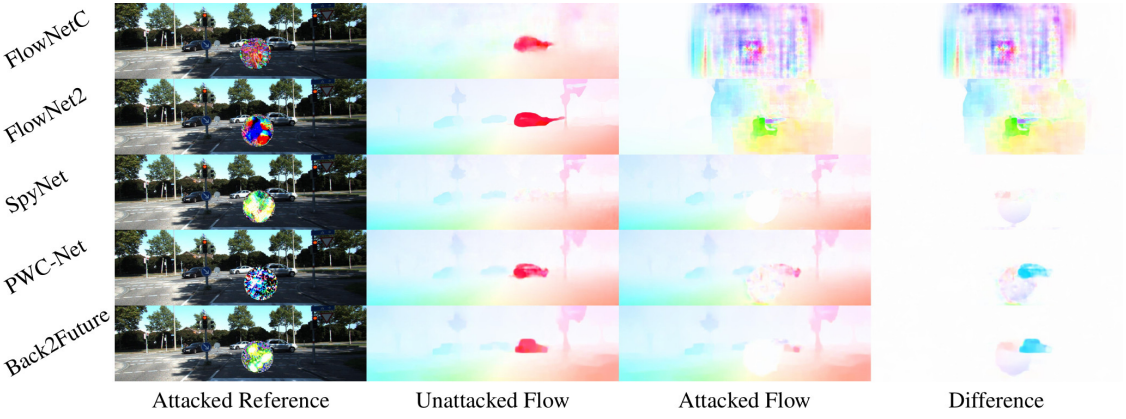
Intuition: Find patch $\hat{p}$ that reverses the direction of the optical flow.

White-Box Attacks



- Attacks **extend beyond region** of patch for encoder-decoder architectures

White-Box Attacks

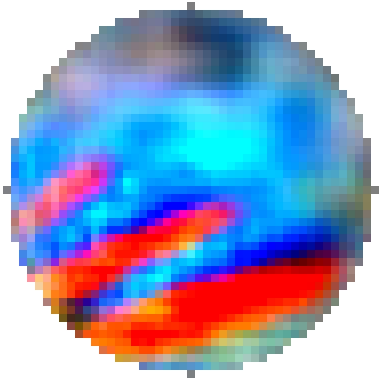


- Attacks **extend beyond region** of patch for encoder-decoder architectures

Black-Box Attacks

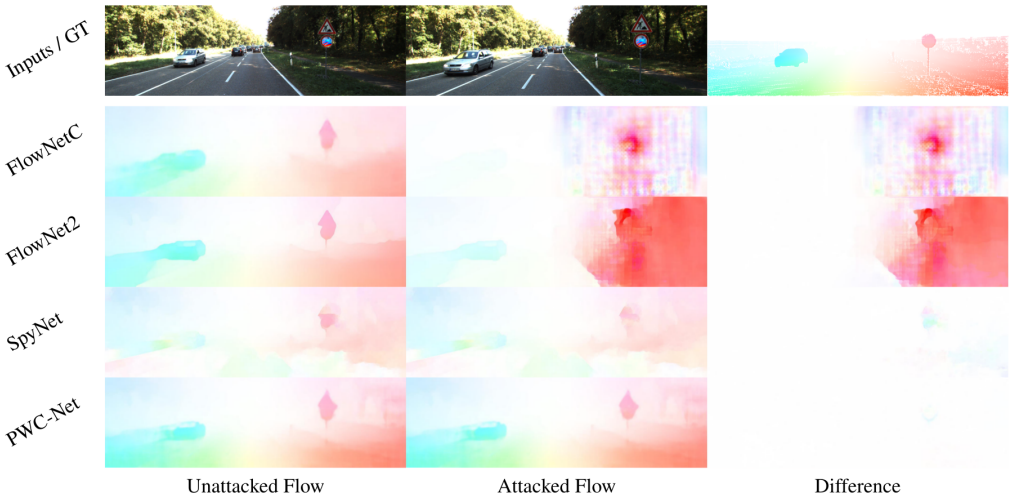
Setting:

- Patch optimized for two networks (FlowNet2 and PWCNet)
- Used to attack unseen network



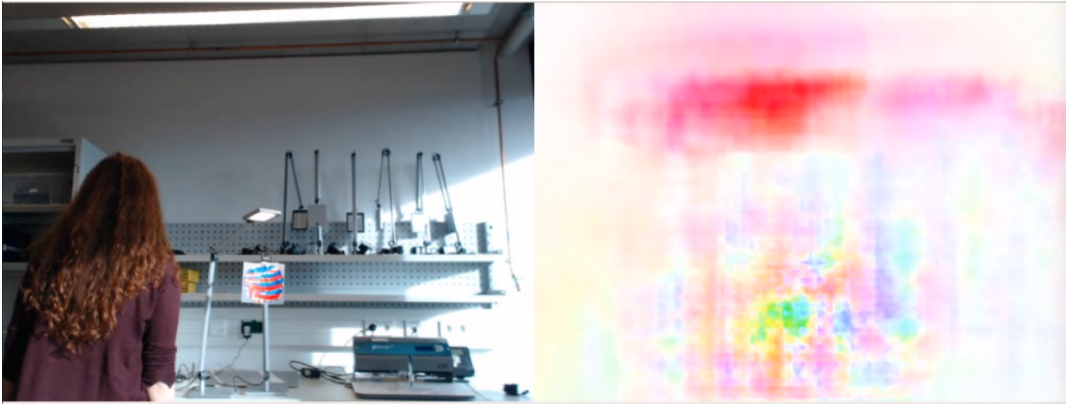
Ranjan, Janai, Geiger and Black: Attacking Optical Flow. ICCV, 2019.

Black-Box Attacks



Ranjan, Janai, Geiger and Black: Attacking Optical Flow. ICCV, 2019.

Real-World Attack



- Printed patch attached to desk lamp

Defenses against Adversarial Attacks

An entire field evolved around **defending adversarial attacks:**

- **Gradient Masking/Obfuscation:** Since most attack algorithms are based on gradient information of the classifier, masking or hiding the gradients will confound the adversaries. [Papernot, 2016] [Song, 2017] [Buckmann, 2018]
- **Robust Optimization:** Re-learning a DNN classifier's parameters can increase its robustness. The trained classifier will correctly classify the subsequently generated adversarial examples. [Goodfellow, 2014] [Madry, 2017] [Hein, 2017]
- **Adversarial Example Detection:** Study the distribution of natural/benign examples, detect adversarial examples and disallow them as input to the classifier. [Carlini, 2017] [Grosse, 2017] [Metzen, 2017]

Xu et al.: Adversarial Attacks and Defenses in Images, Graphs and Text: A Review. IJAC, 2020.

Neural Style Transfer

Neural Style Transfer

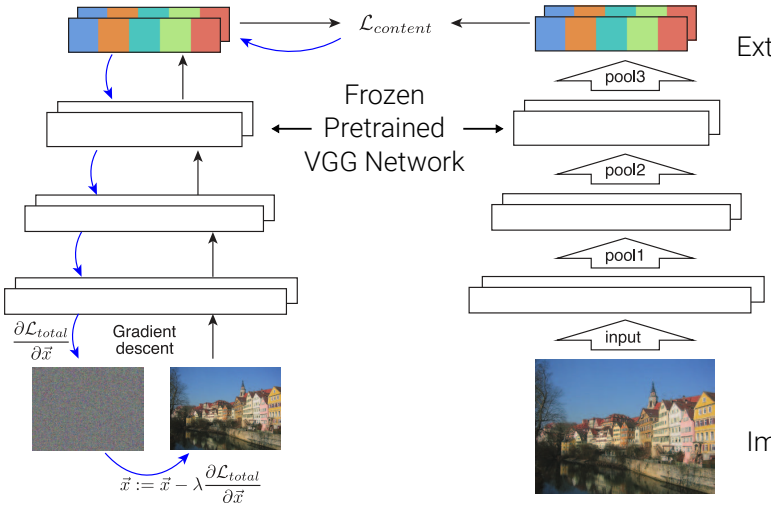
Neural Style Transfer:

- Synthesize an image with the **content** of one image and the **style** of another
- Start with **random image,** optimize content and style loss: $\mathcal{L} = \mathcal{L}_{\text{content}} + \mathcal{L}_{\text{style}}$
- Exploits **VGG** network pre-trained on **ImageNet** as feature extractor

Gatys, Ecker and Bethge: Image Style Transfer Using Convolutional Neural Networks. CVPR, 2016.

Content Reconstruction

Step 4:
Optimize Image to minimize Content Loss



Step 2:
Extract Features

Step 3:
Initialize Generate Image Randomly

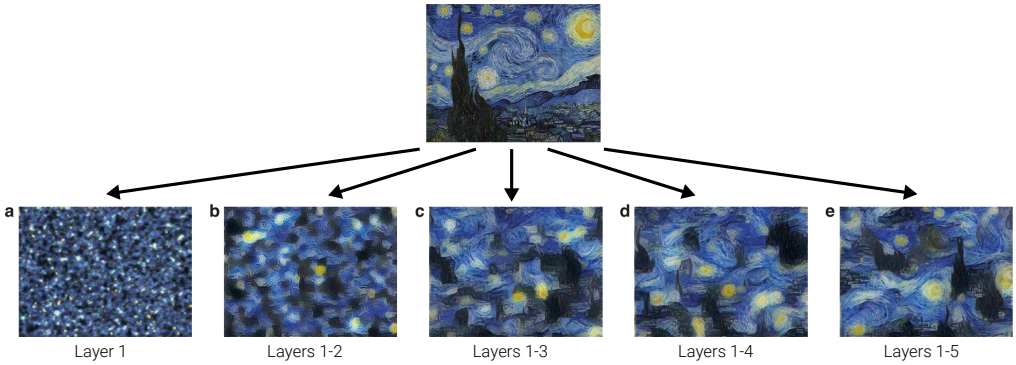
Step 1:
Pass Input Image through VGG

Content Reconstruction

$$\mathcal{L}_{\text{content}}(l) = \|\mathbf{F}_l - \mathbf{P}_l\|_2^2$$

- $\mathbf{F}_l$: Features of **generated** image at VGG layer l
- $\mathbf{P}_l$: Features of **content** image at VGG layer l

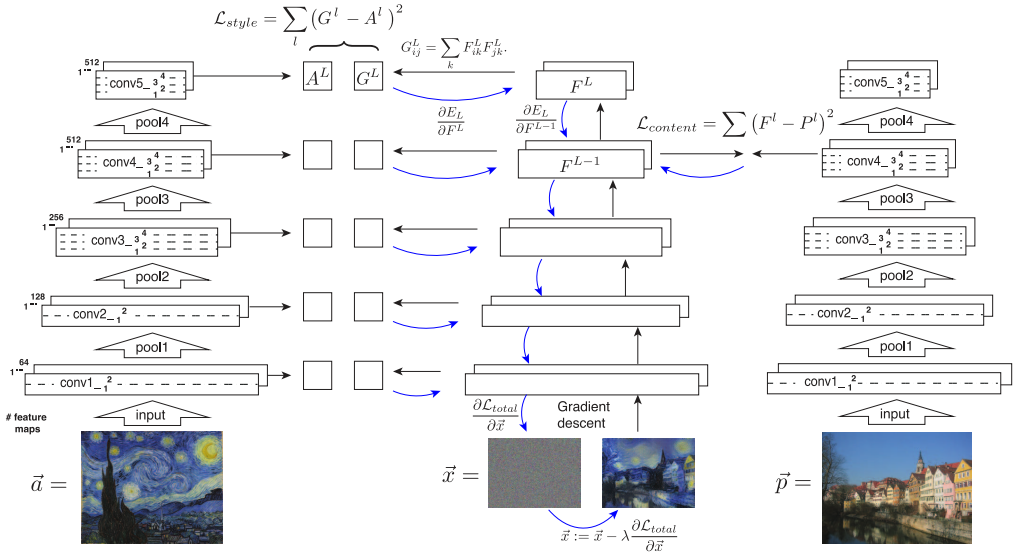
Style Reconstruction



$$\mathcal{L}_{\text{style}}(l) = \|\mathbf{G}_l - \mathbf{A}_l\|_2^2 \qquad G_{ij}^l = \sum_{\mathbf{p}\in\Omega} F_i^l(\mathbf{p}) F_j^l(\mathbf{p})$$

- $\mathbf{G}_l, \mathbf{A}_l$: **Gram matrices** (feature correlations) of generated and style image

Neural Style Transfer

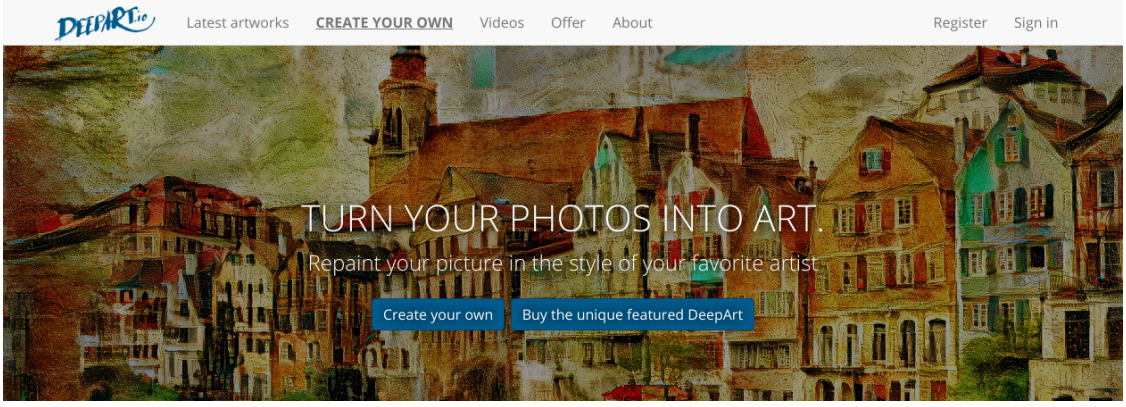


Gatys, Ecker and Bethge: Image Style Transfer Using Convolutional Neural Networks. CVPR, 2016.

Neural Style Transfer

Gatys, Ecker and Bethge: Image Style Transfer Using Convolutional Neural Networks. CVPR, 2016.

Neural Style Transfer



https://deepart.io/

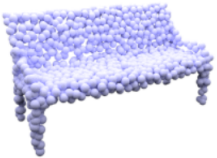
Gatys, Ecker and Bethge: Image Style Transfer Using Convolutional Neural Networks. CVPR, 2016.

12.2
Compositional Models

Shape Abstraction

3D Representations

Voxels



Points

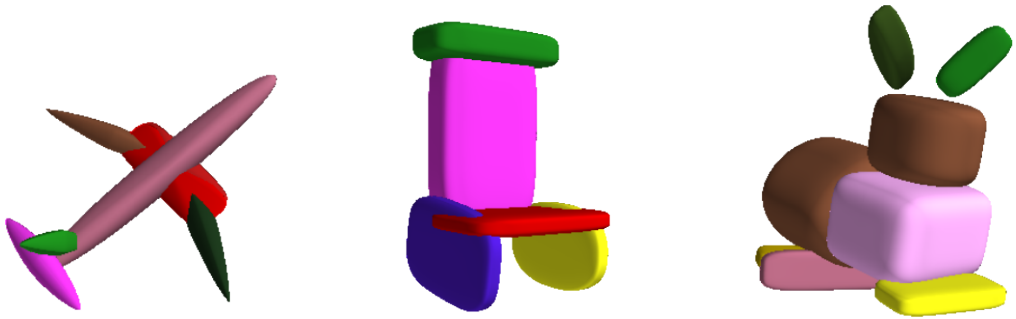
Meshes

Implicit

Limitations of existing 3D representations:

- Large number of unstructured geometric elements
- Do not convey **semantic information** (parts, functionality, etc.)

3D Primitives



Primitive-based 3D Representations:

- **Few primitives** required to represent a 3D object
- Convey **semantic information** (parts, functionality, etc.)
- Challenges: Variable number of primitives, few annotated datasets

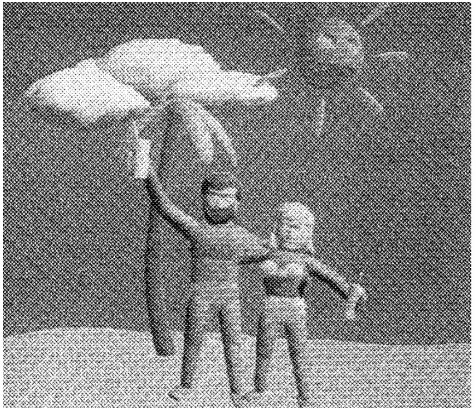
Paschalidou, Ulusoy and Geiger: Superquadrics Revisited: Learning 3D Shape Parsing Beyond Cuboids. CVPR, 2019.

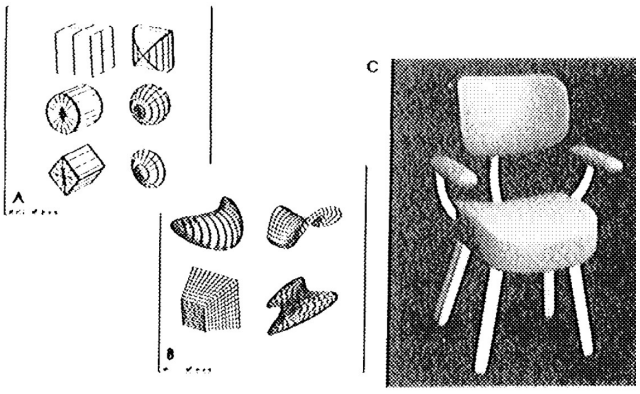
3D Shape Abstraction

Goal:

- Learn 3D shape abstraction
- Infer number of primitives
- No supervision at primitive level
- Input: point cloud or image
- Supervision: point cloud (no labels)

Pentland's Superquadrics Revisited

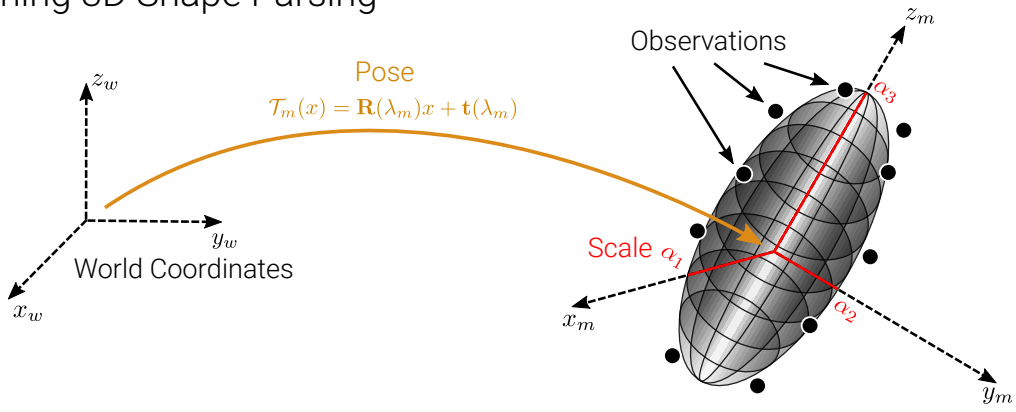




- 1 superquadric = 11 parameters $\Rightarrow$ scene on the left stored in only 1000 bytes!
- However, early fitting-based approaches did not work robustly $\Rightarrow$ learn from data

Pentland: Parts: Structured descriptions of shape. AAAI, 1986.

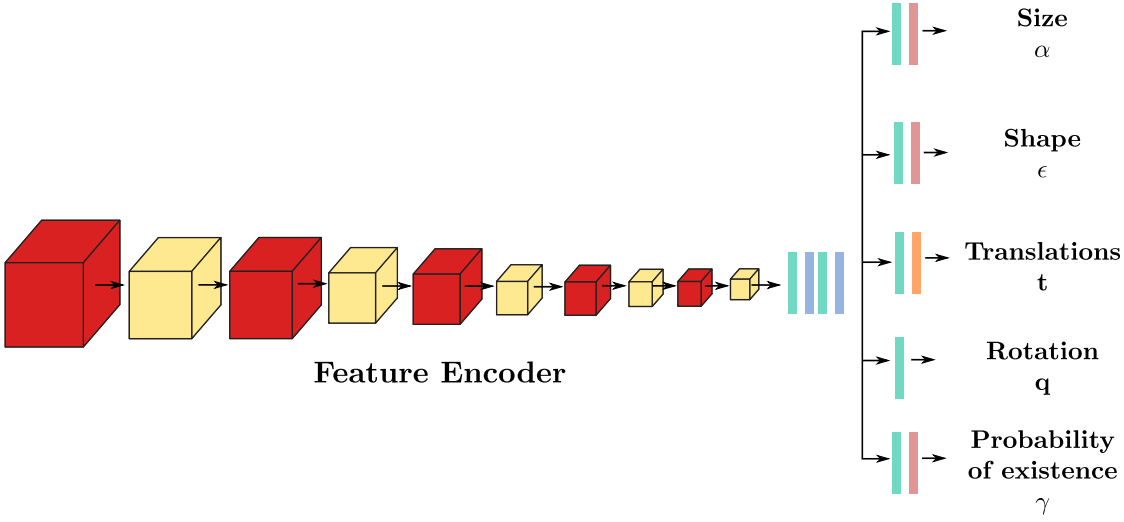
Learning 3D Shape Parsing



Predictions per primitive:

- 11 parameters: 6 pose $(\mathbf{R}, \mathbf{t})$ + 3 scale $(\boldsymbol{\alpha})$ + 2 shape $(\boldsymbol{\epsilon})$
- Probability of existence: $\gamma \in [0, 1]$

Network Architecture



Loss Function

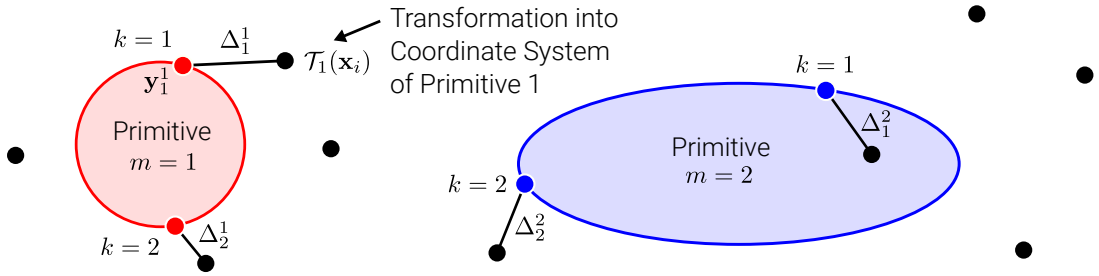
Overall Loss:

$$\mathcal{L}(\mathbf{P}, \mathbf{X}) = \mathcal{L}_{P \to X}(\mathbf{P}, \mathbf{X}) + \mathcal{L}_{X \to P}(\mathbf{X}, \mathbf{P}) + \mathcal{L}_{\gamma}(\mathbf{P})$$

Composed of:

- $\mathcal{L}_{P \to X}(\mathbf{P}, \mathbf{X})$: Primitive-to-Pointcloud Loss
- $\mathcal{L}_{X \to P}(\mathbf{X}, \mathbf{P})$: Pointcloud-to-Primitive Loss
- $\mathcal{L}_{\gamma}(\mathbf{P})$: Existence (at least one primitive) and Parsimony (sparsity) Loss

Primitive-to-Pointcloud Loss



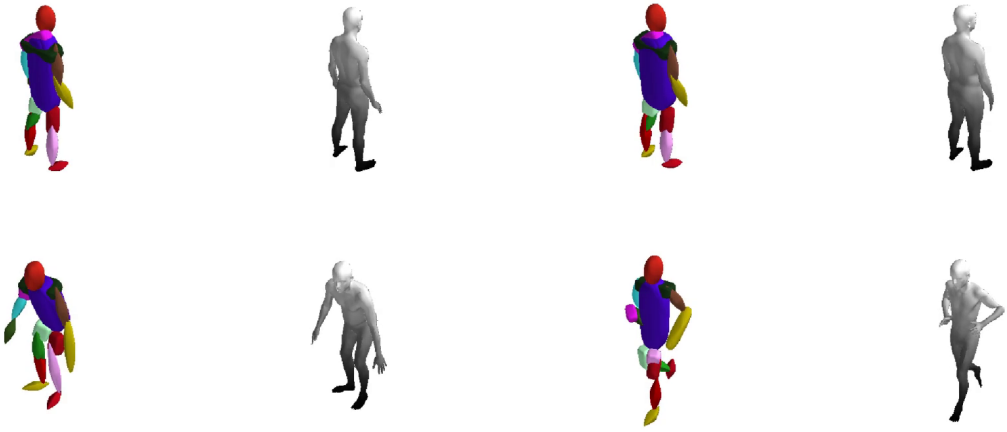
$$\mathcal{L}_{P\to X}^{m}(\mathbf{P},\mathbf{X}) = \frac{1}{K}\sum_{k=1}^{K}\Delta_k^m$$

$$\Delta_k^m = \min_{i=1,..,N} \|\mathcal{T}_m(\mathbf{x}_i) - \mathbf{y}_k^m\|_2$$

$$\mathcal{L}_{P\to X}(\mathbf{P},\mathbf{X}) = \mathbb{E}_{p(\mathbf{z})}\left[\sum_{m|z_m=1}\mathcal{L}_{P\to X}^{m}(\mathbf{P},\mathbf{X})\right]$$

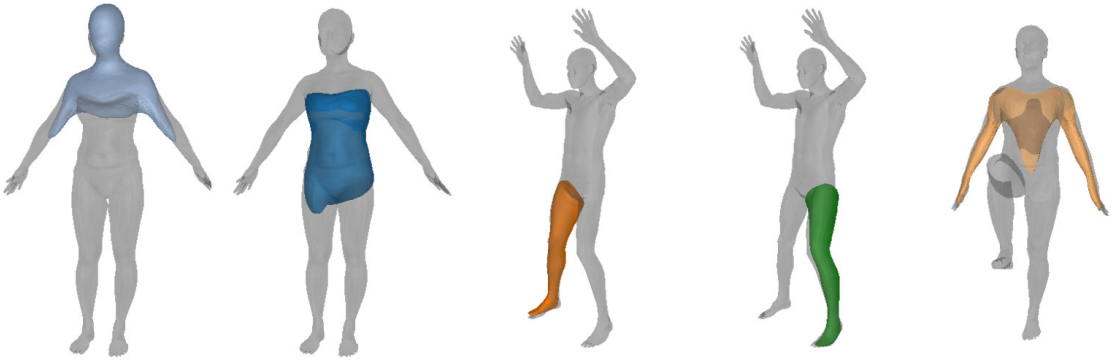
$$= \sum_{m=1}^{M}\gamma_m\,\mathcal{L}_{P\to X}^{m}(\mathbf{P},\mathbf{X})$$

Results



https://autonomousvision.github.io/superquadrics-revisited/

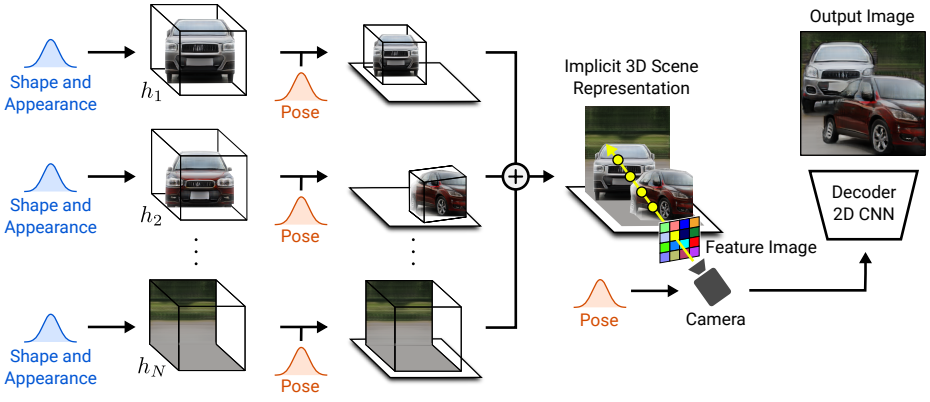
Neural Parts



- More **expressive part shapes** learned using invertible neural networks
- `https://autonomousvision.github.io/neural-parts/`

Compositional Feature Fields

GIRAFFE: Compositional Generative Neural Feature Fields



https://autonomousvision.github.io/giraffe/

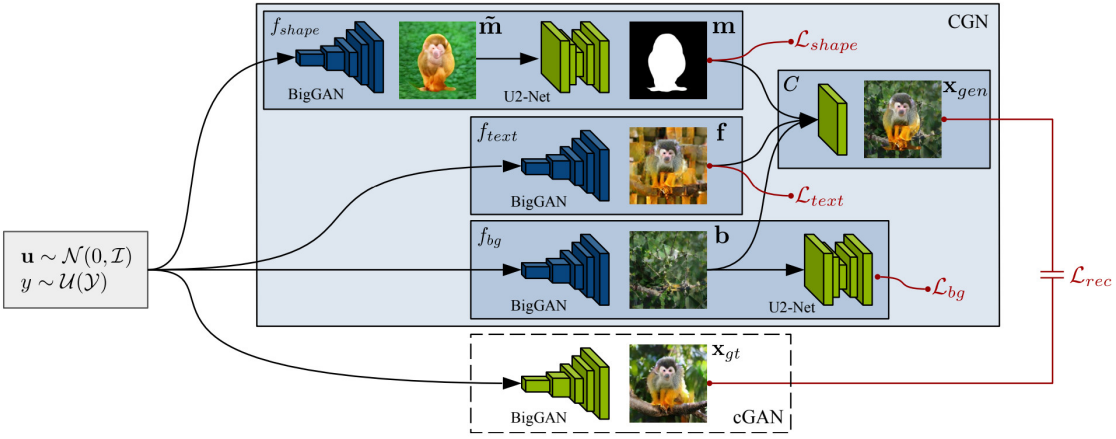
Causal Reasoning

Counterfactual Generative Networks

- Neural networks learn **spurious correlations** and often take **shortcuts**
- Can we learn to decompose image generation into **causal mechanisms?**
- This would allow to ask **counterfactual questions,** e.g.,
  "How would this image look like with a different background?"
- Training on counterfactual data increases classifier **robustness** to OOD data

Sauer and Geiger: Counterfactual Generative Networks. ICLR, 2021.

Counterfactual Generative Networks



Counterfactual Generative Networks

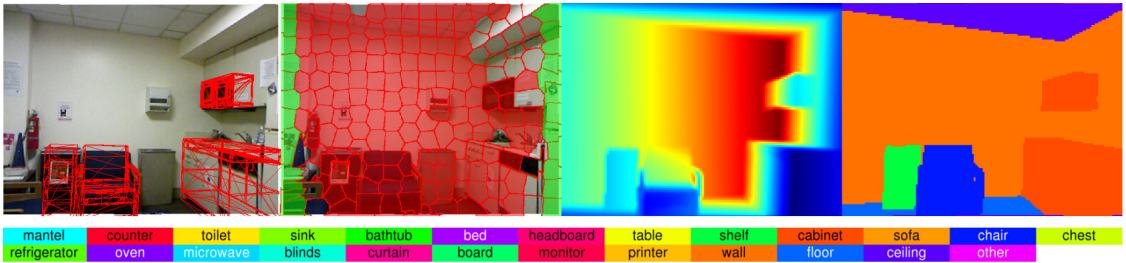
Shape	red wine	cottontail rabbit	pirate ship	triumphal arch	mushroom	hyena dog
Texture	carbonara	head cabbage	banana	Indian elephant	barrel	school bus
Background	baseball	valley	bittern (bird)	viaduct	grey whale	snorkel

Figure 4: **ImageNet Counterfactuals.** The CGN successfully learns the disentangled shape, texture, and background mechanisms, and enables the generation of numerous permutations thereof.

`https://autonomousvision.github.io/cgn/`

Holistic 3D Scene Understanding

Indoor Scene Understanding

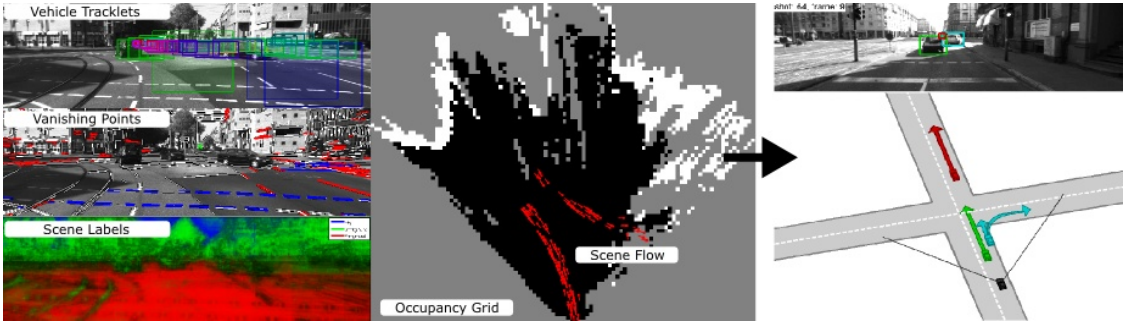


http://www.cvlibs.net/projects/indoor_scenes/

- Infer **3D objects and the layout** of indoor scenes from a single RGB-D image

Geiger and Wang: Joint 3D Object and Layout Inference from a Single RGB-D Image. GCPR, 2015.

Urban Scene Understanding



http://www.cvlibs.net/projects/intersection/

- Multi-object **traffic scene understanding** from movable platforms
- Reason about the 3D scene layout, the location of objects and traffic activities

Geiger, Lauer, Wojek, Stiller and Urtasun: 3D Traffic Scene Understanding From Movable Platforms. PAMI, 2014.

12.3
Human Body Models

Humans Interact through their Bodies

Generative Human Body Model




- Look like **real people**
- Deform like real people
- Small number of parameters
- **Easy to fit** to data
- **Easy to animate**

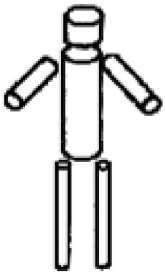
Pose Dynamics Parameters

Shape Texture

$$M(\theta, \beta, \delta, A; P)$$

3D Mesh

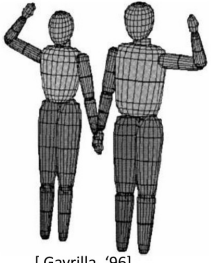
Early Body Models



Marr and Nishihara ’78

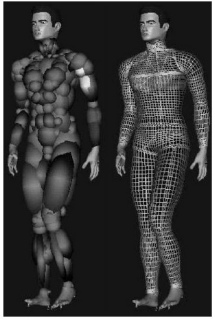
Nevatia and Binford ‘73

[Sminchisescu and Triggs ’03]



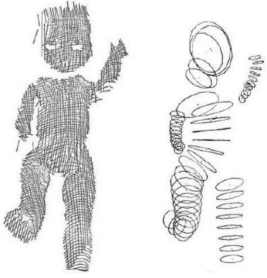
[Gavrilla, ‘96]

[Terzopoulos and Metaxas ’93]



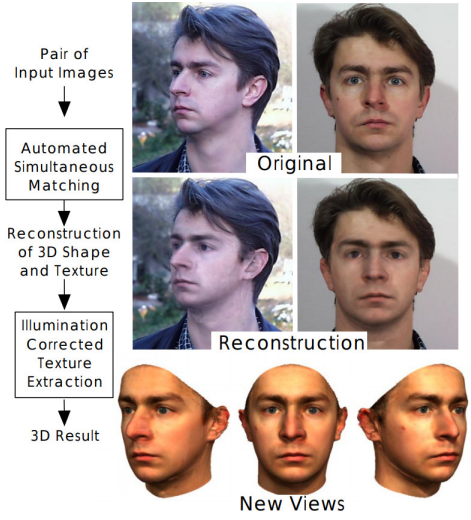
[Plänkers and Fua ’01]

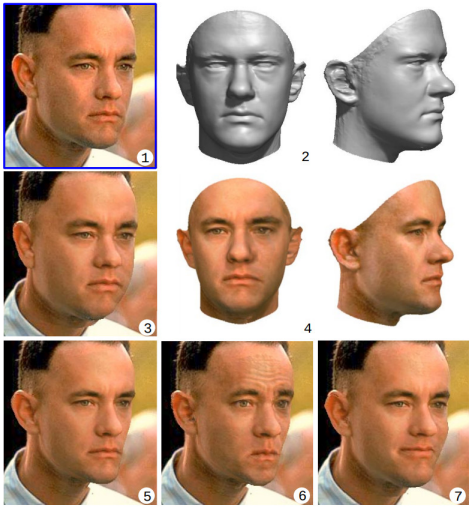
[Kakadiaris and Metaxas ’00]



Nevatia & Binford ’73

Inspiration: Morphable 3D Face Model





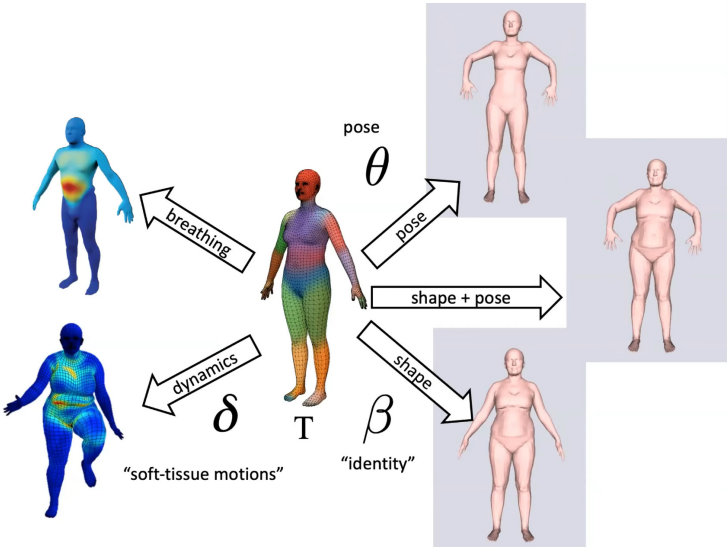
Blanz and Vetter: A Morphable Model for the Synthesis of 3D Faces. SIGGRAPH, 1999.

SMPL: A Skinned Multi-Person Linear Model

Input: Raw 3D Scans

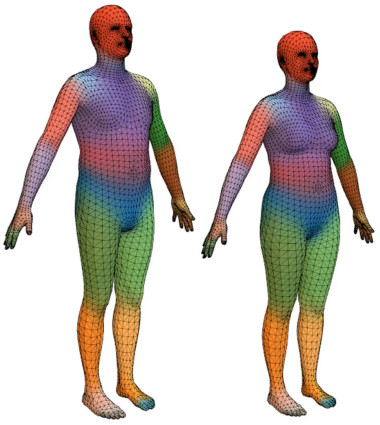
Loper, Mahmood, Romero, Pons-Moll and Black: SMPL: A skinned multi-person linear model. SIGGRAPH, 2015.

Output: Factored Model



Loper, Mahmood, Romero, Pons-Moll and Black: SMPL: A skinned multi-person linear model. SIGGRAPH, 2015.

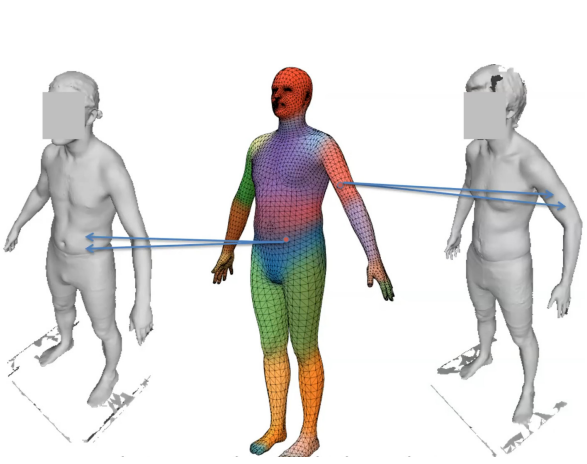
Representation: 3D Mesh



- SMPL uses a **3D mesh** as representation (vertices and faces)
- Base mesh designed by artist
- 7000 vertices $\Rightarrow$ **21000 numbers**
- Most settings of these numbers do not correspond to people or shapes

Loper, Mahmood, Romero, Pons-Moll and Black: SMPL: A skinned multi-person linear model. SIGGRAPH, 2015.

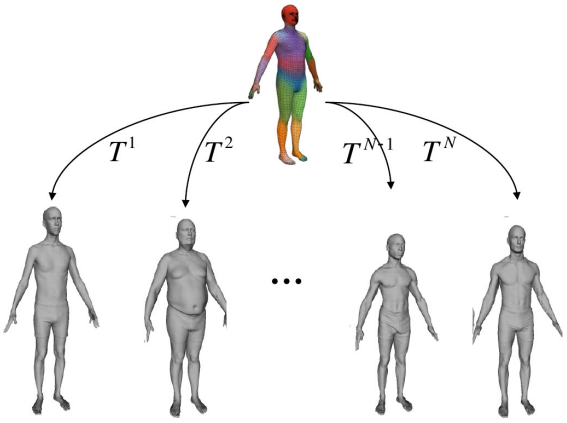
Data Preprocessing: Mesh Registration



- Raw scans are unordered and incomplete 3D points
- To get correspondences, a template mesh must be **aligned** with each scan
- This is a hard problem
- **Chicken-and-egg problem** (requires body model)
- Solution: solve model and registration **jointly**
- `http://faust.is.tue.mpg.de/`

Loper, Mahmood, Romero, Pons-Moll and Black: SMPL: A skinned multi-person linear model. SIGGRAPH, 2015.

Shape Deformation Subspace



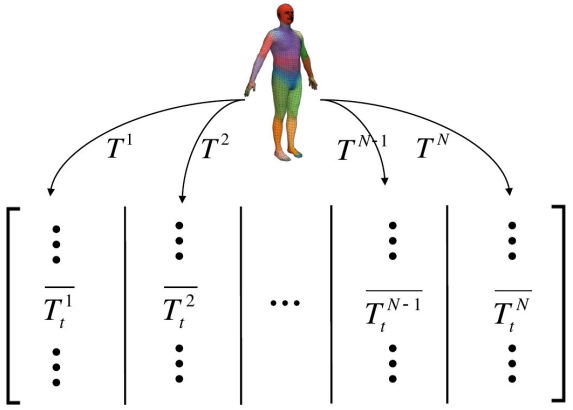
- Vectorize mesh vertices $\mathbf{T}$
- Subtract mean mesh (colored)
- Turn observations into matrix
- **Principal component analysis** (PCA) yields a linear model:

$$\mathbf{T} = \mathbf{S}\boldsymbol{\beta} + \boldsymbol{\mu}$$

- Obtain low-dimensional subspace (10D-300D) that captures most of the variance in the 21000D space
- This works due to canonical pose

Loper, Mahmood, Romero, Pons-Moll and Black: SMPL: A skinned multi-person linear model. SIGGRAPH, 2015.

Shape Deformation Subspace



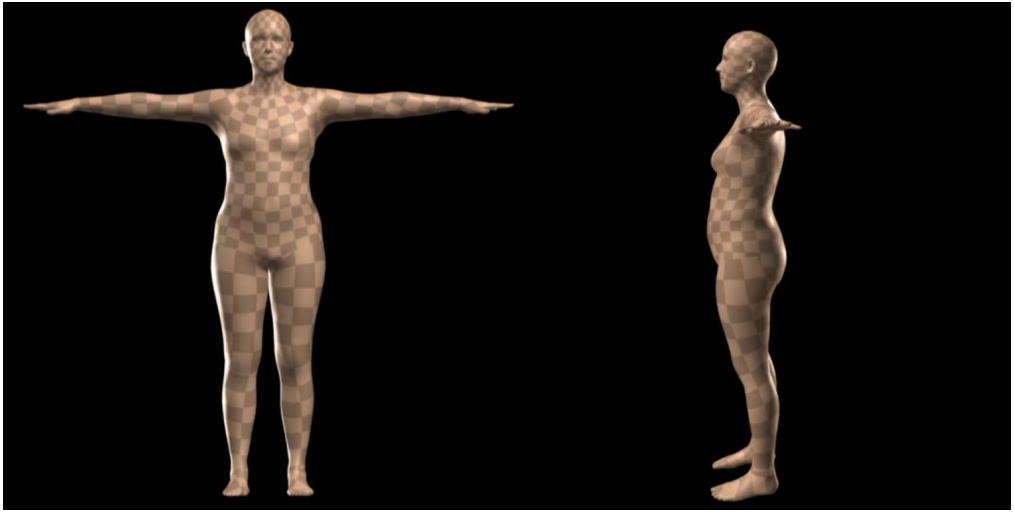
- Vectorize mesh vertices $\mathbf{T}$
- Subtract mean mesh (colored)
- Turn observations into matrix
- **Principal component analysis** (PCA) yields a linear model:

$$\mathbf{T} = \mathbf{S}\boldsymbol{\beta} + \boldsymbol{\mu}$$

- Obtain low-dimensional subspace (10D-300D) that captures most of the variance in the 21000D space
- This works due to canonical pose

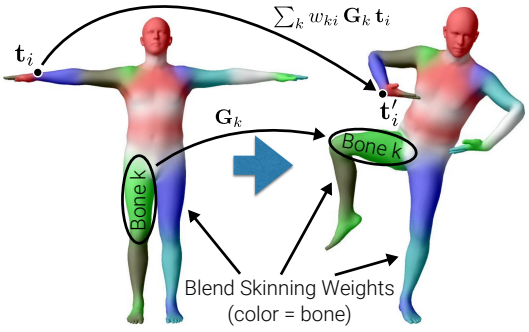
Loper, Mahmood, Romero, Pons-Moll and Black: SMPL: A skinned multi-person linear model. SIGGRAPH, 2015.

Shape Deformation Subspace



Loper, Mahmood, Romero, Pons-Moll and Black: SMPL: A skinned multi-person linear model. SIGGRAPH, 2015.

Linear Blend Skinning



- Vertices are **linear combination** of transformed template vertices

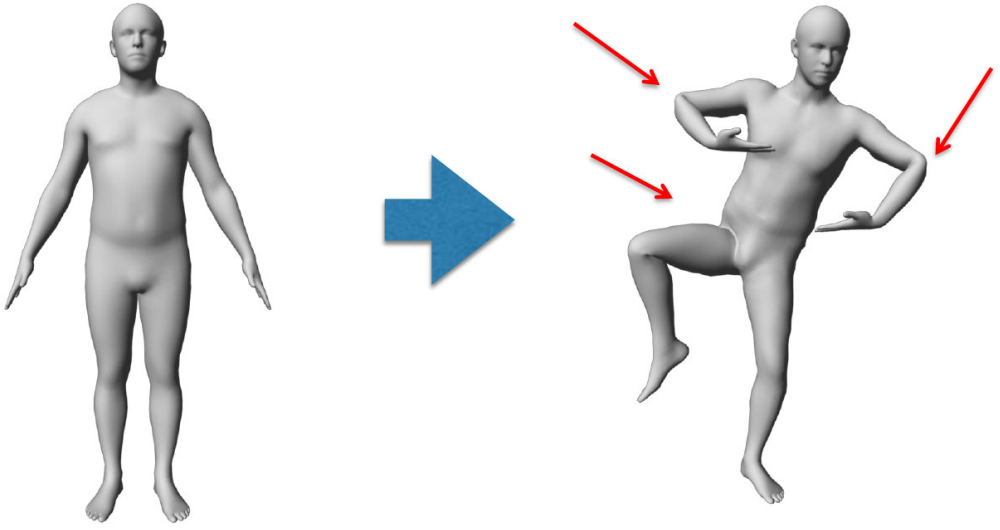
Linear Blend Skinning (LBS):

$$\mathbf{t}'_i = \sum_k w_{ki}\,\mathbf{G}_k(\theta, \mathbf{J})\,\mathbf{t}_i$$

- $\mathbf{t}, \mathbf{t}'$: Rest/transformed vertices
- w_{ki}: Blend skinning weights
- $\mathbf{G}_k$: Rigid bone transformation
- θ: Pose
- $\mathbf{J}$: Joint locations
- Skinning weights created by artist

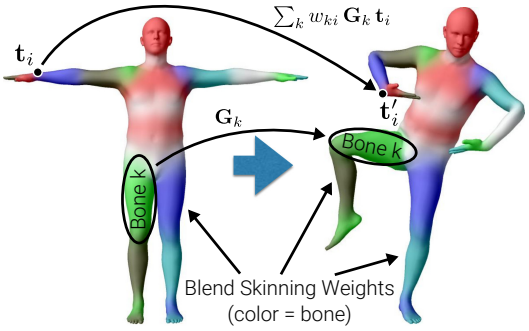
Loper, Mahmood, Romero, Pons-Moll and Black: SMPL: A skinned multi-person linear model. SIGGRAPH, 2015.

LBS Problems



Loper, Mahmood, Romero, Pons-Moll and Black: SMPL: A skinned multi-person linear model. SIGGRAPH, 2015.

SMPL



- Vertices are **linear combination** of transformed template vertices

LBS vs. SMPL:

$$\mathbf{t}'_i = \sum_k w_{ki}\, \mathbf{G}_k(\boldsymbol{\theta}, \mathbf{J})\, \mathbf{t}_i$$

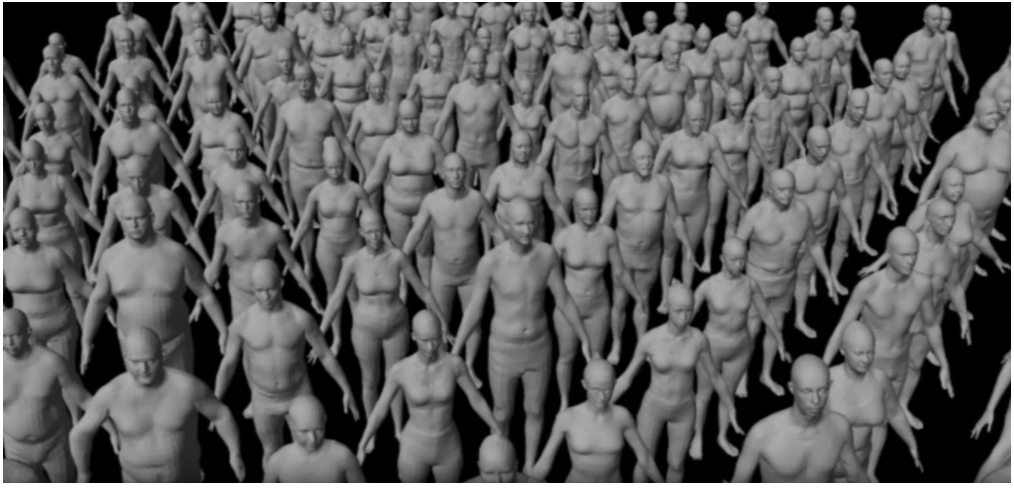
$$\mathbf{t}'_i = \sum_k w_{ki}\, \mathbf{G}_k(\boldsymbol{\theta}, \mathbf{J}(\boldsymbol{\beta}))\, (\mathbf{t}_i + \mathbf{s}_i(\boldsymbol{\beta}) + \mathbf{p}_i(\boldsymbol{\theta}))$$

3 key differences:

- Joints $\mathbf{J}(\boldsymbol{\beta})$ depend on shape $\boldsymbol{\beta}$
- Shape correctives $\mathbf{s}(\boldsymbol{\beta}) = \mathbf{S}\boldsymbol{\beta}$
- Pose correctives $\mathbf{p}(\boldsymbol{\theta}) = \mathbf{P}\boldsymbol{\theta}$

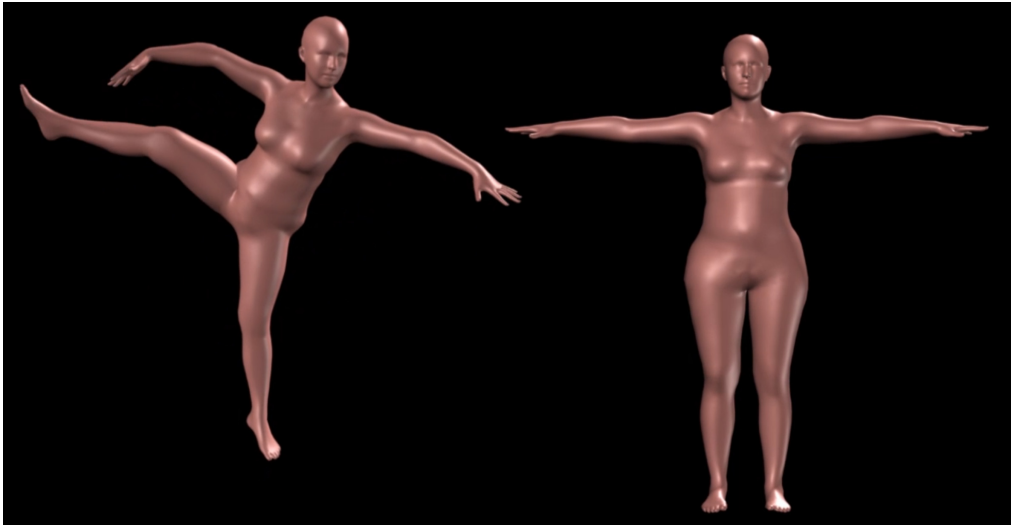
Loper, Mahmood, Romero, Pons-Moll and Black: SMPL: A skinned multi-person linear model. SIGGRAPH, 2015.

Learn Parameters from Registered Scans



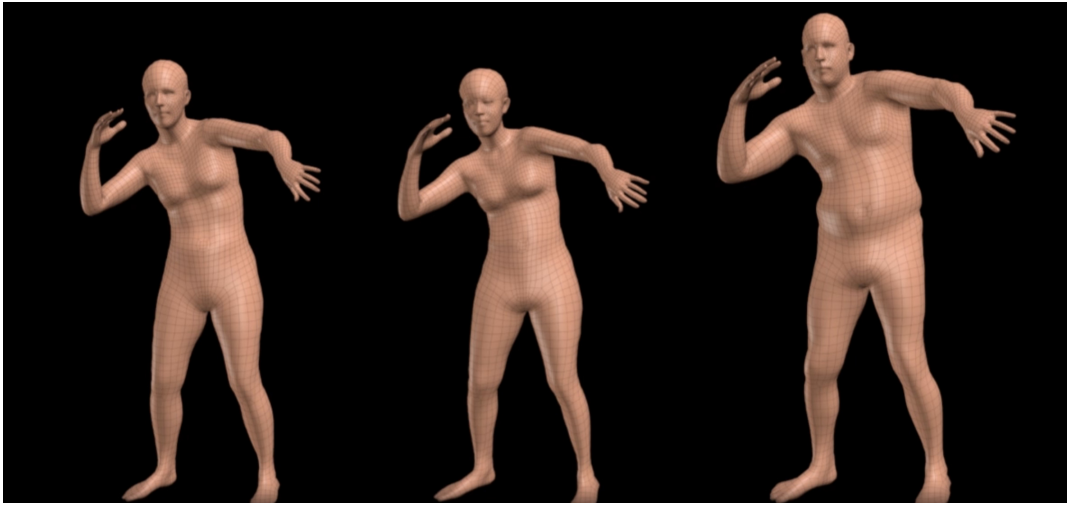
Loper, Mahmood, Romero, Pons-Moll and Black: SMPL: A skinned multi-person linear model. SIGGRAPH, 2015.

Pose Blend Shapes



Loper, Mahmood, Romero, Pons-Moll and Black: SMPL: A skinned multi-person linear model. SIGGRAPH, 2015.

SMPL Results

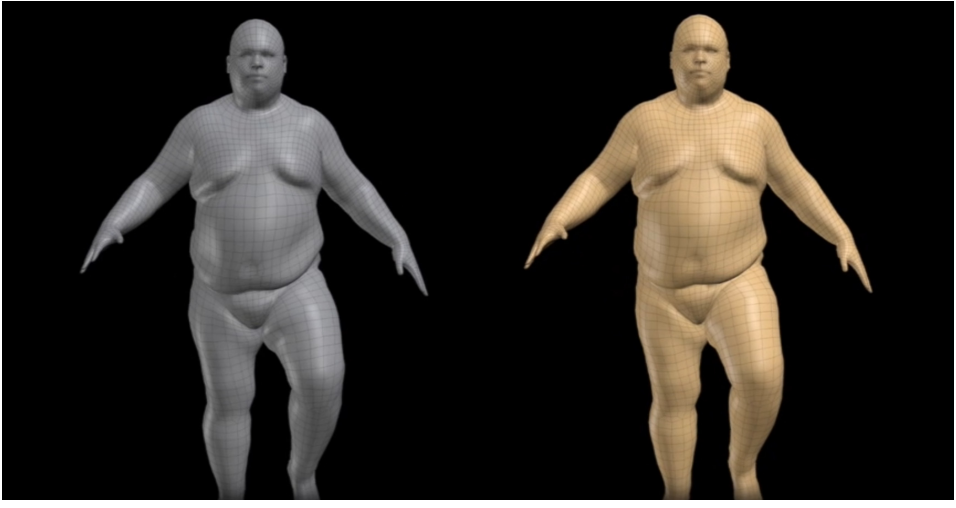


Loper, Mahmood, Romero, Pons-Moll and Black: SMPL: A skinned multi-person linear model. SIGGRAPH, 2015.

Character Retargeting using SMPL

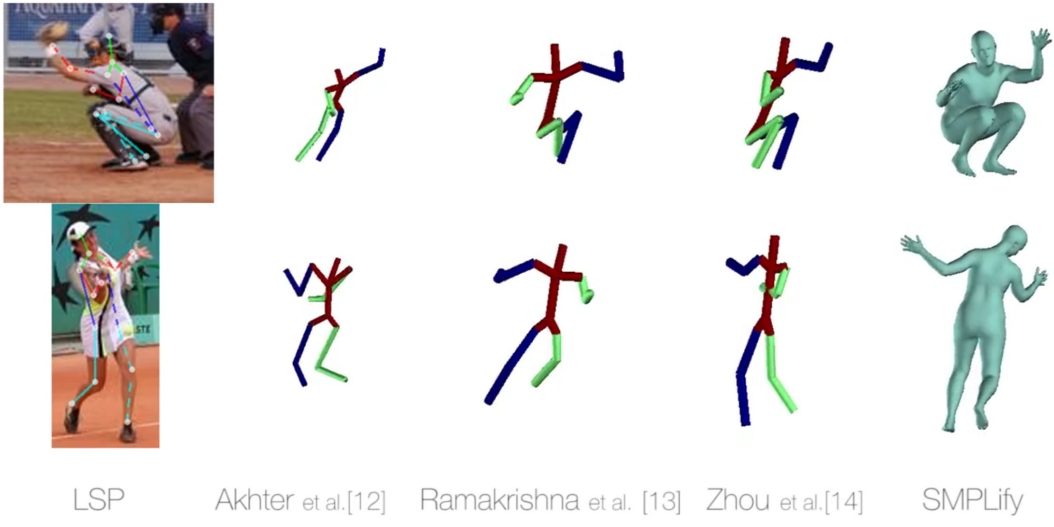
Loper, Mahmood, Romero, Pons-Moll and Black: SMPL: A skinned multi-person linear model. SIGGRAPH, 2015.

Dynamic SMPL (DMPL)



Loper, Mahmood, Romero, Pons-Moll and Black: SMPL: A skinned multi-person linear model. SIGGRAPH, 2015.

SMPLify: Estimation from a Single Image



Loper, Mahmood, Romero, Pons-Moll and Black: SMPL: A skinned multi-person linear model. SIGGRAPH, 2015.

Current Research Directions

- Clothing
- Garment
- Appearance
- Dynamics
- Few-shot
- From RGB
- Avatars
- Interaction
- Scenes

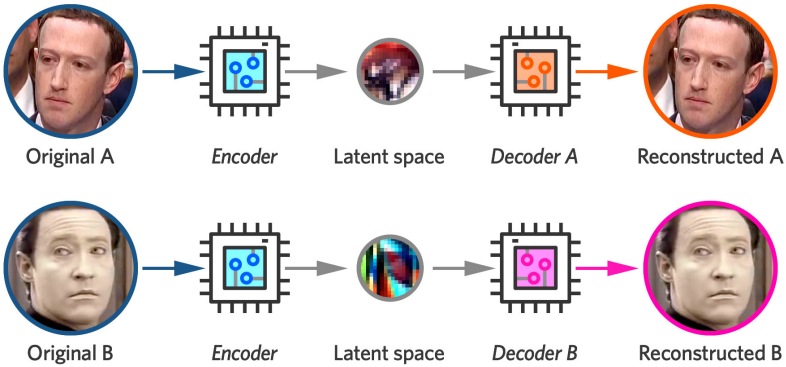
Alldieck, Magnor, Bhatnagar, Theobalt and Pons-Moll: Learning to Reconstruct People in Clothing From a Single RGB Camera. CVPR, 2019.

12.4
Deepfakes

What are DeepFakes?

https://github.com/deepfakes/faceswap

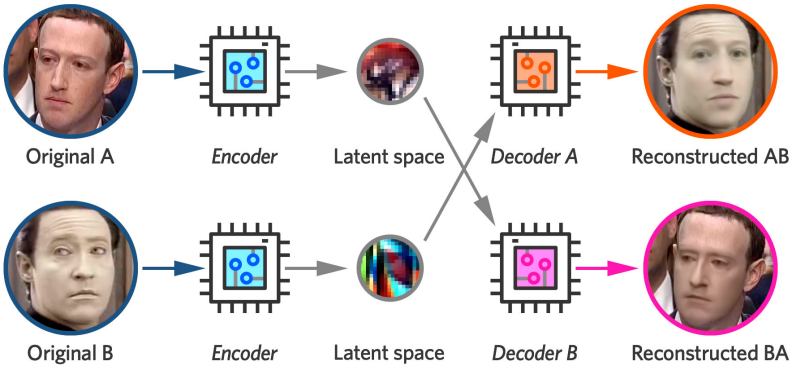
FaceSwap



- Extract faces in two videos, train auto-encoder for each of the two persons
- Shared encoder weights to enforce same latent space (pose, expression, lighting)

https://arstechnica.com/science/2019/12/how-i-created-a-deepfake-of-mark-zuckerberg-and-star-treks-data/

FaceSwap



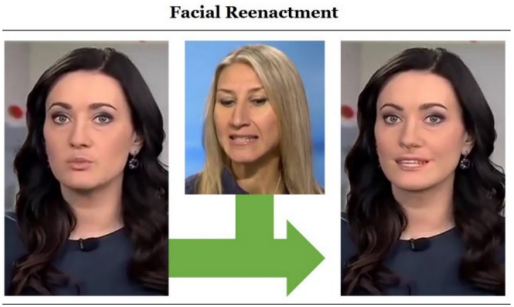
- Extract faces in two videos, train auto-encoder for each of the two persons
- Shared encoder weights to enforce same latent space (pose, expression, lighting)
- At inference time, swap decoders to decode latent code with new identity

https://arstechnica.com/science/2019/12/how-i-created-a-deepfake-of-mark-zuckerberg-and-star-treks-data/

FaceSwap

https://arstechnica.com/science/2019/12/how-i-created-a-deepfake-of-mark-zuckerberg-and-star-treks-data/

FaceSwap vs. Reenactment



- **FaceSwap:** Take face of video and copy it onto another video. Output is a hybrid between background (including hair) and new foreground (face).
- **Reenactment:** Take expression from one video and animate another person. Output is not a hybrid but manipulates expression of a person, enables animation.

Applications in Graphics and the Movie Industry

3D Model + Textures + Shading -> Synthetic Image

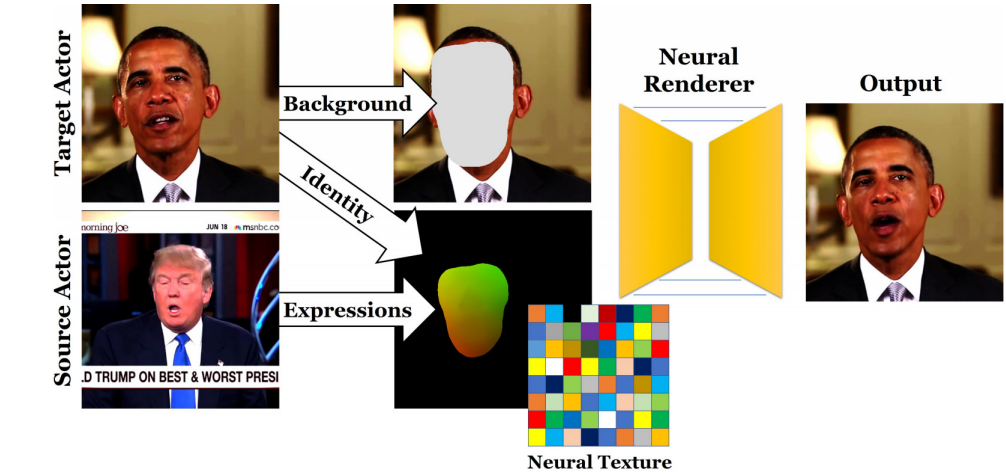
- Reenactment techniques have been used extensively in the **movie industry**
- But graphics methods require careful **3D model** creation and **manual editing**
- Learning has the potential to **automate this process** and enable richer editing

Face2Face

- **Real-time** facial reenactment, source video captured with commodity **webcam**
- **Face model** parameters recovered by non-rigid dense alignment wrt. image
- Track facial expressions of both source and target video for **appearance transfer**

Thies, Zollhöfer, Stamminger, Theobalt and Nießner: Face2Face: Real-Time Face Capture and Reenactment of RGB Videos. CVPR, 2016.

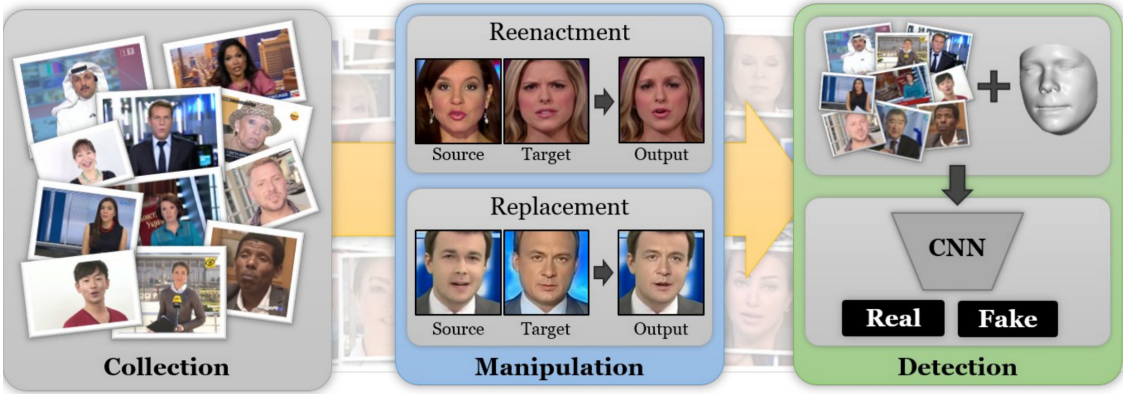
Deferred Neural Rendering



- Learn object-specific **neural textures** that can be “rendered” with a neural network

Deep Fake Detection

FaceForensics

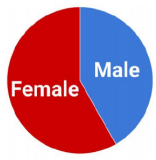


- Goal: Learn binary classifier to detect manipulated images

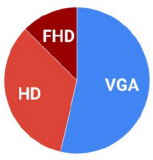
FaceForensics Dataset

Source: 1,000 Videos (510,529 frames)

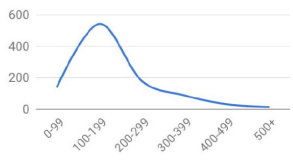
Methods	Train	Validation	Test
Pristine	366,847	68,511	73,770
DeepFakes	366,835	68,506	73,768
Face2Face	366,843	68,511	73,770
FaceSwap	291,434	54,618	59,640
NeuralTextures	291,834	54,630	59,672



(a) Gender



(b) Resolution



(c) Pixel Coverage of Faces

- Publicly available!
- Over 2 million manipulated frames
- Three compression levels for each manipulated frame
- Over 1000 research groups

Rössler, Cozzolino, Verdoliva, Riess, Thies and Nießner: FaceForensics++: Learning to Detect Manipulated Facial Images. ICCV, 2019.

ForensicTransfer

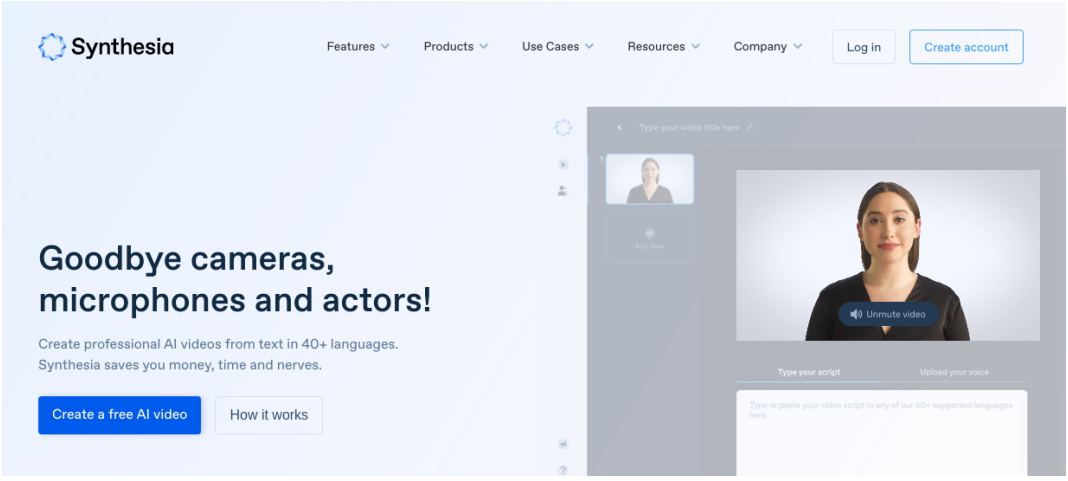
XceptionNet	Test on Face2Face	Test on FaceSwap
Trained on Face2Face	98.13%	50.20%
Trained on FaceSwap	52.73%	98.30%

Conclusions:

- DeepFake detection works well on in-domain data (i.e., knowing the forgery method)
- However, it doesn't generalize
- Requires self-supervised, transfer and few-shot learning techniques for OOD generalization

Cozzolino, Thies, Rössler, Riess, Nießner and Verdoliva: ForensicTransfer: Weakly-supervised Domain Adaptation for Forgery Detection. Arxiv, 2018.

Synthesia



https://www.synthesia.io/

Thank You!

All the best for the exam