

Lecture 12 - Diverse Topics in Computer Vision

开头，教授很开心的说刚刚得知本门课程已经被授予图宾根大学计算机科学与技术系计算机视觉课程的最佳讲座奖

同样，这也是本课程的最后一章节了（真的很有成就感）

- 在本门课程中，我们已经学了很多东西，比如下面这些主题

- ▶ 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

- 从 CV 历史，到基本的几何、图像形成、相机校正、三维重建、双目立体、结构预测、光流、阴影塑形、多视图重建、体积融合、隐式神经表征、可微分渲染的视图合成、各种深度学习方法等...

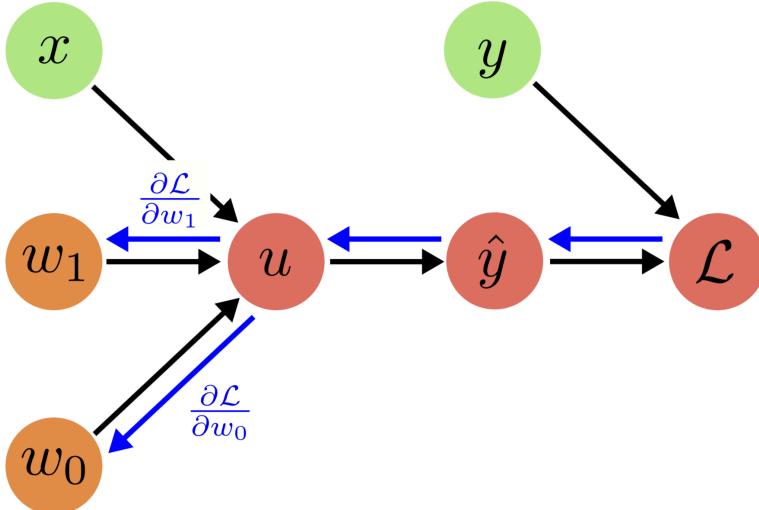
- 与此同时，还有很多同样属于 CV 领域的主题没有提到

- ▶ 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

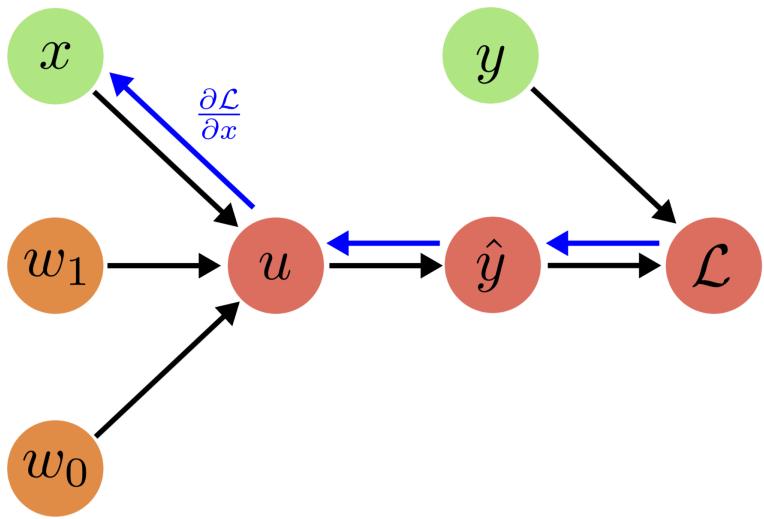
- 在最后一个章节概括所有的这些主题是不可能的——因此，教授决定在最后一个章节选择一些有突破性的领域、大型的子领域（图中标红的内容），进行高抽象层次的讲述
- 最后，致谢制作该课件的一些资源，我也原封不动的放在这里
 - **Justin Johnson** — Convolutional Neural Networks + Neural Style Transfer
http://web.stanford.edu/class/cs20si/lectures/slides_06.pdf
 - **Leon Gatys** — Image Style Transfer Using Convolutional 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=rzpiSYTrRUO>
 - **Matthias Niessner** — Deepfakes Creation and Detection
<https://www.youtube.com/watch?v=Xv2IRs2-KA>

12.1 Input Optimization

- Weight Optimization vs Input Optimization



- 这是一个简单的神经网络的计算图，橙色标签为权重，绿色标签为输入，红色标签的 \mathcal{L} 比较 \hat{y} 和 y 的差别，然后反向传播更新权重； u 为中间计算值，可能是 x 和权重的线性组合
 - 因此，这种方式的优化目标是权重
- 但是实际上，我们也可以不反向传播给权重——假设我们有一个预先训练好的网络，优化到某个状态时，然后我们根据 y 和 \hat{y} 的 loss 反向传播来调整 x 的值，如下图

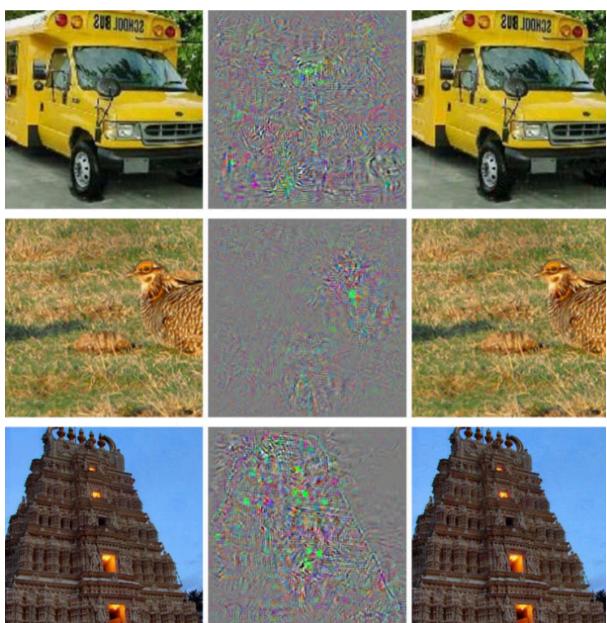


- 也即，计算图结构不变，我们将优化的目标从权重改为了输入——因此称这一类想法为输入优化 (input optimization) 问题
- 其典型案例包括 **Adversarial attacks** (对抗攻击) 和 **style transfer** (风格迁移) ，我们接下来介绍这两个案例

Adversarial Attacks

在 2014 年左右，人们还沉浸于深度学习网络的实用性，以及为很多领域带来的突破性进展——此时，有一篇文献，名字大概为“神经网络的有趣特性”，首次提出了，这些神经网络都有一个根本上的缺陷

- Adversarial Attacks on Image Classification



- 人们发现，假设我们有一个训练好的神经网络，可以正确将左侧第一列的图像识别为校车，然后我们在上面对每个像素增加一个非常轻微的噪声（中间的图只是夸张的示例）得到右侧的图。从我们的角度来看，它们没

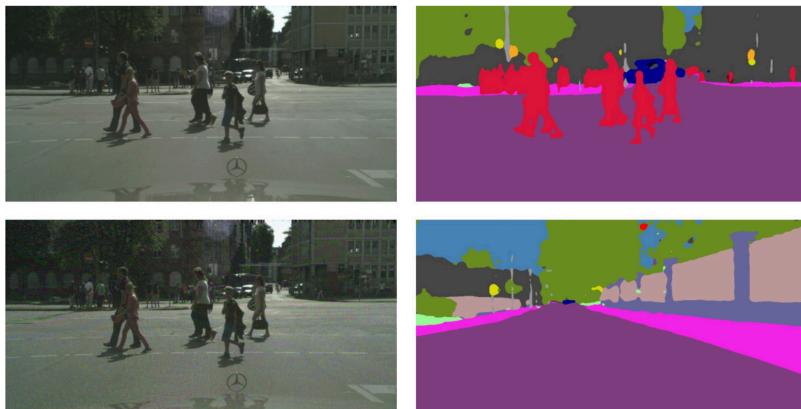
有什么区别，仍然是校车——可是此时神经网络会 99% 确信它是一只鸵鸟

- 我们将这个过程数学化如下，给定一个分类器 $f: \mathbb{R}^m \rightarrow \{1, \dots, L\}$ ，它将 m 个像素的图像分类到一个维度为 L 的分类空间中，然后我们为图像 x 找到它的对抗样例如下 $\$$

$$x + \underbrace{\Delta x}_{\{\arg\min\}} \text{ s.t. } \| \Delta x \|_2 : f(x + \Delta x) = y_t$$

- 其中 y_t 是特征空间中距离正确样例最近的一个错误样例，我们在样本 x 的

- 同样，这种现象也会发生在语义分割中



- 左列的图，仅仅添加了一点噪声，语义分割的结果就截然不同

接着，人们想知道这是否真的是一个问题——我们想要攻击一个分类器，就需要对输入的每一个像素进行修改；但是对于整个系统而言，我们并不具有访问权限——否则我们可以直接修改模型的输出，从而进行破坏

- Physical Adversarial Attacks [Lu, Sibai, Fabry and Forsyth: NO Need to Worry about Adversarial Examples in Object Detection in Autonomous Vehicles. Arxiv, 2017](#)



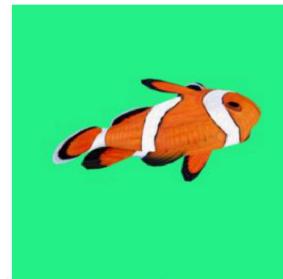
- 有一篇具有争议性的论文，题为“无需担心自动驾驶汽车物体检测中的对抗性样本”
- 他们在许多实验，特别是自动驾驶汽车的场景下经过实验后得出结论——对抗攻击方法在应用于停车标识时，只有精心选择的情况下才可

以攻击生效，因此我们不需要担心这些攻击的发生，特别是在自动驾驶场景下

- 他们试图说服整个社区，这并不是一个问题，至少在这些早期方法的支持下，实验表明似乎我们并不需要担心这个问题

然而，很快有团队发现，似乎仍然有一些存在于物理世界的攻击情况，我们不得不关注

- Robust Adversarial Attacks [Athalye, Engstrom, Ilyas and Kwok: Synthesizing Robust Adversarial Examples. ICML, 2018.](#)

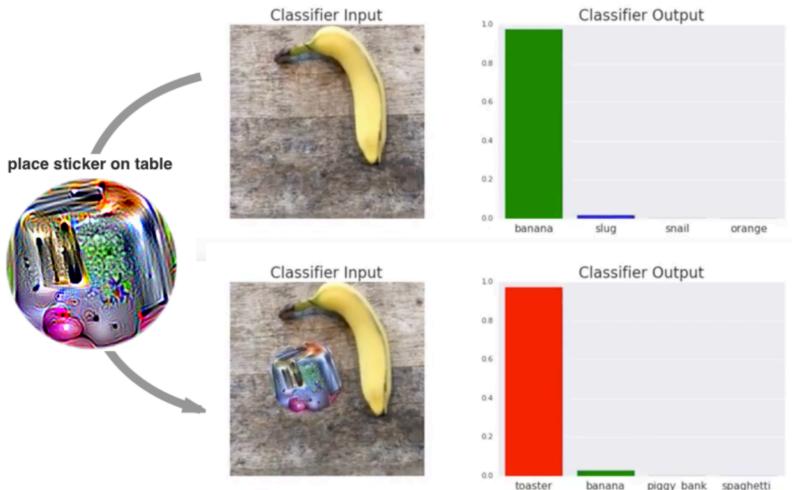


- 证明了现实世界中也有鲁棒的对抗攻击存在
- 他们通过最大化变换后期望分类错误的概率 (expectation over transformation, EOT) \$\$\arg\max_{x'} \mathbb{E}_{t \sim \mathcal{T}} [\log P(y_t | t(x')) - \lambda \|t(x') - t(x)\|_2]

—其中 $t \sim \mathcal{T}$ ，对输入数据 x' 应用的随即变换，从变换分布 \mathcal{T} 中采样 — $\$$



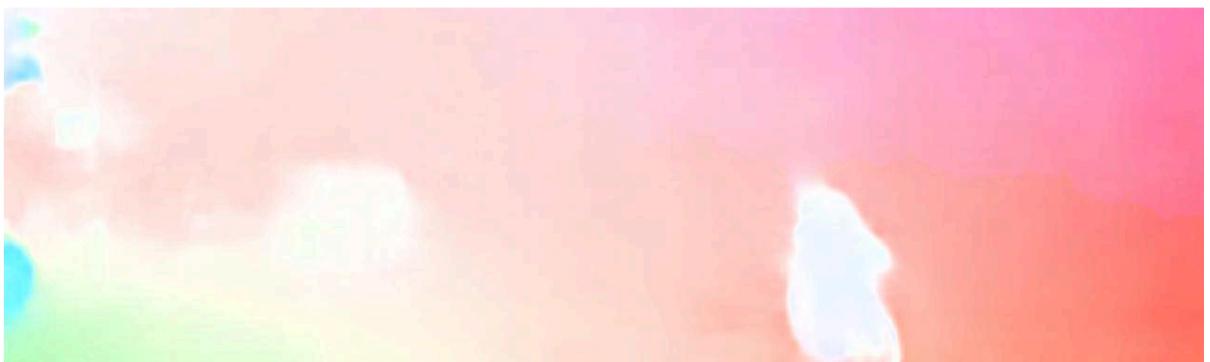
- 也有其他的文献针对于之前的停车标志做了测试——你可以通过在标识上涂鸦，或者贴上一些贴纸就可以起到欺骗模型的效果，因此这种危险性仍然是不可忽视的
- Adversarial Patch Attacks [Brown, Mane, Roy, Abadi and Gilmer. Adversarial patch. Arxiv, 2017](#)



- 在此基础之上，有一种想法称为 patch attacks，同样使用 EOT 的想法来生成。我们试图设计一个 patch，使得它可以插入许多图像中来欺骗分类器——当然，这个 patch 如果你仔细看是一定可以看到的，不过一般来说攻击者会将 patch 做的很小，并且隐藏在图中的某个区域
- 这种方法很容易应用在现实世界（只需要打印出来，然后贴在物体上即可）
- Attacking Optical Flow [Ranjan, Janai, Geiger and Black: Attacking Optical Flow. ICCV, 2019] (Ranjan, Janai, Geiger and Black: Attacking Optical Flow. ICCV, 2019)



- 例如这里有一个 optical flow 的例子，从上面这个视频中截取相邻的两帧，然后利用一个成熟的网络，例如 FlowNet2 预测其光流图如下



- 而当我们添加一个整幅图占比不到 1% 的 patch 后



- 模型的输出变成了这样



- 这是怎么做的呢？

- 先做如下假设

- 令 $F(I, I')$ 表示一个光流网络（输入时间上相邻的两帧图像），用 \mathcal{I} 表示相邻帧对的数据集 $\{(I, I')\}$

- 令 $A(I, p, t, l)$ 表示图像 I 在位置 l 添加了 patch p 的一个变换 t

- 令 τ 表示一个二维空间上仿射变换的分布

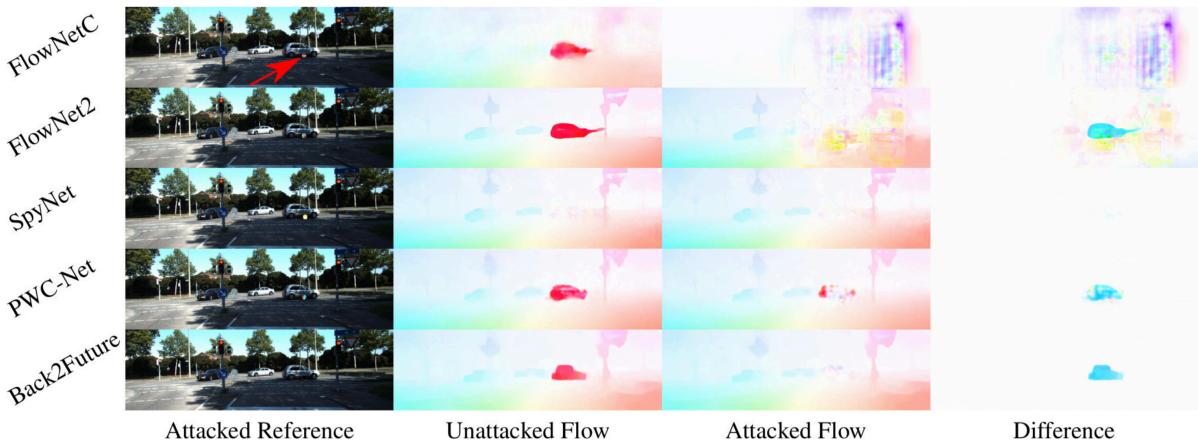
- 令 \mathcal{L} 表示图像域上的一个均匀分布

- 我们的目标是寻找到一个 patch \hat{p} , 使得 \$\$

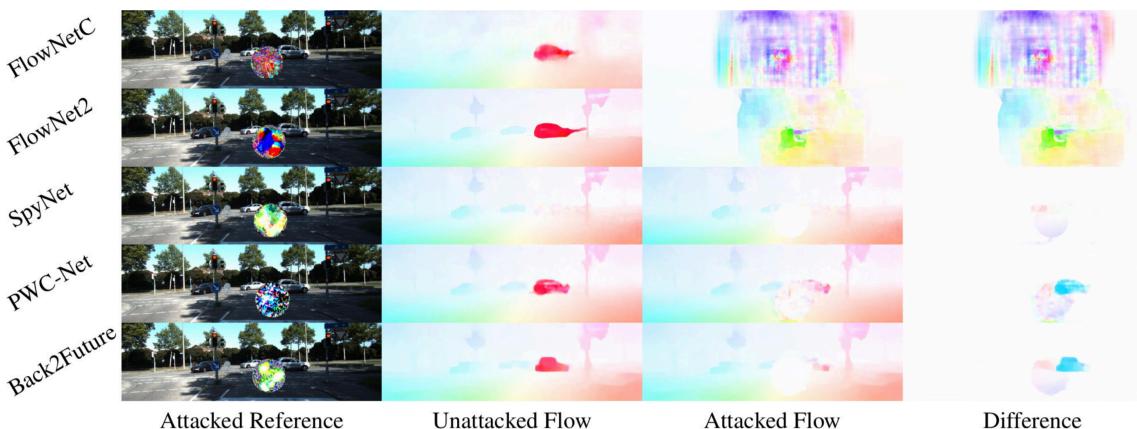
$$\begin{aligned} \hat{p} &= \arg \min \{p\} \mathbb{E}\{(I, I') \sim \mathcal{I}, t \sim \mathcal{T}, l \sim \mathcal{L}\} \\ &\left[\frac{\langle u, v \rangle \cdot \tilde{u} \cdot \tilde{v}}{\|\tilde{u}\|_2 \|\tilde{v}\|_2} \right] \\ &\text{with } (u, v) = F(I, I'), \\ &\langle \tilde{u}, \tilde{v} \rangle = A(l, p, t, l), A(l', p, t, l)). \end{aligned}$$

—换句话说，我们想要找到一个与现有的光流方向都相反的patch

- White-Box Attacks

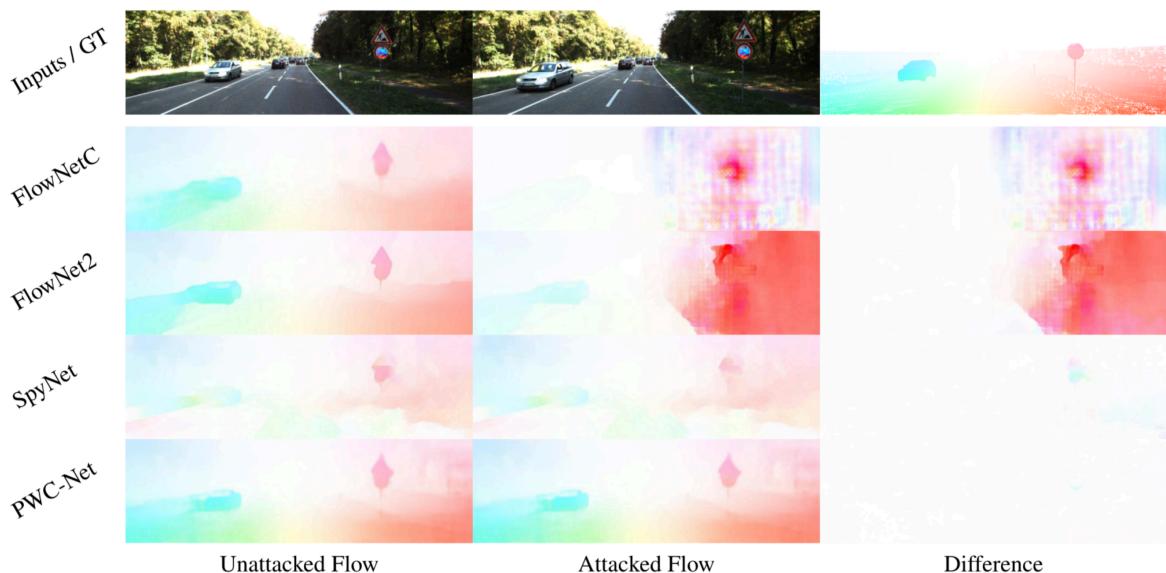


- 我们称这种 patch 设计为白盒攻击——也即已知想要干扰的图像，有针对性地设计攻击 patch
- 如图所示，如果 patch 太小，它可能超出编码器-解码器捕捉特征的区域，因此对一些模型不会造成太大影响

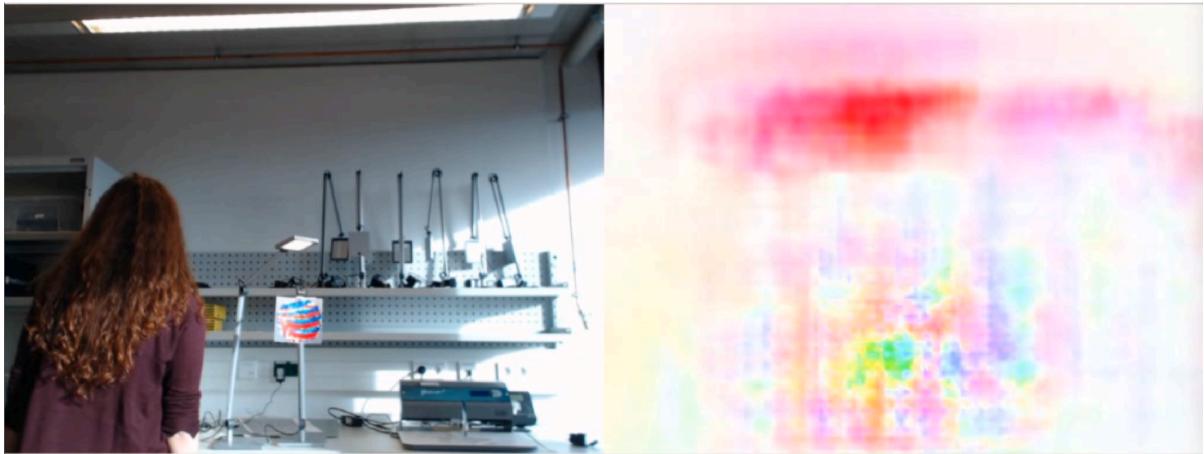


- ■ 如果我们放大这个补丁，可以看到它的影响效果变大

- Black-Box Attacks Setting



- 与之对应的还有黑盒攻击——也即我们不知道待干扰图像内容是什么样的，我们希望设计一个更具通用性的攻击 patch
- 在上图中，我们试图针对 FlowNet2 和 PWCNet 模型优化 patch，攻击效果如图所示
- Real-World Attack



- 可以看到这种 patch 式的攻击很容易应用在现实世界中——我们只需要将其打印出来贴在背景中，就可以扰乱模型的输出

通过上述内容可以看到，对抗攻击确实是一个令人担心的问题，因此也诞生了抵抗对抗攻击的领域

- Defenses against Adversarial Attacks
 - 整个领域是围绕着抵抗对抗攻击而逐渐进步的——攻击和反攻击两个领域此消彼长，互相推动彼此的进步
 - 主要有以下几种方法形成的子领域
 - **Gradient Masking/Obfuscation** [Papernot, 2016] [Song, 2017] [Buckmann, 2018]
 - 由于大多数攻击算法都是基于模型的梯度信息设计的，因此尝试遮盖或隐藏模型的梯度，从而使得攻击者无法设计有效的攻击 patch
 - **Robust Optimization** [Goodfellow, 2014] [Madry, 2017] [Hein, 2017]
 - 通过重新学习神经网络分类器的参数来增强网络的鲁棒性，使其能够正确分类对抗样本，通常结合“对抗训练”技术（在训练中加入对抗样本，让模型学会识别和防御它们）
 - **Adversarial Example Detection** [Carlini, 2017] [Grosse, 2017] [Metzen, 2017]
 - 通过分析自然样本和对抗样本的分布差异，检测并拒绝对抗样本的输入——开发对抗样本的检测算法

这些子领域也诞生出了很多的论文，在这里是指做一个高度的概括，以及对应子领域具有代表性的论文

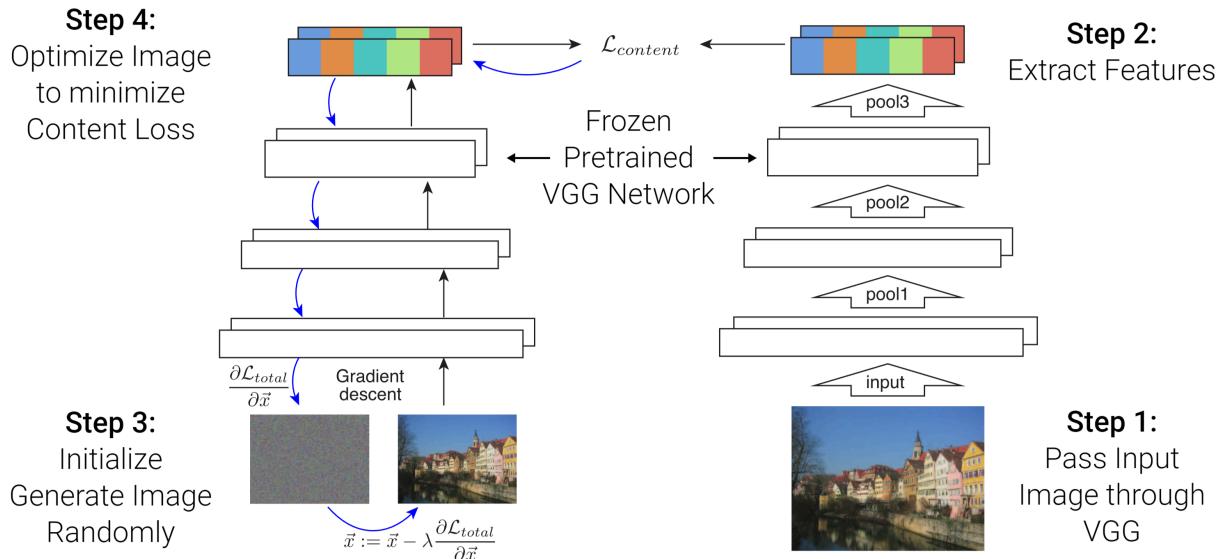
Neural Style Transfer

这是与对抗攻击完全不同的一种研究方向——它试图通过优化输入样本来生成美丽的图像

- Neural Style Transfer [Gatys, Ecker and Bethge: Image Style Transfer Using Convolutional Neural Networks. CVPR, 2016](#)

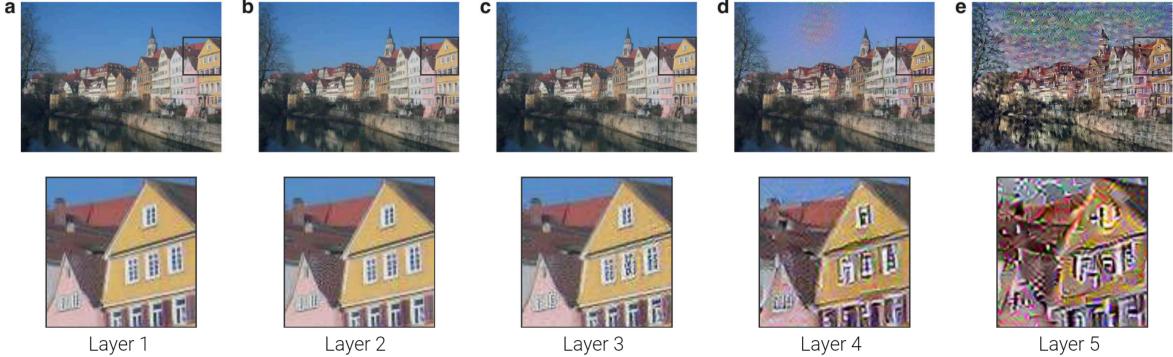


- 如图所示，我们试图将一幅图的内容和一幅图的风格合成在一起
- 如何实现这一点呢？总的说，我们从一个随机的噪声图像开始，然后分别优化 content 和 style 的 loss $\mathcal{L} = \mathcal{L}_{content} + \mathcal{L}_{style}$
- 下面分为 Content Reconstruction 和 Style Reconstruction 两部分来讲述上图这个例子
- Content Reconstruction



- 我们预先在 ImageNet 上训练好一个 VGG 网络，作为我们的特征提取器

- 首先将图像输入 VGG 网络（冻结权重，不更新它），随着层数增加，图像的特征被提取出来，然后我们保留最后提取出的特征
- 然后我们随机初始化一个噪声图像，通过 $\mathcal{L}_{content}$ 不断优化它，使得它在内容角度上的特征不断接近我们输入的 content 图像



- 随着 VGG 层数增加，细节逐渐丢失，其风格、整体细节结构似乎也逐渐丢失；我们需要权衡这两点，谨慎选择一层（例如在这里，似乎选择第 4 层会比较好——其整体结构和风格丢失，但是像素的基本统计属性仍然保留）

- 我们设计 content loss 如下 \$\$

$$\mathcal{L}_{content}(I) = \|\textbf{F}(I) - \textbf{F}(P)\|^2$$

- 也即最小化生成图像在第 1 层的特征 \textbf{F}_1 与 content 图像

$$G_{ij} = \sum_p \lim_{\Omega} \int_{\Omega} F_i^p F_j^p$$

$$(\textbf{F}_p) = \textbf{F}_i \textbf{F}_j^T$$

- 也即其本质就是第 i 个特征图和第 j 个特征图的内积， $G \in \mathbb{R}^{C \times C}$ ，其中

$$\mathcal{L}_{style}(I) = \|\textbf{G}(I) - \textbf{G}(A)\|^2$$

- 也即生成图像的 Gram 矩阵 \textbf{G}_1 和 style 图像的 Gram 矩阵

补充：这里的 content 图像就是 Tubingen 大学的河畔

补充 2：关于 Gram 矩阵，它如何可以表示图像的纹理特征呢？我的理解是，纹理特征可以理解为图像中某些区域和某些区域的相关性——它们可以是同时更换颜色，可以同时移动位置，可以同时放大缩小，但是必须始终保持关联；整幅图的所有具有这种关联的一组一组的区域组成的拓扑就是这幅图的“纹理特征”

12.2 Compositional Models

Compositional Models，尝试将模型视为一个个组件，通过研究它们的组合结果来实现更加强大的模型

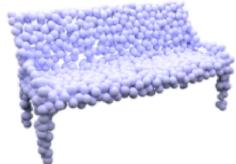
Shape Abstract

其中一个我们很感兴趣的领域称为形状抽象 (Shape Abstract)，特别是无监督方法的 3D 形状抽象

- 3D Representations [Paschalidou, Ulusoy and Geiger: Superquadrics Revisited: Learning 3D Shape Parsing Beyond Cuboids. CVPR, 2019.](#)



Voxels



Points



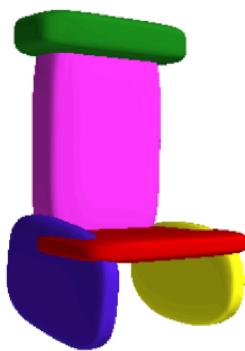
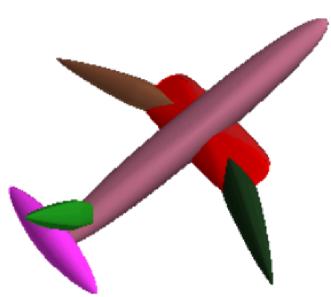
Meshes



Implicit

- 这里有很多我们之前已经讲过的 3D 对象表示方式——体素、点云、网格、隐式表示等
- 然而这些现存的 3D 表示方式存在一些局限：
 - 依赖大量几何信息，而不包含结构化信息
 - 无法传递语义信息（如物体的部分结构、功能性等）

- 3D Primitives

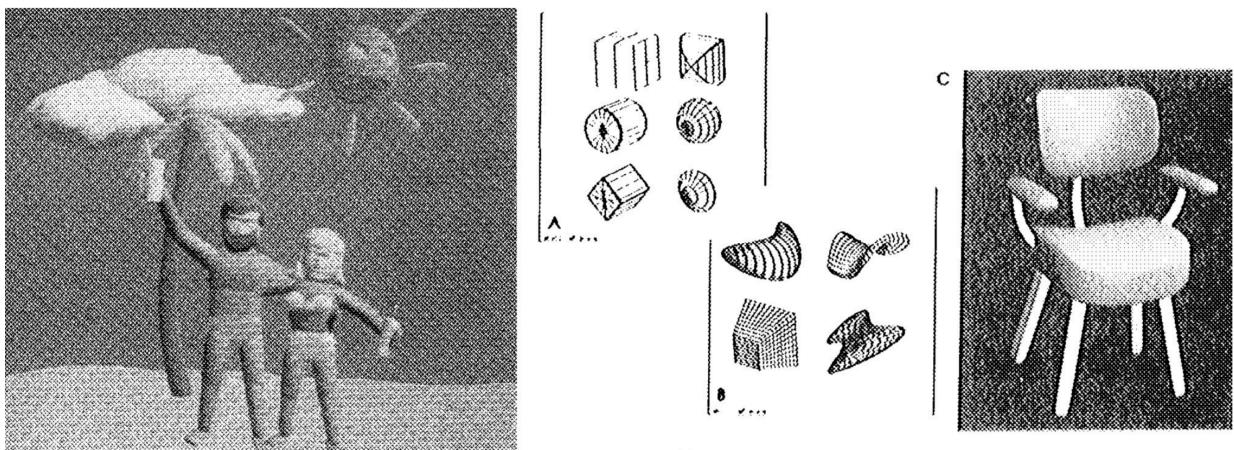


- 因此，有一些基于基元的 3D 表示方法
 - 使用少量的几何基元即可表示复杂的 3D 对象
 - 可以传递语义信息（如部分结构和功能等）
- 挑战：
 - 基元的种类和数量该如何确定？不同 3D 物体可能需要不同种类的基元，设计具有通用性的基元十分困难

- 需要大量带有几何和语义标注的数据集，但现有的标注数据集有限
- 因此，我们希望利用无监督的方法来实现将一个物体的 3D 表示转换为其基元表示
- 这项工作被称为 3D Shape Abstraction

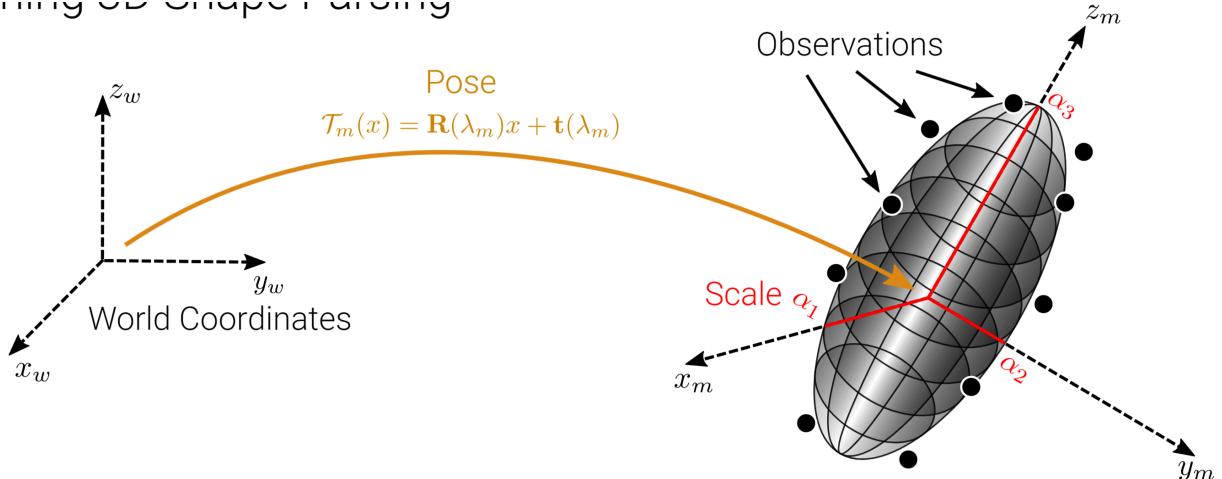


- 学习 3D 形状的抽象
- 推断使用基元的数量
- 在基元层次不使用监督方法，直接使用点云作为标签
- 输入：点云或者一幅图像
- Pentland's Superquadrics Revisited



- 这是来自 MIT 的 Pentland 学院的一个研究团队在 1986 年提出的想法——超二次曲面 (superquadrics)
- 1 个超二次曲面只需要 11 个参数（包括形状、尺度、旋转等），即可生成一个三维形状——图中左侧的场景如果用超二次曲面表示，其存储体积甚至只需要 1000 个字节大小
- 然而，在当时这种方法并不能很好的处理现实世界的数据，表现出鲁棒性很差的问题——但是深度学习可以从大量数据中提取特征，因此这种想法在今天得以实现

- Learning 3D Shape Parsing [Paschalidou, Ulusoy and Geiger: Superquadrics Revisited: Learning 3D Shape Parsing beyond Cuboids. CVPR, 2019.](#)



- 具体来说，对应每个超二次曲面

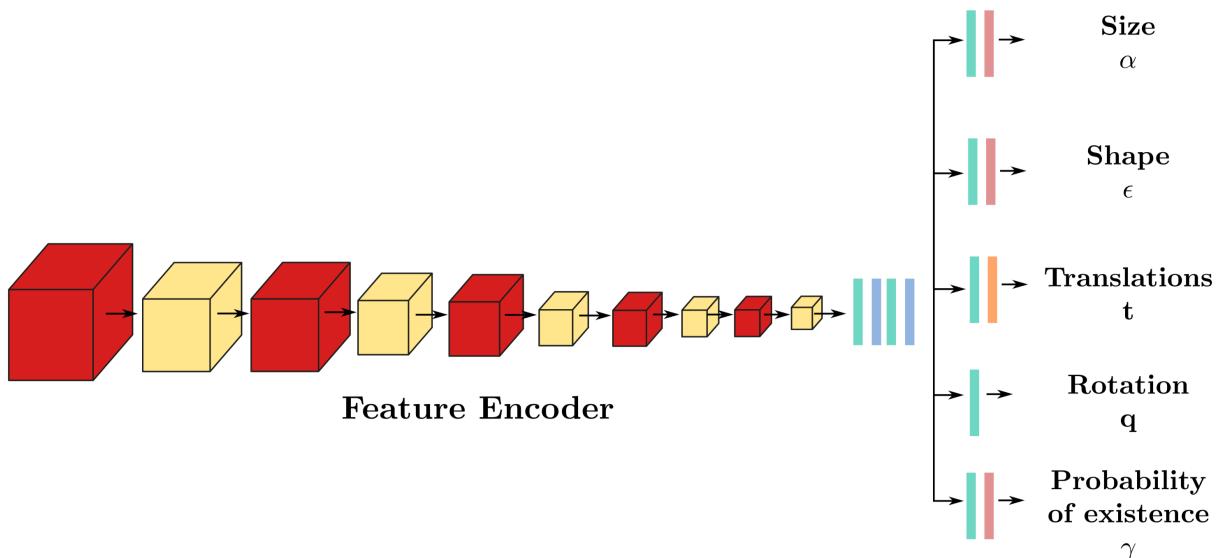
- 11个参数包括：pose (\mathbf{R}, \mathbf{t}) 6个 + scale (α) 3个 + shape (ϵ) 2个

- 有一个对应的存在概率 $\gamma \in [0, 1]$ ，表示该超二次曲面是否存在

- 由世界坐标系 (x_w, y_w, z_w) 转换为形状所在的局部坐标系 (x_m, y_m, z_m) 过程如下：

- pose 变换 $T_m(x) = \mathbf{R}(\lambda_m)x + \mathbf{t}(\lambda_m)$
- 然后由三个维度的 α 来描述其缩放尺度
- ϵ 描述其弯曲程度

- Network Architecture



- 整体的网络架构如下

- 通过一系列卷积层逐步提取输入的点云数据的几何特征
- 编码后的特征被映射到多个输入 head，用于预测不同的参数
- Size

- 形状的尺度参数，描述形状在不同方向的大小（如长、宽、高）
 - Shape
 - 形状的弯曲特性参数，用于确定超二次曲面的形状类型（如接近球形、柱状或盒状）
 - Translations
 - 平移参数，用于定义形状在世界坐标系中的位置。
 - Rotation
 - 旋转参数，用于定义形状的方向
 - Probability of existence
 - 存在概率，用于判断形状基元是否存在
 - Loss Function \$\$
- $$\mathcal{L}(P, X) = \mathcal{L}_{P \rightarrow X}(P, X) + \mathcal{L}_{X \rightarrow P}(X, P)$$
- $$+ \mathcal{L}_{\gamma}(P)$$

- 其中 P 表示预测的基元集合、 X 表示输入的点云数据 - 损失函数的组成 -

1. 单个形状基元的损失（局部损失）

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

- K : 是形状基元 m 上的采样点数目
 - Δ_k^m : 表示点云的点 x_i 到基元表面上最近点 y_k^m 的距离 \$\$
- $$\Delta_k^m = \min_{i=1, \dots, N} \| \mathbf{T}_m(\mathbf{x}_i) - \mathbf{y}_k^m \|_2$$
- $-\mathbf{T}_m(\mathbf{x}_i)$: 将点云中的点 \mathbf{x}_i 转换到形状基元 m 的坐标系中 - \mathbf{y}_k^m : 形

2. 整个场景的损失 \$\$

$$\mathcal{L}_{P \rightarrow X}(P, X) = \mathbb{E}_{p(z)} [\left(\sum_m z_m \mathcal{L}_{P \rightarrow X}^m(P, X) \right)]$$

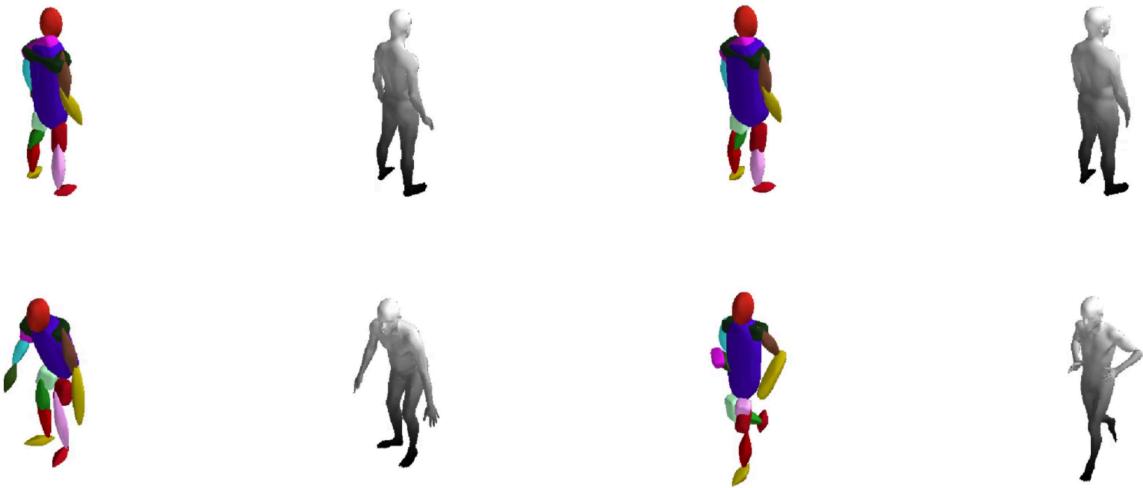
- 使用了期望 $\mathbb{E}_{p(z)}$ 和条件 $z_m = 1$ (即形状基元 m 存在时的损失)

3. 形状基元加权后的总损失 \$\$

$$\mathcal{L}(P, X) = \sum_m \gamma_m \mathcal{L}_{P \rightarrow X}^m(P, X)$$

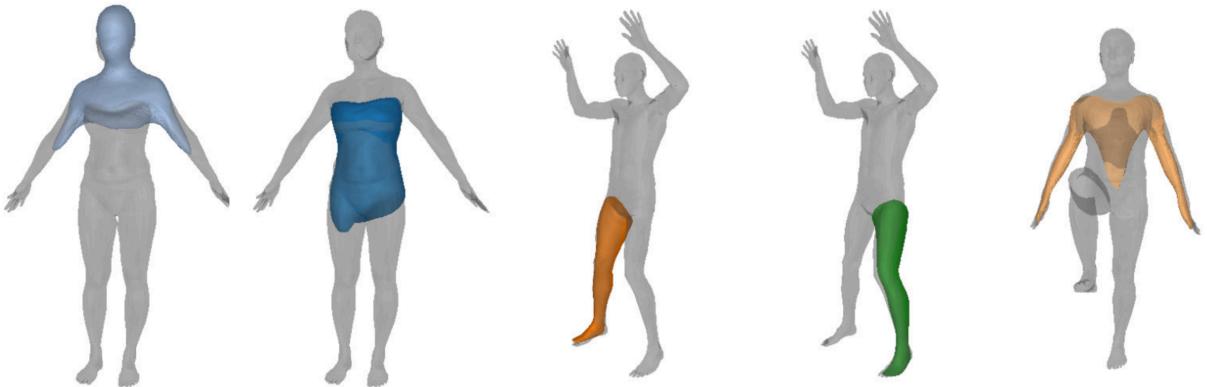
γ_m : 形状基元的存在概率，衡量基元 m 的重要性 - M : 场景中预测的

- 结果展示 <https://autonomousvision.github.io/superquadrics-revisited/>



- 可以看出，尽管转换后的基元表示模型并没有那么精细，但是模型的
确按照我们的意愿工作，并且这是采用无监督方法进行工作的
- 而且可以看出一些规律，例如人体的驱赶总是紫色的块，左小腿总是
绿色的块——这证明我们生成的模型确实是携带有意义的语义信息的

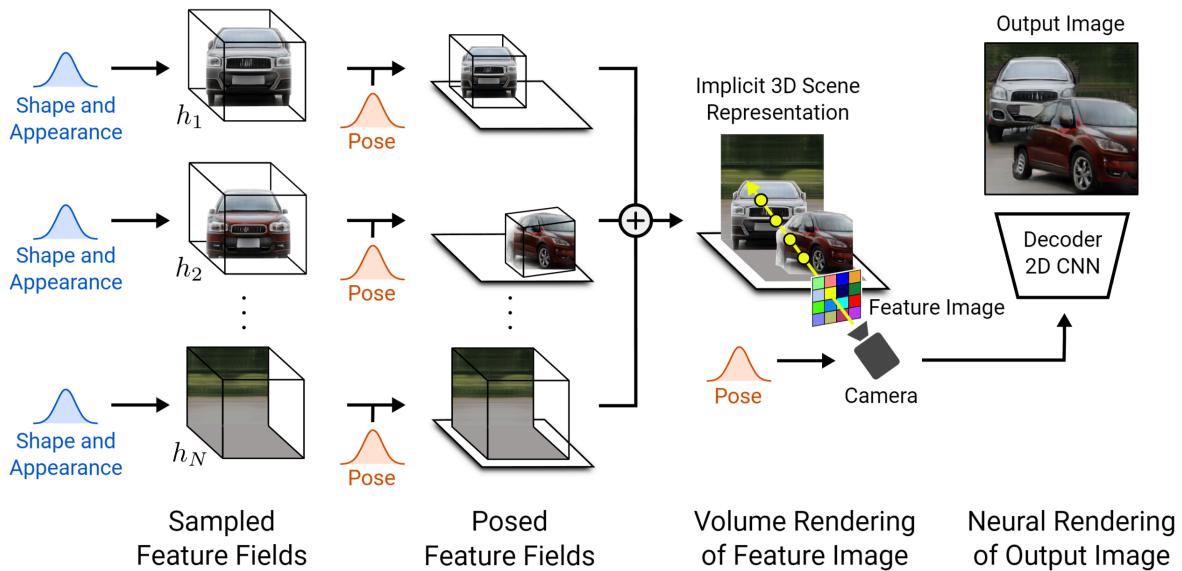
- Neural Parts [Paschalidou, Katharopoulos, Geiger and Fidler: Neural Parts:
Learning Expressive 3D Shape Abstractions with Invertible Neural
Networks. CVPR, 2021](#)



- 该领域的后续工作，称为 Neural Parts——一种基于神经网络的形状
分割方法，将复杂的三维物体分解为语义上有意义的部分（例如手
臂、躯干等）
- 使用 **invertible neural networks**（可逆神经网络）来学习三维形状的
部分——网络可以在输入和输出之间进行双向映射，使得网络能够有
效地学习部分形状的紧凑表示，同时保持生成的形状与输入三维数据
之间的高保真度
- 了解更多关于 Neural Parts 方法的细节和代码实现：
<https://autonomousvision.github.io/neural-parts/>

Compositional Feature Fields

- GIRAFFE: Compositional Generative Neural Feature Fields



- 这是我们之前讲过的 GIRAFFE，获得 2021 年 CVPR 最佳论文奖的文章
- 它将图像的场景和对象分开，试图用多个 NeRF 的组合来表示一个 3D 场景
- 这样做有一个好处——我们可以独立的对 3D 场景中的各个对象（包括背景）进行变换操作，而这一点对于利用整体表示 3D 场景（例如 GAN）来说是无法做到的
- 具体可以查看 blog <https://autonomousvision.github.io/giraffe/>，其中有针对场景中单个对象变换的动态示意图

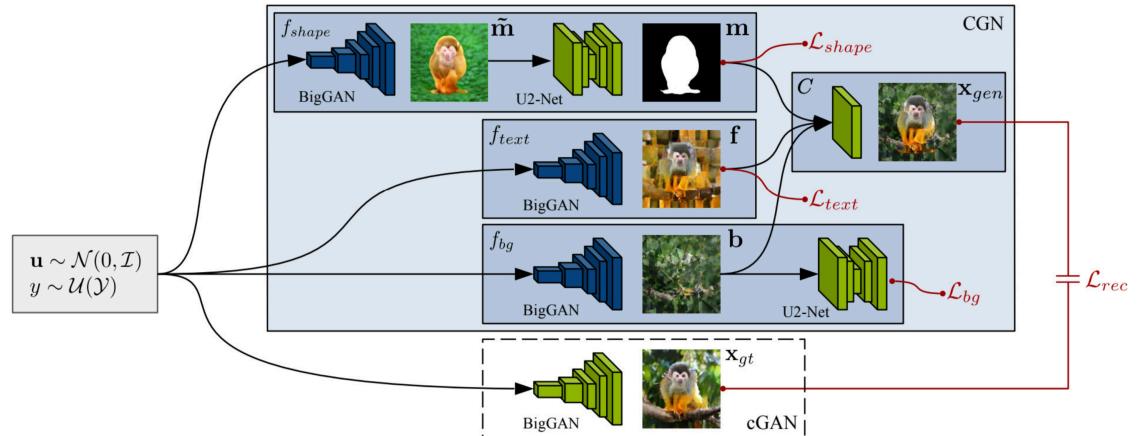
Causal Reasoning

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



- 如果你观察上图，你很容易就能看出来，骆驼、牛、骆驼、牛
- 但是对于神经网络而言，由于它们具有走“捷径”的特性，因此它们往往学习一些错误的相关性——比如牛一定是在草地上出现，骆驼一定是在沙漠里出现；那么当一个和骆驼很相似的动物出现在沙漠上，它可能就会错误地认为是骆驼

- 我们是否可以通过学习，将图像生成分解为一种“因果机制”，从而正确的学习到数据中真实的因果关系？
 - 这将会让模型考虑“这张图在不同背景下看起来会是什么样？”，也即，模型需要能够回答 **counterfactual questions** (反事实问题)
- 因此，我们用反事实的数据来训练模型，提高其面对 OOD (Out of Distribution) 数据时的鲁棒性
- 网络架构



- CGN 将图像分解为 3 部分因果机制：形状、纹理、背景，分别由 3 个对应的编码器提取
- 通过输入随机噪声 u 和类别标签 y ，CGN 利用生成器生成形状的遮罩 m ，物体的纹理信息 f ，背景 b ，然后将其组合为新的图像

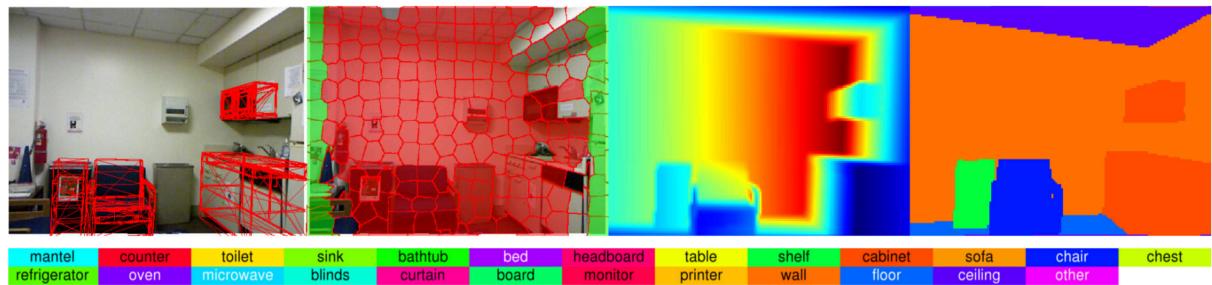
Shape	red wine	cottontail rabbit	pirate ship	triumphal arch	mushroom	hyena dog
Texture	carbonara	head cabbage	banana	Indian elephant	barrel	school bus
Background	baseball	valley	bittern (bird)	viaduct	grey whale	snorkel

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

- ■ 其效果如图所示，我们可以看到不同形状、纹理、背景的组合生成出的 OOD 数据
 - 这篇内容的 blog：<https://autonomousvision.github.io/cgn/>

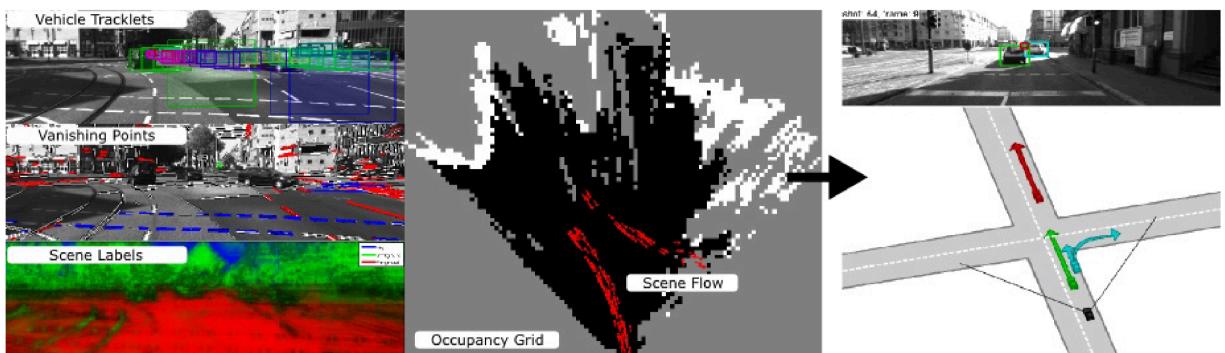
Holistic 3D Scene Understanding

- Indoor Scene Understanding [Geiger and Wang: Joint 3D Object and Layout Inference from a Single RGB-D Image. GCPR, 2015.](#)



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

- 同样的，能够理解整个全景 3D 场景也是一件非常棘手的任务
- 这篇工作可以从单张 RGB-D (包含色彩 RGB 信息和深度 D 信息) 中推断出室内场景的 3D 物体和场景布局
- 该项目网址：https://www.cvlabs.net/projects/indoor_scenes/
- Urban Scene Understanding Geiger, Lauer, Wojek, Stiller and Urtasun: 3D Traffic Scene Understanding From Movable Platforms. PAMI, 2014



<http://www.cvlabs.net/projects/intersection/>

- 也有关于室外场景（主要指城市街景）的场景理解，从移动平台，如车辆的传感器数据中推断城市交通的 3D 布局、物体位置、交通活动
- 该项目网址：<https://www.cvlabs.net/projects/intersection/>

12.3 Human Body Models

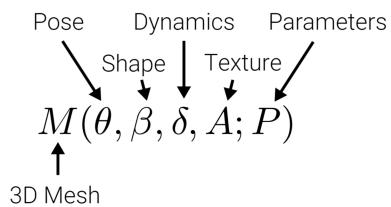
- Humans Interact through their Bodies



- 人类的互动几乎都是通过肢体动作完成的——拥抱、合作、搬运、舞蹈、握手等
- Generative Human Body Model

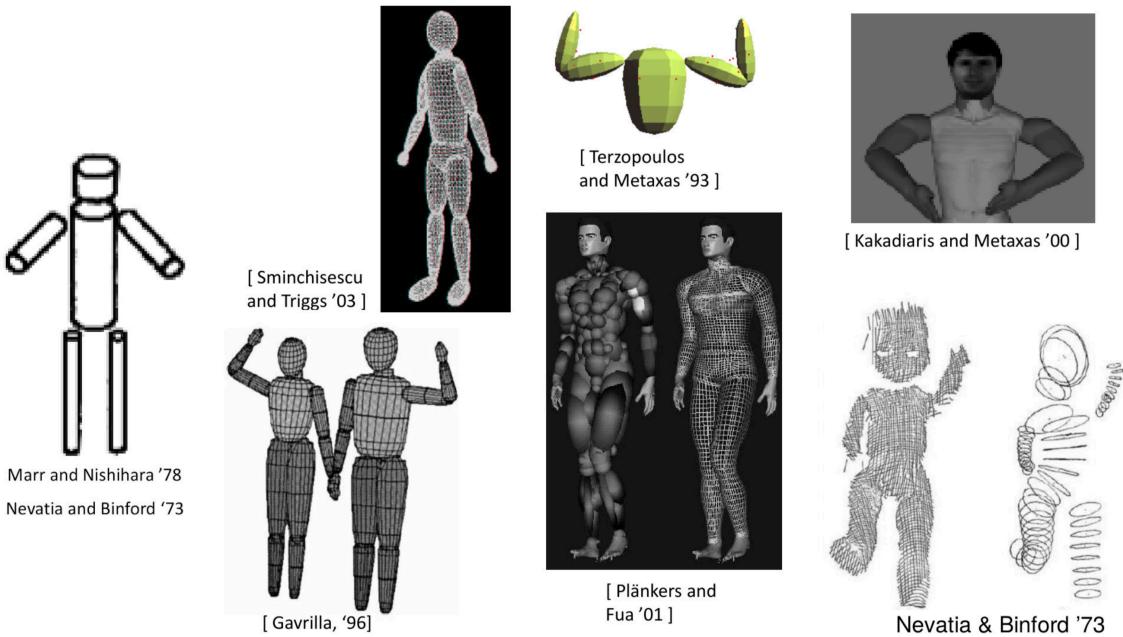


- 我们首先只考虑最简单的人体模型——穿最少衣服的人体如何表示
- 模型设计目标

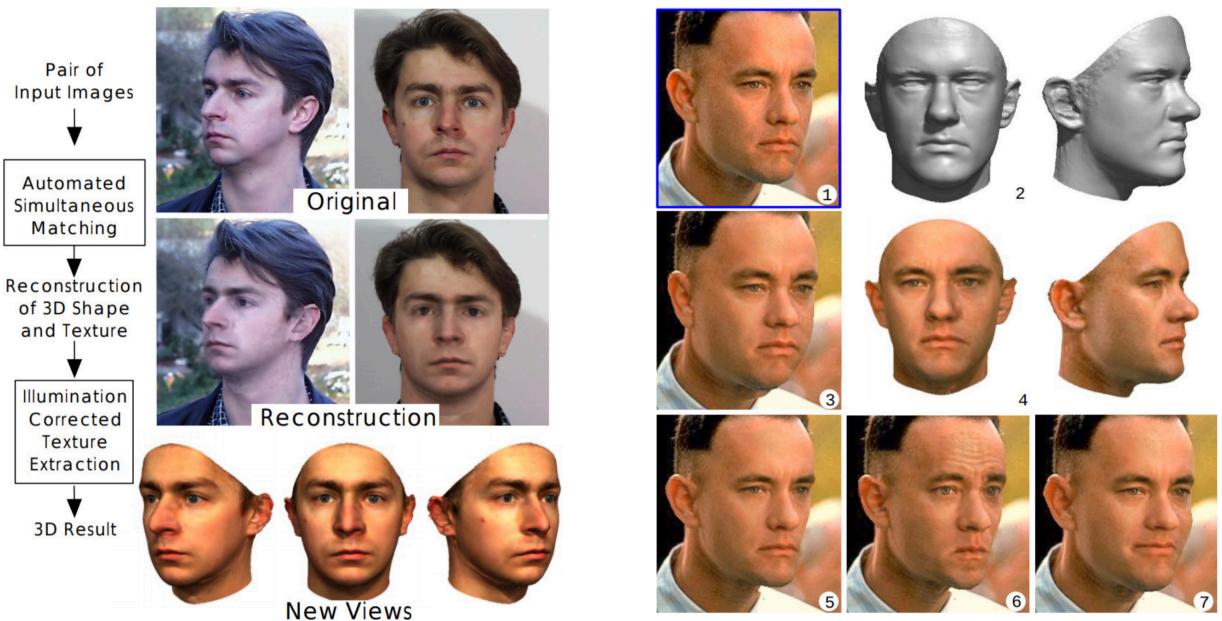


- 看起来像是真人
- 可以像真人一样活动
- 尽可能少的参数
- 能够方便地拟合真实数据
- 能够方便地转换为动画表示

- Early Body Models



- 对于人体模型的研究在很早开始就出现了——我们不考虑人体的内部结构，只考虑能够肉眼观察到的部分，并且只考虑“皮肤级别”，也即穿着最少衣服的简单情况
- Inspiration: Morphable 3D Face Model [Blanz and Vetter: A Morphable Model for the Synthesis of 3D Faces. SIGGRAPH, 1999](#)



- 该领域最早的启发是来自 Blanz 和 Vetter 的工作——可变形的 3D 面部模型
- 对于 1999 年的工作来说，它真的很令人印象深刻，还原效果非常精细
- 这项工作的具体内容可以查看这个古老的视频：
<https://www.youtube.com/watch?v=pSRA8GpWIrA>

SMPL: A Skinned Multi-Person Linear Model

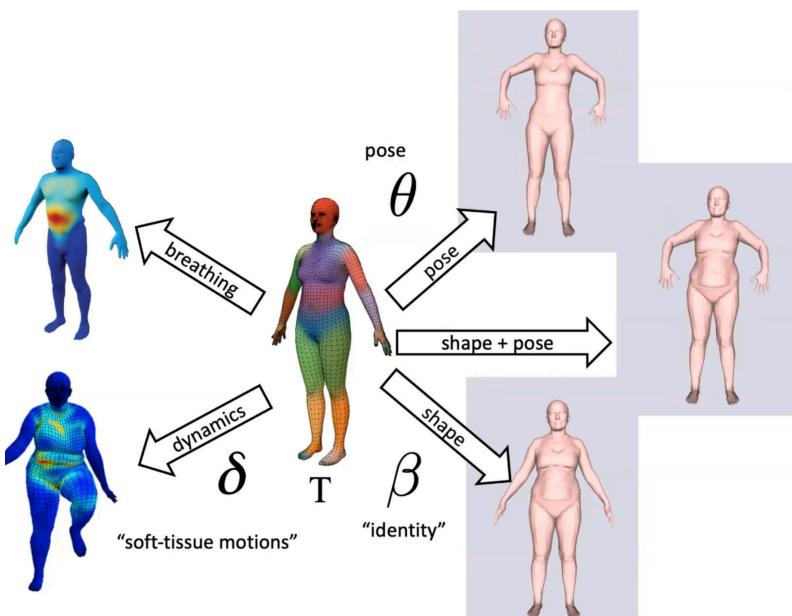
下面我们来考虑这种最简单的人体模型如何实现，称之为 SMPL [Loper, Mahmood, Romero, Pons-Moll and Black: SMPL: A skinned multi-person linear model. SIGGRAPH, 2015](#)

- Input: Raw 3D Scans



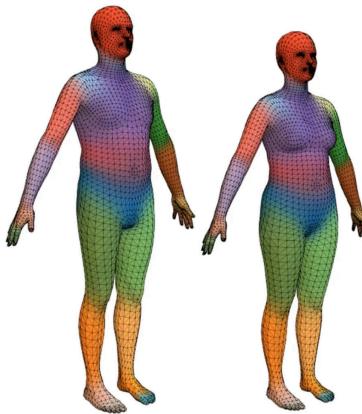
- 输入人体的 3D 扫描数据

- Output: Factored Model

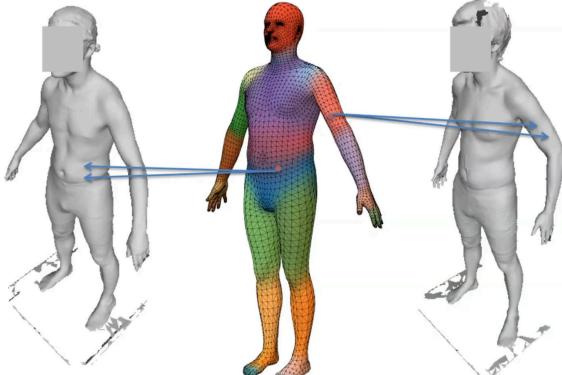


- 输出因子分解后的模型，也即我们可以分别独立的控制模型的姿势、形状、动态、呼吸等我们感兴趣的所有因素

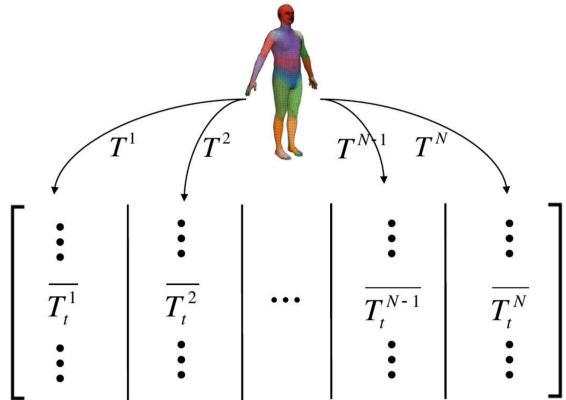
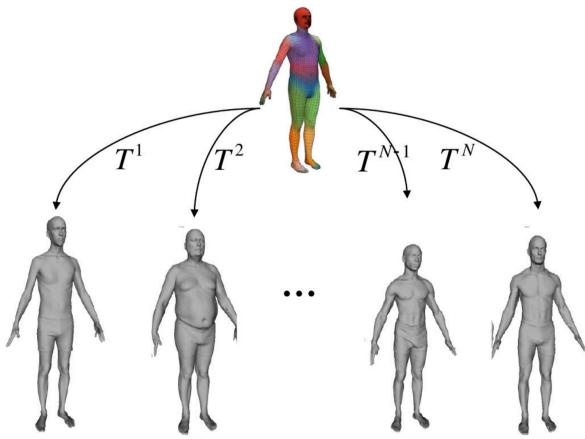
- Representation: 3D Mesh



- SMPL 使用 3D 网格（顶点和面）作为表示
- 上图是由艺术家设计的男性和女性的基础网格，它可以适配绝大多数的人体作为基础模型
- 该网格由 7000 个顶点——可以用 21000 个数字来表示
- 可以看到这些数字的大多数设置只是一些合理随机初始化的配置，并不适用于任何一个具体的人或者形状；因此，我们在这个基础上考虑如何修改这些数字的配置来让它表示某个具体的人
- Data Preprocessing: Mesh Registration

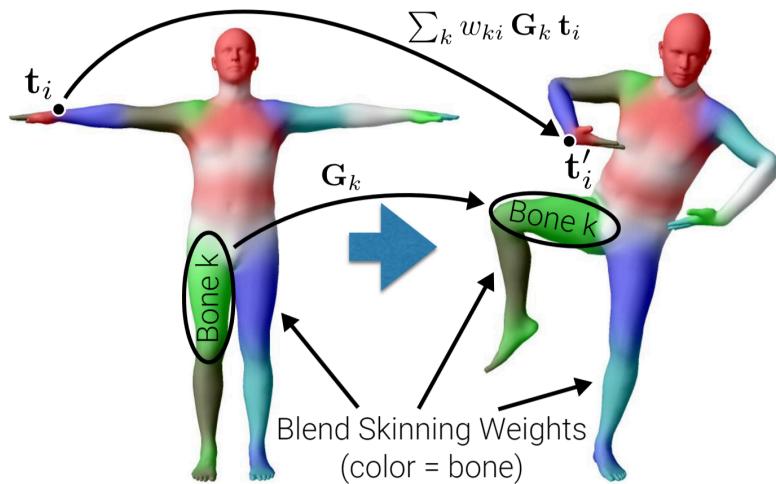


- 首先我们要做的工作称为网格注册——我们需要将扫描得到的无序、不完整的 3D 点云对齐到一个模板网络上，例如图中灰色的部分为扫描结果，它存在着孔洞、伪影等；因此我们需要考虑如何将其对齐到标准网络上，形成一个完整的、连续的模型
- 但是这实际上是一个 chicken-and-egg 问题（鸡生蛋问题），因为要完成配准需要一个现有的身体模型，但是身体模型的构建本身又依赖于配准的结果
- 解决方案：联合求解模型和配准问题
- Shape Deformation Subspace



- 我们希望构建一个较低维度的形状表示，仅用来描述某个特定的体型和标准体型的差别，从而表示这个特定体型
- 我们将人体模型的顶点坐标表示为一个高维向量 \mathbf{T} (例如 21000 维度)
- 然后用这个向量减去均值模型的向量 (艺术家设计的基础模型)
- 然后我们将所有的网格数据堆叠为一个矩阵，每一列代表一个网格样本
- 对数据进行 **Principal component analysis (PCA)** (主成分分析) 处理，得到一个线性模型 \$\$
\mathbf{T} = \mathbf{S}\beta + \mu
\$\$
- \$S\$：主成分基矩阵 - \$\beta\$：低维的表示系数 (通常为 10 – 300 维度) - \$\mu\$：

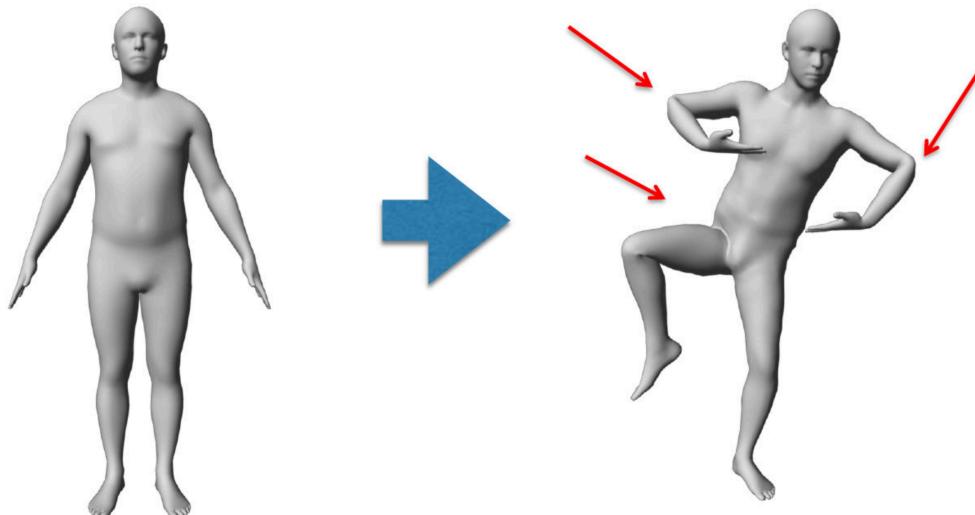
- **Linear Blend Skinning**



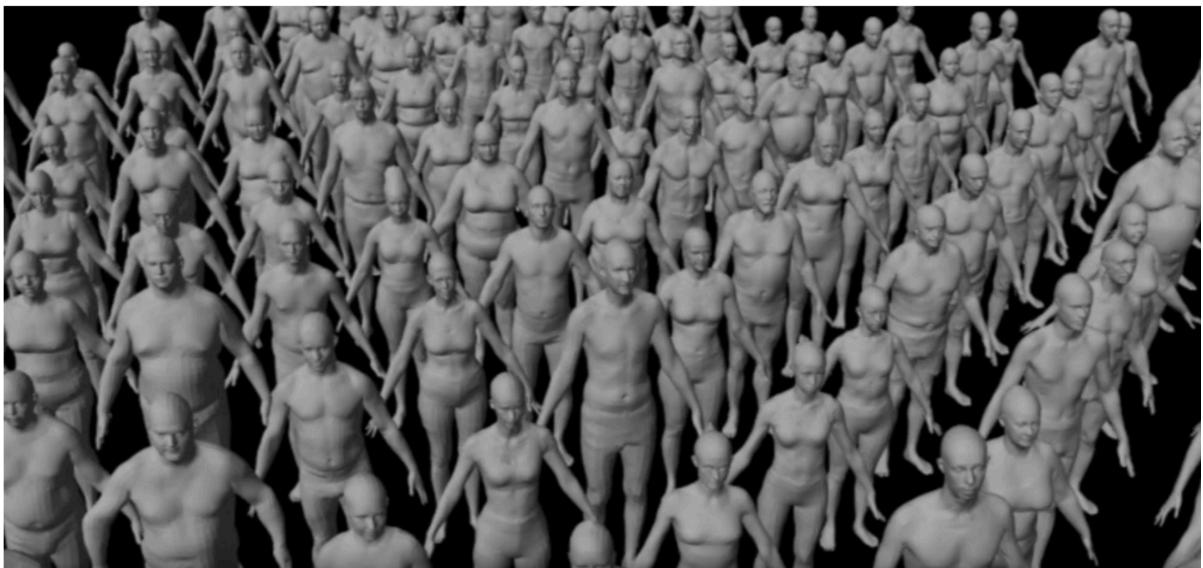
- 这是一种另外的人体表达方式——它尝试将皮肤的变化用多个骨骼的变
化的线性组合来表示，称为线性混合蒙皮 (LBS)

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

- \mathbf{t}_i : 顶点 i 的初始位置 (休息状态下)
- \mathbf{t}'_i : 顶点 i 在变形后的位置
- w_{ki} : 骨骼 k 对顶点 i 的影响权重 (Blend skinning weight)
- θ : 姿态参数
- \mathbf{J} : 骨骼连接处 (关节)
- $\mathbf{G}_k(\theta, \mathbf{J})$: 骨骼 k 的刚体变换，依赖于姿态参数 θ 和关节位置 \mathbf{J}
- 其中，权重 w_{ki} 通常由艺术家手动创建，或者由某些算法计算
- 骨骼变换 \mathbf{G}_k 是刚性变换，包括旋转和平移
- 这种模型简单高效，适合实时应用 (例如游戏，虚拟现实)，可控制性
强
- 问题



- 尽管 LBS 的计算非常简单，但它在处理一些复杂的非刚性变形时，可能会出现一些不自然的现象——例如皮肤穿插，不自然的体积损失，关节区域的扭曲等
- LBS vs SMPL
 - 传统的 LBS 公式 \$\$
 \mathbf{t}_i' = \sum_k w_k \mathbf{G}_k(\boldsymbol{\theta}, \textcolor{red}{\mathbf{J}(\boldsymbol{\beta})})
 \$\$
 \$\$
 \mathbf{t}_i' = \sum_k w_k \mathbf{G}_k(\boldsymbol{\theta}, \mathbf{J}(\boldsymbol{\beta})) = \sum_k w_k \mathbf{G}_k(\boldsymbol{\theta}, \mathbf{J}(\boldsymbol{\beta})) + \sum_k w_k \mathbf{G}_k(\boldsymbol{\theta}, \mathbf{J}(\boldsymbol{\beta}))
 \$\$
 - SMPL 在此基础上进行了改进
- SMPL 从非常庞大的数据集中学习、优化参数（大约 1000 以上的人体扫描数据）





- 实现了相当惊人的效果
 - 另外，人们也可以利用 SMPL 来重定向角色——例如将一个角色的动作变化完全转移到另一个不同外观的人体上



- Dynamic SMPL (DMPL)