人脸3D模型的发展

罗德里格斯公式推导

基于图像的人脸三维重建方法

1. 立体匹配（Structure From Motion,SfM)
2. Shape from Shading,sfs
3. ***三维可变形人脸模型（3DMM)***

什么是3DMM模型

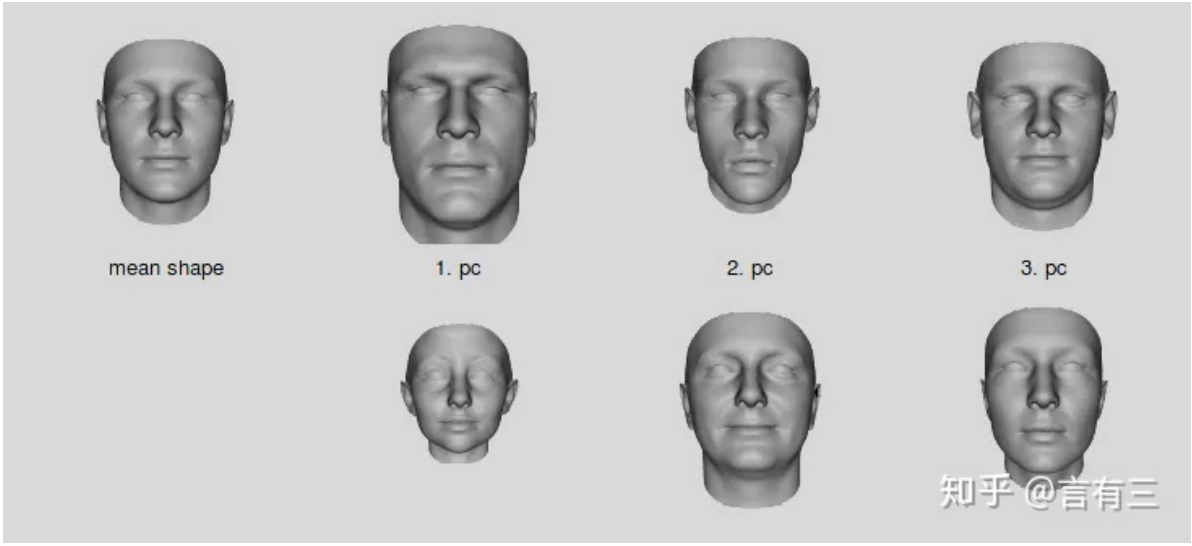
3DMM,即三维可变形人脸模型，它可以使用固定数量的参数来表示一个三维人脸。

核心思想：一个三维人脸可以可以由其他许多幅人脸正交基加权线性相加而来。

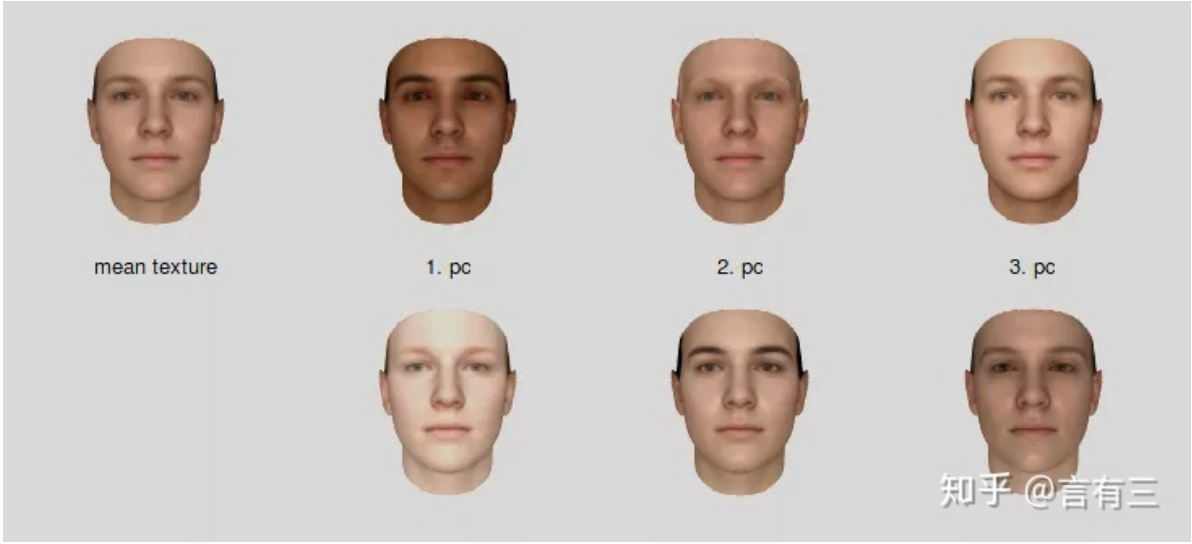
类比我们所处三维空间中的点，每一点(x,y,z),实际上都由三维空间三个方向的基量，（1,0,0），（0,1,0），（0,0,1）加权相加所得，权重分别是x,y,z.

那么对于人脸来说，一个人脸可以由其他多幅人脸加权相加得到。在BFM模型中，将人脸的表示，分为形状向量和纹理向量，即一个人脸分为形状和纹理两部分叠加。如图所示：

人脸的形状可以表示为一个向量Shape Vector: $S = (x_1, y_1, z_1, x_2, y_2, z_2, \ldots, x_n, y_n, z_n)$ 即人脸表面点的三维坐标。



纹理向量Texture Vector:$T = (r_1, g_1, b_1, r_2, g_2, b_2, \ldots, r_n, g_n, b_n)$,即每个点的的颜色信息。



任意的人脸模型可以由数据集中的m个人脸模型进行加权组合如下：

$$\mathbf{S}_{\text{mod}} = \sum_{i=1}^{m} a_i \mathbf{S}_i, \quad \mathbf{T}_{\text{mod}} = \sum_{i=1}^{m} b_i \mathbf{T}_i, \quad \sum_{i=1}^{m} a_i = \sum_{i=1}^{m} b_i = 1$$

其中S_i 和 T_i 就是数据库中的第i张人脸的形状向量和纹理向量。

但是在实际构建模型的时候不能使用S_i 和 T_i 作为基向量，因为它们之间不是正交的。

使用PCA进行降维分解，求正交基

1. 首先计算形状和纹理向量的平均值。
2. 中心化人脸数据。
3. 分别计算协方差矩阵
4. 求得形状和纹理协方差矩阵的特征值λ_1，λ_2和特征向量si，ti。

转化后的模型为：

$$S_{model} = \bar{S} + \sum_{i=1}^{m-1} \lambda_{1i} s_i, T_{model} = \bar{T} + \sum_{i=1}^{m-1} \lambda_{2i} t_i$$

BFM模型

Model

$$\mathbf{s} = (x_1, y_1, z_1, \ldots x_m, y_m, z_m)^T$$
$$\mathbf{t} = (r_1, g_1, b_1, \ldots r_m, g_m, b_m)^T$$

一个人脸使用两个向量表示，顶点坐标$(x_j, y_j, z_j)^T \in \mathbb{R}^3$,顶点颜色 $(r_j, g_j, b_j)^T \in [0,1]^3$ 。m=53490个顶点。

BFM假定形状和纹理是相互独立的。

使用数据集构建一个高斯模型：

$$\mathcal{M}_s = (\boldsymbol{\mu}_s, \boldsymbol{\sigma}_s, \mathbf{U}_s) \text{ and } \mathcal{M}_t = (\boldsymbol{\mu}_t, \boldsymbol{\sigma}_t, \mathbf{U}_t)$$

其中均值：$\boldsymbol{\mu}_{\{s,t\}} \in \mathbb{R}^{3m}$,

标准差：$\boldsymbol{\sigma}_{\{s,t\}} \in \mathbb{R}^{n-1}$

正交基：$\mathbf{U}_{\{s,t\}} = [\mathbf{u}_1, \ldots \mathbf{u}_n] \in \mathbb{R}^{3m \times n-1}$

值得注意的是正交基的长度为1，所以乘 $diag(\sigma_s)$ 相当于将U还原到协方差的量级。

那么最后的模型为：

$$\mathbf{s}(\alpha) = \boldsymbol{\mu}_s + \mathbf{U}_s \operatorname{diag}(\boldsymbol{\sigma}_s)\alpha$$
$$\mathbf{t}(\beta) = \boldsymbol{\mu}_t + \mathbf{U}_t \operatorname{diag}(\boldsymbol{\sigma}_t)\beta$$

α, β 是需要学习的系数向量。

FLAME 模型

FLAME构造了更加精确的和富于表情的头部和人脸模型，并且引入了头部姿势和眼球旋转。

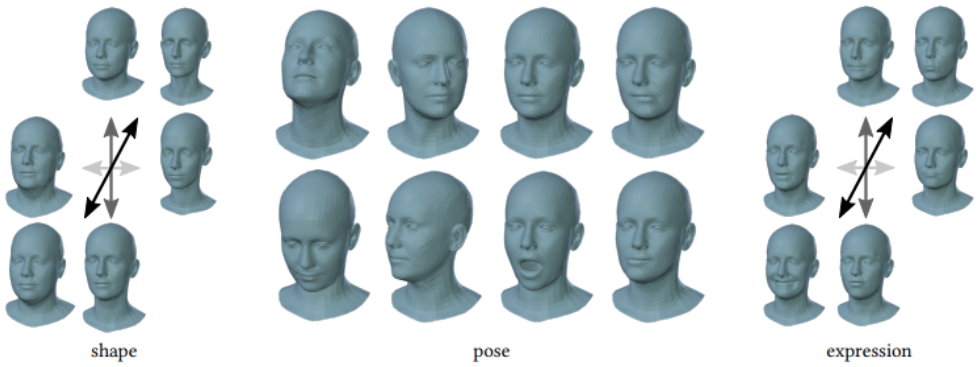


Fig. 2. Parametrization of our model (female model shown). Left: Activation of the first three shape components between −3 and +3 standard deviations. Middle: Pose parameters actuating four of the six neck and jaw joints in a rotational manner. Right: Activation of the first three expression components between −3 and +3 standard deviations.

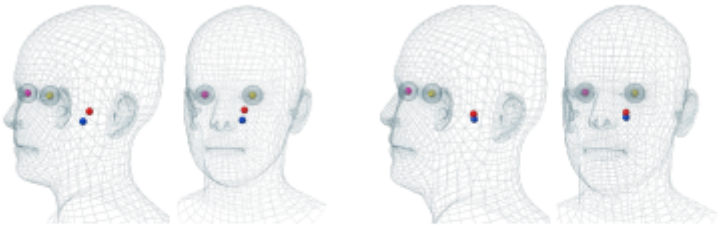


Fig. 3. Joint locations of the female (left) and male (right) FLAME models. Pink/yellow represent right/left eyes. Red is the neck joint and blue the jaw.

模型细节

FLAME模型和SMPL模型一样，使用LBS。该模型设置了5023个顶点，4个关键点（脖子，下巴，和眼球），

$$M(\vec{\beta}, \vec{\theta}, \vec{\psi}) : \mathbb{R}|\vec{\beta}| \times |\vec{\theta}| \times |\vec{\psi}| \rightarrow \mathbb{R}^{3N} \vec{\beta} \in \mathbb{R}^{|\vec{\beta}|}, \text{ pose } \vec{\theta} \in \mathbb{R}^{|\vec{\theta}|}, \vec{\psi} \in \mathbb{R}^{|\vec{\psi}|}$$

模型输入三种参数，就能得到5023个顶点坐标。

和SMPL模型一样，FLAME的组成部分有： template mesh, shape blend shape,pose blend shape,expression blend shape.

所以最终的模型是：

$$M(\vec{\beta}, \vec{\theta}, \vec{\psi}) = W\left(T_P(\vec{\beta}, \vec{\theta}, \vec{\psi}), \mathbf{J}(\vec{\beta}), \vec{\theta}, \mathcal{W}\right)$$

where

$$T_P(\vec{\beta}, \vec{\theta}, \vec{\psi}) = \overline{\mathbf{T}} + B_S(\vec{\beta}; \mathcal{S}) + B_P(\vec{\theta}; \mathcal{P}) + B_E(\vec{\psi}; \mathcal{E})$$

这个模型怎么理解呢？

通过形状，姿态和表情参数以及一个均值模板可以得到一个特定人脸的静态3D模型，这个时候的人脸处在一个标准的姿态下。要想使头部的姿态发生变化，通过形状参数可以获得 0 pose,即一个标准姿态下人脸的关节点位置，然后再通过姿态参数θ 可以获取当前姿态下关节点的位置，然后再通过LBS,即当前的关节点位置×W,获得顶点的坐标。

(1)shape blendshapes

$$B_S(\vec{\beta};\mathcal{S})=\sum_{n=1}^{|\vec{\beta}|}\beta_n\mathbf{S}_n$$
$$\text{where } \vec{\beta}=\left[\beta_1,\cdots,\beta_{|\vec{\beta}|}\right]^T \text{ denotes the shape coefficients, and}$$
$$\mathcal{S}=\left[\mathbf{S}_1,\cdots,\mathbf{S}_{|\vec{\beta}|}\right]\in\mathbb{R}^{3N\times|\vec{\beta}|} \text{ denotes the orthonormal shape basis,}$$

(2)pose blendshapes

$$B_P(\vec{\theta};\mathcal{P})=\sum_{n=1}^{9K}\left(R_n(\vec{\theta})-R_n\left(\vec{\theta}^*\right)\right)\mathbf{P}_n$$

其中，$R_n(\vec{\theta})$,表示将轴角向量转化为旋转矩阵。

$$\mathbf{P}_n\in\mathbb{R}^{3N} \text{ describes the vertex offsets from}$$
$$\text{the rest pose activated by } R_n\text{, and the pose space } \mathcal{P}=[\mathbf{P}_1,\cdots,\mathbf{P}_{9K}]\in\mathbb{R}^{3N\times 9K}$$

包含所有的pose blend shapes.

$\mathcal{P}$可以看做一种形式的权重。

这里的$\mathcal{P}$ 是直接定义损失函数训练出来的。

(3)expression blendshapes

$$B_E(\vec{\psi};\mathcal{E})=\sum_{n=1}^{|\vec{\psi}|}\vec{\psi}_n\mathbf{E}_n$$
$$\text{where } \vec{\psi}=\left[\psi_1,\cdots,\psi_{|\vec{\psi}|^T}^T\right. \text{ denotes the expression coefficients, and}$$
$$\mathcal{E}=\left[\mathbf{E}_1,\cdots,\mathbf{E}_{|\vec{\psi}|}\right]\in\mathbb{R}^{3N\times|\vec{\psi}|} \text{ denotes the orthonormal expression}$$

(4)Template shape:

从3D扫描数据集得到的平均模型。

2D->3D 驱动参数的学习

DECA 模型

加亿点点细节～

DECA主要关注于如何从2D图像恢复出逼真的3D人脸，所以它的主要内容是从2D图像中恢复出3DMM模型需要的参数及其他的一些细节内容。DECA不同于之前工作的主要内容是对皱纹如何跟随表情变化进行了建模，所以说是加入了一些细节，使生成的3D图形更加逼真。

前置知识：

(1)Geometry prior:

本文用到的3D人头模型是FLAME，FLAME是一个统计学的模型，该模型输入三种参数：$\boldsymbol{\beta}\in\mathbb{R}^{|\beta|}$ 表示shape参数或者叫identity参数，$\boldsymbol{\theta}\in\mathbb{R}^{3k+3}$ 表示关节点参数，FLAME中有四个关节点两眼，下巴和脖子。$\boldsymbol{\psi}\in\mathbb{R}^{|\psi|}$ 表情参数。输出n=5023个vertices.模型可以表示为：

$M(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\psi}) = W\left(T_P(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\psi}), \mathbf{J}(\boldsymbol{\beta}), \boldsymbol{\theta}, \boldsymbol{W}\right)$

W()是blend skining function,就是通过joint的位置和相应的权重W对顶点位置做一些变换。

其中：

$T_P(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\psi}) = \mathbf{T} + B_S(\boldsymbol{\beta}; \mathcal{S}) + B_P(\boldsymbol{\theta}; \boldsymbol{P}) + B_E(\boldsymbol{\psi}; \mathcal{E})$

人头当前的形状，由人头模板加上三种blend shape组成，包括shape blend shape,pose blend shape,expression blend shape.

(2)Apperance model:表观模型，即皮肤的纹理颜色这些

本文用的是FLAME模型，但是FLAME模型没有表观模型，所以作者将BFM模型的albedo subspace转换到FLAME的uv layout.这个模型输入是$\boldsymbol{\alpha} \in \mathbb{R}^{|\alpha|}$,输出是UV alebedo map$A(\boldsymbol{\alpha}) \in \mathbb{R}^{d \times d \times 3}$.

（3）camera model

本文作者使用了一个正交相机模型，将3D mesh投影到了2d图像空间，映射关系为：

$\mathrm{v} = s\Pi\left(M_i\right) + \mathrm{t}$

其中M_i是3d顶点，Π 是3d to 2d 的映射矩阵，s sacle,t是平移。

（4）Illumination model：

人脸领域最常用的光照模型是SH模型，该模型假设光源比较远，表面反射是Lambertian,即理想散射，那shaded image的计算公式是：

$B(\boldsymbol{\alpha}, \mathbf{l}, N_{uv})_{i,j} = A(\boldsymbol{\alpha})_{i,j} \odot \sum_{k=1}^{9} \mathbf{l}_k H_k\left(N_{i,j}\right)$

A : albedo N:surface normal B:shaded texture

H_k 表示SHbasis,l_k表示系数。

（5）texture rendering

Given the geometry parameters ($\boldsymbol{\beta}$, $\boldsymbol{\theta}$, $\boldsymbol{\psi}$), albedo ($\boldsymbol{\alpha}$), lighting (l) and camera information $\boldsymbol{c}$, we can generate the 2D image Ir by rendering as Ir = R (M, B, c), where R denotes the rendering function

方法

关键思想：

人脸会随着不同的表情变化，表现出不同的细节，但是他的一些固有的形状是不会变化的。

并且，人脸的细节信息应该被分成两种，一种是静态不变的个人细节，（比如痣，胡子，睫毛）和基于表情的细节（比如皱纹）。为了保持在表情变化引起的动态细节同时时保持静态细节，DECA学习了一个expression-conditional 细节模型，该模型能够产生出独立于表情的细节displacement map.个人理解将表情参数和人脸特征一同送入细节decoder模型，可以学习到一些不随表情变化的细节特征。

还有一个问题是，训练数据的获取比较困难，所以提出了一种直接从wild image学习几何细节的方法。

1.coarse recontruction

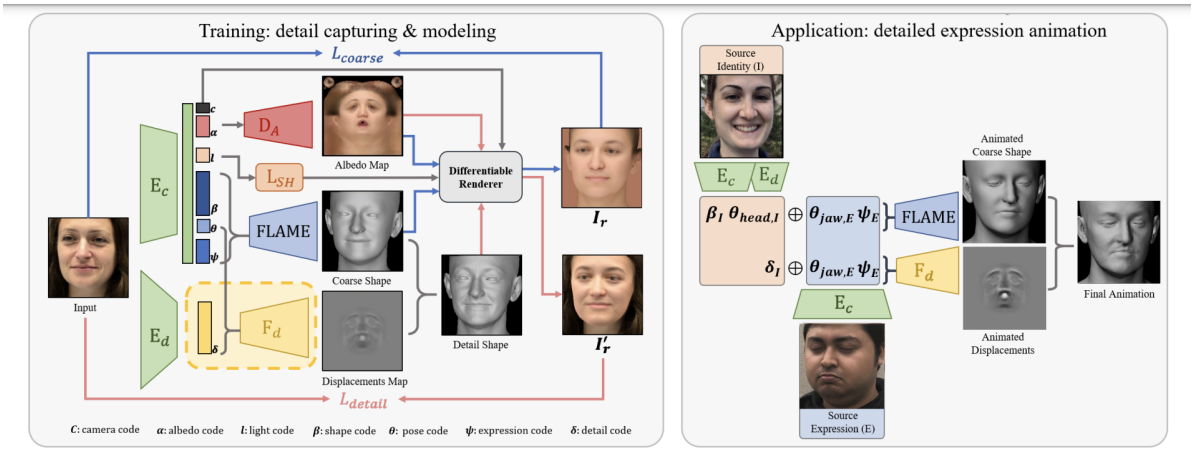


Fig. 2. DECA training and animation. During training (left box), DECA estimates parameters to reconstruct face shape for each image with the aid of the shape consistency information (following the blue arrows) and, then, learns an expression-conditioned displacement model by leveraging detail consistency information (following the red arrows) from multiple images of the same individual (see Sec. 4.3 for details). While the analysis-by-synthesis pipeline is, by now, standard, the yellow box region contains our key novelty. This displacement consistency loss is further illustrated in Fig. 3. Once trained, DECA animates a face (right box) by combining the reconstructed source identity's shape, head pose, and detail code, with the reconstructed source expression's jaw pose and expression parameters to obtain an animated coarse shape and an animated displacement map. Finally, DECA outputs an animated detail shape. Images are taken from NoW [Sanyal et al. 2019]. Note that NoW images are not used for training DECA, but are just selected for illustration purposes.

粗糙重建指的是只学习FLAME模型的输入参数。如图所示，使用一个Encoder 模型直接回归出一些参数，(比如FLAME模型需要的参数（β, θ, ψ）,反射率系数α ,相机参数c,光照参数l). 模型采用resnet50 模型，一共输出236维的latent code。并从重建的3d模型 投影出一张2d图片I_r和原来的图片进行对比，求一个损失。损失函数为：

$$L_{\text{coarse}} = L_{lmk} + L_{eye} + L_{pho} + L_{id} + L_{sc} + L_{reg}$$

关键点损失：2d ground truth和3d 重投影的损失：

$$L_{lmk} = \sum_{i=1}^{68} \left\| \mathbf{k}_i - s\Pi\left(M_i\right) + \mathrm{t} \right\|_1$$

闭眼损失：

$$L_{eye} = \sum_{(i,j)\in E} \left\| \mathbf{k}_i - \mathbf{k}_j - s\Pi\left(M_i - M_j\right) \right\|_1$$

上眼皮关键点和下眼皮关键点距离的损失，这个损失可以减少3d和2d关键点没有对齐的影响。

图像本身的loss:

$$L_{pho} = \left\| V_I \odot (I - I_r) \right\|_{1,1}$$

其中V_I 表示脸部区域的mask ,通过脸部分割模型获得。

身份损失：

就是用一个特征提取网络，提取ground truth 图片和重投影图片的人脸特征，然后求一个余弦相似度。

$$L_{id} = 1 - \frac{f(I)f(I_r)}{\|f(I)\|_2 \cdot \|f(I_r)\|_2}$$

形状一致性损失：

给出一个人的两张不同照片Encoder E_c 应该输出同样的参数，因为一个人的shape是不变的，变的是细节。

$$L_{sc} = L_{\text{coarse}}\left(I_i, \mathcal{R}\left(M\left(\boldsymbol{\beta}_j, \boldsymbol{\theta}_i, \boldsymbol{\psi}_i\right), B\left(\boldsymbol{\alpha}_i, \mathbf{l}_i, N_{uv,i}\right), \mathbf{c}_i\right)\right)$$

正则化项：

对需要学习的β, ψ, α 进行L_2正则化。

2.细节重建

细节重建，使用一张细节UV偏移map,去增强FLAME的几何细节。和coarse重建一样，使用一个同样结构的Encoder,E_d ,将输入图像编码到128维的latent code δ .然后再将这个latent code和FLAME的表情参数ψ 和pose参数θ .拼接起来，通过F_d 解码成D（UV displacement map). 为了渲染，D被转换为一个normal map.

细节渲染：

为了得到具有细节的M‘，我们将M和他的normal map，转化到UV 空间，

$$M'_{uv} = M_{uv} + D \odot N_{uv}$$

其中D是detail code,N_{uv} 代表normal map，M_{uv} 应该是coarse model的UV map.

从M’ 计算得到N‘。然后就可以调用渲染函数进行渲染。B表示的是texture.

$$I'_r = \mathcal{R}\left(M, B\left(\boldsymbol{\alpha}, \mathbf{1}, N'\right), \mathbf{c}\right)$$

从而可以得到渲染后的图片I'_r .

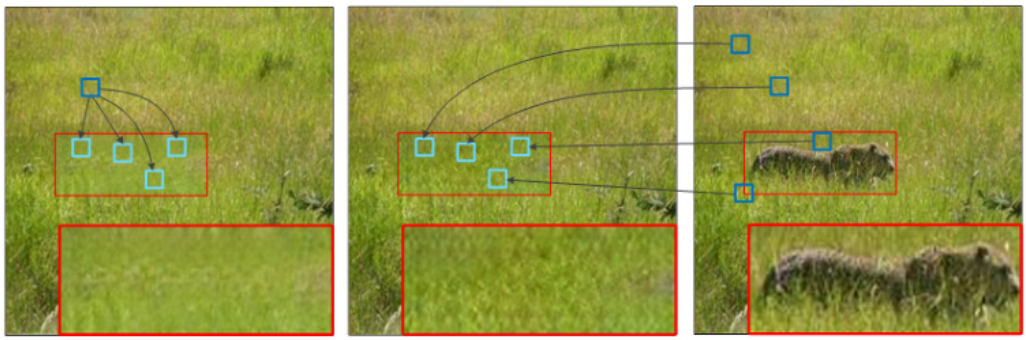
$$L_{\text{detail}} = L_{phoD} + L_{mrf} + L_{sym} + L_{dc} + L_{regD} .$$

ID-MRF loss:

ID-MRF

隐式多元马尔科夫随机场损失.用来惩罚生成图像中的每个patch只和target中大部分的patch比较相似的情况，所以能够恢复出细节。

要计算ID-MRF损失，可以简单地使用直接相似度度量(如余弦相似度)来找到生成内容中的补丁的最近邻居。但这一过程往往产生平滑的结构，因为一个平坦的区域容易连接到类似的模式，并迅速减少结构的多样性。我们采用相对距离度量[17,16,22]来建模局部特征与目标特征集之间的关系。它可以恢复如图3(b)所示的细微细节。



(a) (b) (c)

Figure 3: Using different similarity measures to search the nearest neighbors. (a) Inpainting results using cosine similarity. (b) Inpainting results using our relative similarity. (c) Ground truth image where red rectangle highlights the filling region (Best viewed in original resolution and with color).

具体地，用$Yg*$ 代表对缺失区域的修复结果的内容，Y_g^{*L}和 Y^L 分别代表来自预训练模型的第L层的特征。

patch v和s分别来自Y_g^{*L}和Y^L ,定义v与s的相对相似度为：

$$\mathrm{RS}(\mathbf{v}, \mathbf{s}) = \exp\left(\left(\frac{\mu(\mathbf{v}, \mathbf{s})}{\max_{\mathbf{r} \in \rho_{\mathbf{s}}\left(\mathbf{Y}^L\right)} \mu(\mathbf{v}, \mathbf{r}) + \epsilon}\right) / h\right)$$

这里$\mu()$ 是计算余弦相似度。$r \in \rho_s(Y^L)$ 意思是r是Y^L 中除了sd的其他patch.h和ϵ 是两个超参数常数。仔细观察这个相对相似度和原始相似度的关系，会发现如果最高的相似度作为分母的话，那相对相似度就会变小，也就是小的更小，大的更大。接下来：RS(v,s)归一化为：

$$\overline{\mathrm{RS}}(\mathbf{v}, \mathbf{s}) = \mathrm{RS}(\mathbf{v}, \mathbf{s}) / \sum_{\mathbf{r} \in \rho_{\mathbf{s}}\left(\mathbf{Y}^L\right)} \mathrm{RS}(\mathbf{v}, \mathbf{r})$$

最后，根据上式，最终的ID-MRF损失被定义为：

$$\mathcal{L}_M(L) = -\log\left(\frac{1}{Z}\sum_{\mathbf{s}\in\mathbf{Y}^L}\max_{\mathbf{v}\in\hat{\mathbf{Y}}_g^L}\overline{\mathrm{RS}}(\mathbf{v},\mathbf{s})\right)$$

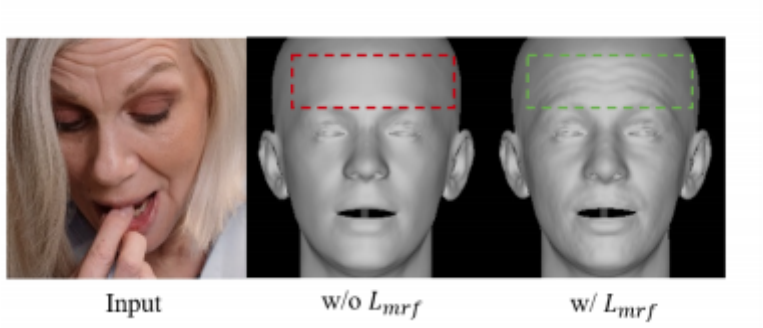
一个极端的例子Y_g^{*L} 中的所有patch都非常接近目标中的一个patch s.而对于其他的patch r $max_v RS(v,r)$ 就会变小。L_m 就会变大。

另一方面，Y^L中的每一个patch r 在Y_g^{*L} 中有一个唯一的最近邻。那么结果就是RS(v,r)变大。L_m就会变小。

从这个观点触发，最小化，LM(L)鼓励Y_g*^L 中的每一个patch v都匹配Y^L中不同的patch.是的变得多样化。

$$L_{mrf} = 2L_M\left(conv4_2\right) + L_M(conv3_2)$$

lmrf的影响



Soft symmetry loss：

对称损失，增加遮挡的损失。

$L_{sym} = \|V_{uv} \odot (D - flip(D))\|_{1,1}$

正则化损失：

$L_{regD} = \|D\|_{1,1}$

4.3细节解耦

核心的依据是，同一个人的不同照片，除了表情控制的细节。其他的细节和大致的形状是不变的。

交换同一个人两张照片的detail code , 不会影响照片的三维重建，也就是说他们的detail code 应该是相同的。

所以构造了如下损失函数：

Detail consistency loss:

$$L_{dc} = L_{detail}\left(I_i, \mathcal{R}\left(M\left(\boldsymbol{\beta}_i, \boldsymbol{\theta}_i, \boldsymbol{\psi}_i\right), A\left(\boldsymbol{\alpha}_i\right)\right.\right.$$
$$\left.\left.F_d\left(\boldsymbol{\delta}_j, \boldsymbol{\psi}_i, \boldsymbol{\theta}_{jaw,i}\right), \mathbf{l}_i, \mathbf{c}_i\right)\right)$$

给出一个人两张不同的照片I_i 和 I_j .损失函数如上所示。其中δ_j 表示I_j 的detail code .

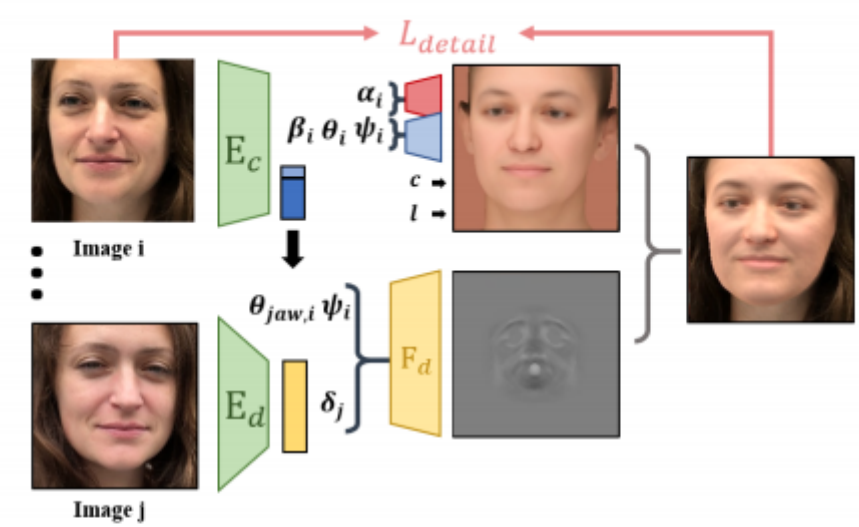


Fig. 3. Detail consistency loss. DECA uses multiple images of the same person during training to disentangle static person-specific details from expression-dependent details. When properly factored, we should be able to take the detail code from one image of a person and use it to reconstruct another image of that person with a different expression. See Sec. 4.3 for details. Images are taken from NoW [Sanyal et al. 2019]. Note that NoW images are not used for training, but are just selected for illustration purposes.

L_{dc} 对模型的影响。