SMPL: A Skinned Multi-Person Linear Model

· Introduction

SMPL 的主要目标是创造出逼真的动画人体模型，使模型可以表现不同的体型，可以随姿势自然地变形，并模拟出像真人一样的软组织运动。作者希望 SMPL 能够具备快速渲染、易于使用、兼容现有渲染引擎的特点。目前，商业上普遍的做法是通过手动建立 rigging 或者人工调整 blend shape 来纠正传统 skinning 会遇到的问题。而想要追求好的动画效果，我们通常需要用到较多的 blend shape，并耗费大量的人力。相对地，在学术上，我们一般会凭借拥有不同体型与姿势的人体扫描模型，来训练出可靠的统计学模型。但使用这种方法得到的模型很难兼容现有的图形软件与渲染引擎。（这一段话翻译自原论文的 Introduction 部分，它主要引出了 SMPL 在动画人体模型上的改进。在整篇论文中，作者使用了较多了篇幅来介绍 SMPL 在动画人体模型方面做出的改进。而考虑到虚拟试衣领域并不涉及动画方面的技术，我在后文中将省去一部分内容的介绍，若有需要的话，可在日后补充。）

目前，在构建顶点与底层骨架结构的关系时，使用得最广泛的模型是 Linear Blend Skinning (LBS)。但 LBS 有时会导致不真实的关节点变形，如 Taffy and Bowtie effects。（我尝试过在网上搜索资料，但最终还是没能找到 Taffy and Bowtie effects 的相关资料。下图分别展示了 5 种 skinning 方法对应的建模效果，其中，最右边的模型为 3D scan，在它旁边的是通过配准获得的 ground truth，其余模型从左到右依次使用了下列 skinning 方法：Linear BS、Dual-quaternion BS、BlendSCAPE、SMPL-LBS、SMPL-DQBS。根据作者的描述，下图的变形问题主要出现在右手肘和胯部两个区域。可以看到，在关节的连接处，LBS 与 DQBS 都不能提供平滑的过渡。至于其他模型，我认为没太大区别。）

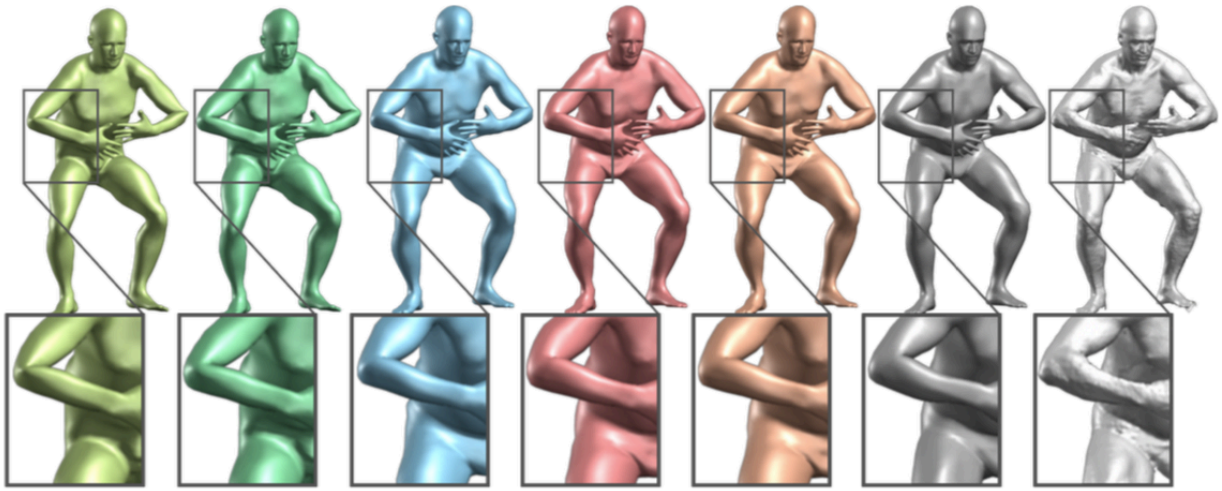


Figure 2: Models compared with ground truth. *This figure defines the color coding used throughout the paper and* **Supplemental Video.** *The far right (light gray) mesh is a 3D scan. Next to it (dark gray) is a registered mesh with the same topology as our model. We ask how well different models can approximate this registration. From left to right: (light green) Linear blend skinning (LBS), (dark green) Dual-quaternion blend skinning (DQBS), (blue) BlendSCAPE, (red) SMPL-LBS, (orange) SMPL-DQBS. The zoomed regions highlight differences between the models at the subject's right elbow and hip. LBS and DQBS produce serious artifacts at the knees, elbows, shoulders and hips. BlendSCAPE and both SMPL models do similarly well at fitting the data.*

与现有的方法相比，SMPL 在让人体模型尽可能简单、标准的同时，保持了模型的真实性。具体上，SMPL 通过学习 blend shape 的方式克服了标准 skinning 的缺陷。在使用 blend skinning 进行变形前，SMPL 动态地将分别表示个体、姿势、软组织的 blend shape 与一个 rest template 叠加在一起。通过将 pose blend shape 视作身体部分旋转矩阵的线性函数，SMPL 能够让基于 blend shape 的训练和动画化变得简单。且外，由于旋转矩阵中的元素是有界的，所产生的变形也会受到限制。

在训练模型方面，SMPL 使用一个目标函数来惩罚配准网格与模型间的顶点差异。SMPL 扫描不同体格、姿势的人体来获取 ground truth，通过优化 blend weight、pose blend shape、mean template shape、与体型相关的关节点回归元来减少模型在训练集上的顶点误差。为了令模型能够反映不同体型下的姿势变形，作者在 CAESAR 数据集上使用 PCA 学习了不同性别的 body shape blend shape。

首先，我想尝试解释下论文中出现的一些专业名词，但因为我自己不是很熟悉这方面的内容，我的理解可能会有误。

rigging：即骨骼绑定，将模型的网格与骨骼绑定在一起。先在模型上选取若干个控制单元作为骨骼，然后建立骨骼与模型网格顶点间的关系。在模型发生变形时，我们只需要改变控制单元的位置，再由控制单元带动模型网格顶点位置的变动。

blend skinning：一个顶点可能由多个骨骼进行控制，如果无法为这些控制关系计算出合适的权值，模型就无法实现自然的变形。blend skinning 是更新权重的过程，即设置骨骼影响的网格的权重。

blend shape：在实现一些比较细节的变形时，如果使用 rigging 的话，需要用到大量的骨骼才能使变形显得真实。根据我的理解，blend shape 即取定多个参考 shape，然后通过插值的方法来生成中间 shape。

总的来说，SMPL 先通过学习的方式，获取一些列的分别反映体型、姿势、软组织的 blend shape，然后使用 weighted pose space deformation 的方法，纠正普通 blend skinning 存在的变形问题（处理蒙皮模型时，如果模型处于特定姿势，通常需要使用特殊形状来修复蒙皮变形问题。变形问题通常出现在模型的关节连接点处，如肩部、下臂、膝盖和腹股沟部位。）。其中，软组织的 blend shape 仅应用在 SMPL 的拓展版 Dynamic-SMPL (DMPL)，因此，在下文将不介绍这方面的内容。

• Model Formulation

SMPL 模型拥有 6890 个网格顶点与 24 个关键点，关节点的具体信息如下：0-Pelvis、1-L_Hip、2-R_Hip、3-Spine1、4-L_Knee、5-R_Knee、6-Spine2、7-L_Ankle、8-R_Ankle、9-Spine3、10-L_Foot、11-R_Foot、12-Neck、13-L_Collar、14-R_Collar、15-Head、16-L_Shoulder、17-R_Shoulder、18-L_Elbow、19-R_Elbow、20-L_Wrist、21-R_Wrist、22-L_Hand、23-R_Hand。

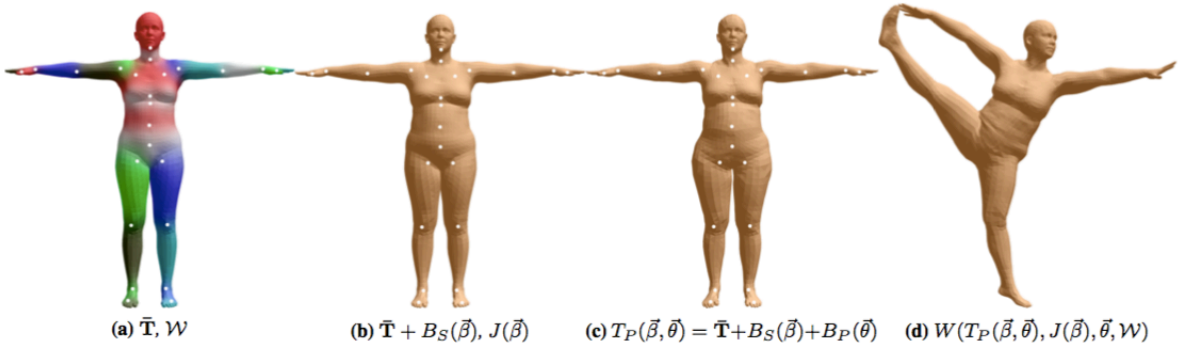


Figure 3: SMPL model. *(a) Template mesh with blend weights indicated by color and joints shown in white. (b) With identity-driven blendshape contribution only; vertex and joint locations are linear in shape vector $\vec{\beta}$. (c) With the addition of of pose blend shapes in preparation for the split pose; note the expansion of the hips. (d) Deformed vertices reposed by dual quaternion skinning for the split pose.*

上图定义了 SMPL 模型。$\bar{T}$ 表示三维顶点参数；$\mathcal{W}$ 表示 blend skinning 权重参数；β 表示体型参数；B_S 表示体型参数到 shape blend shape 的映射关系；J 表示体型参数到关节点的映射关系；θ 表示姿势参数；B_P 表示姿势参数到 pose blend shape 的映射关系；W 表示 blend skinning 函数 (在 SMPL 中为 LBS 或 DQBS)。我们可以看到，SMPL 首先定义了一个 mean template shape (Fig. 3(a))，它仅包括 $\bar{T}$ 和 $\mathcal{W}$。接下来，SMPL 引入 shape blend shape (Fig. 3(b))，重新计算顶点与关键点的位置。之后，SMPL 在上一步基础上引入 pose blend shape (Fig. 3(c))，再重新计算顶点的位置。最后，SMPL 将 template 和两种 blend shape 的集联输入到 blend skinning 函数 (Fig. 3(d))，实现模型的变形。SMPL 模型可以表示成 $M(\beta, \theta; \Phi)$，其中，Φ 为学习参数。下面将简略地描述下 blend skinning、shape blend shape、pose blend shape 等过程的计算。

Blend Skinning。在这一步中，只需根据关节点的位置计算顶点的相对位置即可。首先，J 和 θ 分别有 3 × 24 个参数，表示的是关节点的三维坐标。根据 J 和 θ，我们可以计算出每个关节点对应的旋转矩阵。然后，$\mathcal{W}$ 包含 6890 × 24 个参数，定义了顶点受每个旋转矩阵影响的权重。最后，在上一步的基础上引入 shape blend shape 偏移量与 pose blend shape 偏移量，即可获得变形后的顶点坐标。

Shape Blend Shape。B_S 的公式如下，其中：β 代表线性体型参数；S 代表体型位移的正交主成分。

$$B_S(\vec{\beta}; \mathcal{S}) = \sum_{n=1}^{|\vec{\beta}|} \beta_n \mathbf{S}_n$$

$S \in 3N \times |\beta|$，所以 $B_S \in 3N$，S 的求值将在后文介绍。

Pose Blend Shape。B_P 的公式如下，其中：θ 代表姿势参数，R 代表展开的旋转矩阵，$\mathcal{P}$ 代表姿势位移。

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

因为每个旋转矩阵都是 3 × 3 的大小，所以 $R \in 9K$。在这里，我们用 rest pose 作为基准 (Fig. 3(a) 中模型的姿势)， 则姿势的变化可以表示成 $R(\theta)$ - $R(\theta^*)$。$\mathcal{P} \in 3N \times |\theta|$，所以 $B_P \in 3N$，$\mathcal{P}$ 的求值将在后文介绍。

Joints Location。由于在不同的体型下，关节点处于 rest pose 时的位置将不一样，我们需要学习 $\mathcal{J}(\bar{T} + B_S)$ 来求出不同体型下的 joints loc.。

上文提到的学习参数 $\Phi = \{\bar{T}, \mathcal{W}, S, \mathcal{J}, \mathcal{P}\}$，变形后的顶点参数 $T' = \bar{T} + B_S + B_P$。

· Training

首先，作者使用配准的方法将训练集中的 3D scan 转化成与 SMPL 模型一样的拓扑结构 (这个过程叫 registration，我们把得到的结果视为 ground truth)。训练中用到的数据集有 multi-shape dataset 和 multi-pose dataset，下图为数据集中的部分模型。

Figure 4: *Sample registrations from the multipose dataset.*

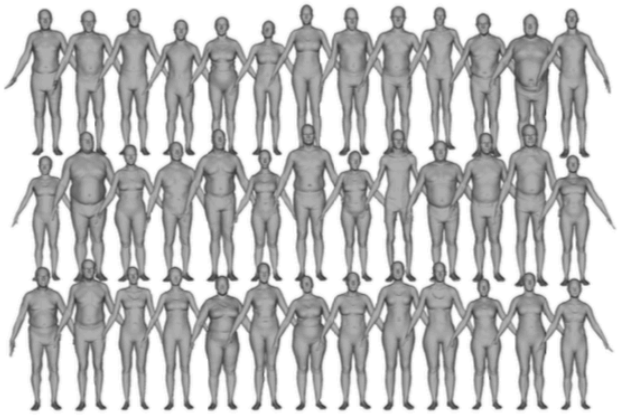


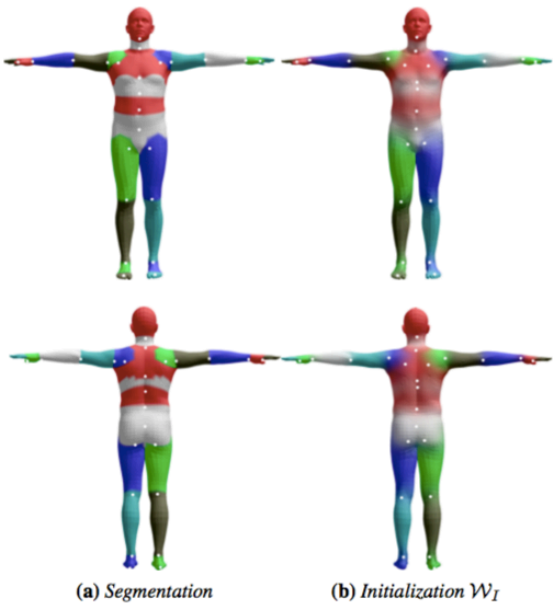
Figure 5: *Sample registrations from the multishape dataset.*

作者一共训练了两组学习参数 ($\{\Phi_m, \Phi_f\}$，分别用于男性与女性)，先使用 multi-pose dataset 来训练 $\{\mathcal{W}, \mathcal{J}, \mathcal{P}\}$，再用 multi-shape dataset 来训练 $\{\bar{\mathbf{T}}, \mathcal{S}\}$。我们用 $\mathbf{V}_j$ 表示数据集中的第 j 个 registration。

Pose Parameter Training。multi-pose dataset 中包括 20 个女性的共 891 个 registration 与 20 个男性的共 895 个 registration。我们把一个人称为 subject，那么在训练中，一个 subject 对应若干个不同姿势的 registration。首先，我们针对每一个 subject 计算出其处在 rest pose 时对应的顶点参数与关节点参数 (由于每个人体型都不一样，所以每个 subject 都有独立的参数)，分别记为 $\mathbf{T}$、$\mathbf{J}$ (文章并没有介绍计算 $\mathbf{T}$ 的方法)。之后，令 $\mathbf{T}_{s(j)}$ 表示第 j 个 registration 对应的 $\mathbf{T}$，$\mathbf{J}_{s(j)}$ 表示第 j 个 registration 对应的 $\mathbf{J}$。我们用以下的公式来学习 $\{\mathcal{W}, \mathcal{P}\}$，即最小化生成模型与 ground truth 顶点的欧氏距离和。

$$E_D(\hat{\mathbf{T}}^P, \hat{\mathbf{J}}^P, \mathcal{W}, \mathcal{P}, \Theta) = \sum_{j=1}^{P_{\text{reg}}} ||\mathbf{V}_j^P - W(\hat{\mathbf{T}}_{s(j)}^P + B_P(\vec{\theta}_j; \mathcal{P}), \hat{\mathbf{J}}_{s(j)}^P, \vec{\theta}_j, \mathcal{W})||^2$$

作者将模型手工划分成了 24 个区域，每个部分都有对应的初始化关节点与 blend weight，其中，blend weight 的初值是通过在一个区域中进行漫射获得的。



(a) *Segmentation* **(b)** *Initialization* $\mathcal{W}_I$

Figure 6: Initialization of joints and blend weights. *Discrete part segmentation in (a) is diffused to obtain initial blend weights, $\mathcal{W}_I$, in (b). Initial joint centers are shown as white dots.*

且外，为了进了一步优化模型的表现，作者还定义了其他的误差项。

$$E_Y(\hat{\mathbf{J}}^P, \hat{\mathbf{T}}^P) = \sum_{i=1}^{P_{\text{subj}}} \lambda_U ||\hat{\mathbf{J}}_i^P - U(\hat{\mathbf{J}}_i^P)||^2 + ||\hat{\mathbf{T}}_i^P - U(\hat{\mathbf{T}}_i^P)||^2$$

E_Y 用以限定生成模型的对称性，U 为纵向面翻转函数。

$$E_J(\hat{\mathbf{T}}^P, \hat{\mathbf{J}}^P) = \sum_{i=1}^{P_{\text{subj}}} ||\mathcal{J}_I \hat{\mathbf{T}}_i^P - \hat{\mathbf{J}}_i^P||^2$$

E_J 用以惩罚预测关节点与生成关节点的欧氏距离，$\mathbf{J}_i$ 可基于 $\mathbf{T}$ 用非负最小二乘获得。

$$E_P(\mathcal{P}) = ||\mathcal{P}||_F^2$$

$$E_W(\mathcal{W}) = ||\mathcal{W} - \mathcal{W}_I||_F^2$$

E_P 与 E_W 是防止过拟合的正则化项。

$$E_*(\hat{\mathbf{T}}^P, \hat{\mathbf{J}}^P, \Theta, \mathcal{W}, \mathcal{P}) = E_D + \lambda_Y E_Y + \lambda_J E_J + \lambda_P E_P + E_W$$

至此，Pose Parameter Training 的误差项如上。

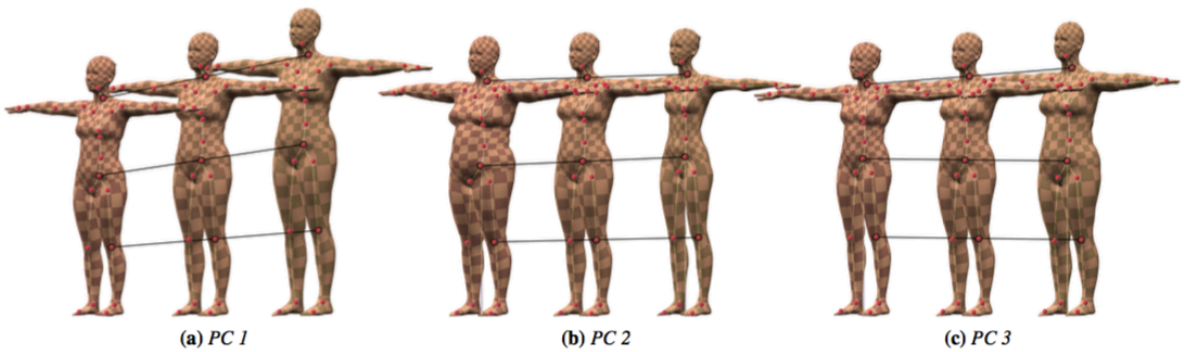
Shape Parameter Training。首先，multi-shape dataset 中的 registration 的姿势是不一致的，因此，我们需要先将它们全部转换至 rest pose (我们称这个过程为 pose normalization)。在 pose normalization 中，我们需要使用上一步训练中得到的参数 $\{\mathcal{W}, \mathcal{J}, \mathcal{P}\}$。我们先把 multi-pose dataset 中所有 registration 的平均顶点参数与平均关节点参数分别记作 $\mathbf{T}_\mu$ 与 $\mathbf{J}_\mu$，然后用 W_e 与 $V_{j,e}$ 分别表示生成模型与 registration 的边，其中，边的定义为邻居顶点间的连线。然后，我们用下面的公式预测出 multi-shape dataset 中不同 registration 的姿势参数。

$$\arg\min_{\vec{\theta}} \sum_e ||W_e(\hat{\mathbf{T}}_\mu^P + B_P(\vec{\theta}; \mathcal{P}), \hat{\mathbf{J}}_\mu^P, \vec{\theta}, \mathcal{W}) - \mathbf{V}_{j,e}^S||^2$$

预测出姿势参数后，我们再使用下列公式预测出 multi-shape dataset 中不同 registration 处在 rest pose 姿势下的顶点参数。

$$\hat{\mathbf{T}}_j^S = \arg\min_{\hat{\mathbf{T}}} ||W(\hat{\mathbf{T}} + B_P(\vec{\theta}_j; \mathcal{P}), \mathcal{J}\hat{\mathbf{T}}, \vec{\theta}_j, \mathcal{W}) - \mathbf{V}_j^S||^2$$

在完成 pose normalization 后，我们可以通过 PCA 求得 $\{\mathcal{W}, \mathcal{P}\}$。



(a) *PC 1* (b) *PC 2* (c) *PC 3*

Figure 8: Shape blend shapes. *The first three shape principal components of body shape are shown. PC1 and PC2 vary from -2 to +2 standard deviations from left to right, while PC3 varies from -5 to +5 standard deviations to make the shape variation more visible. Joint locations (red dots) vary as a function of body shape and are predicted using the learned regressor,* $\mathcal{J}$.

我们能看到，不同 body shape 的模型的关节点位置是不一样的。

· Evaluation

在 Quantitative Evaluation 中，作者评估了两种误差：Model Generalization 与 Pose Generalization。其中，Model Generalization 同时衡量 shape blend shape 与 pose blend shape 的效果；Pose Generalization 仅衡量 pose blend shape 的效果。在测试过程中，作者使用了 Dyna dataset 中来自 4 个女性与 2 个男性 的共 120 个 registration。我们可以先从下图直观地感受下 SMPL 的效果，在这里，我们使用 BlendSCAPE 方法生成的模型作为参照。

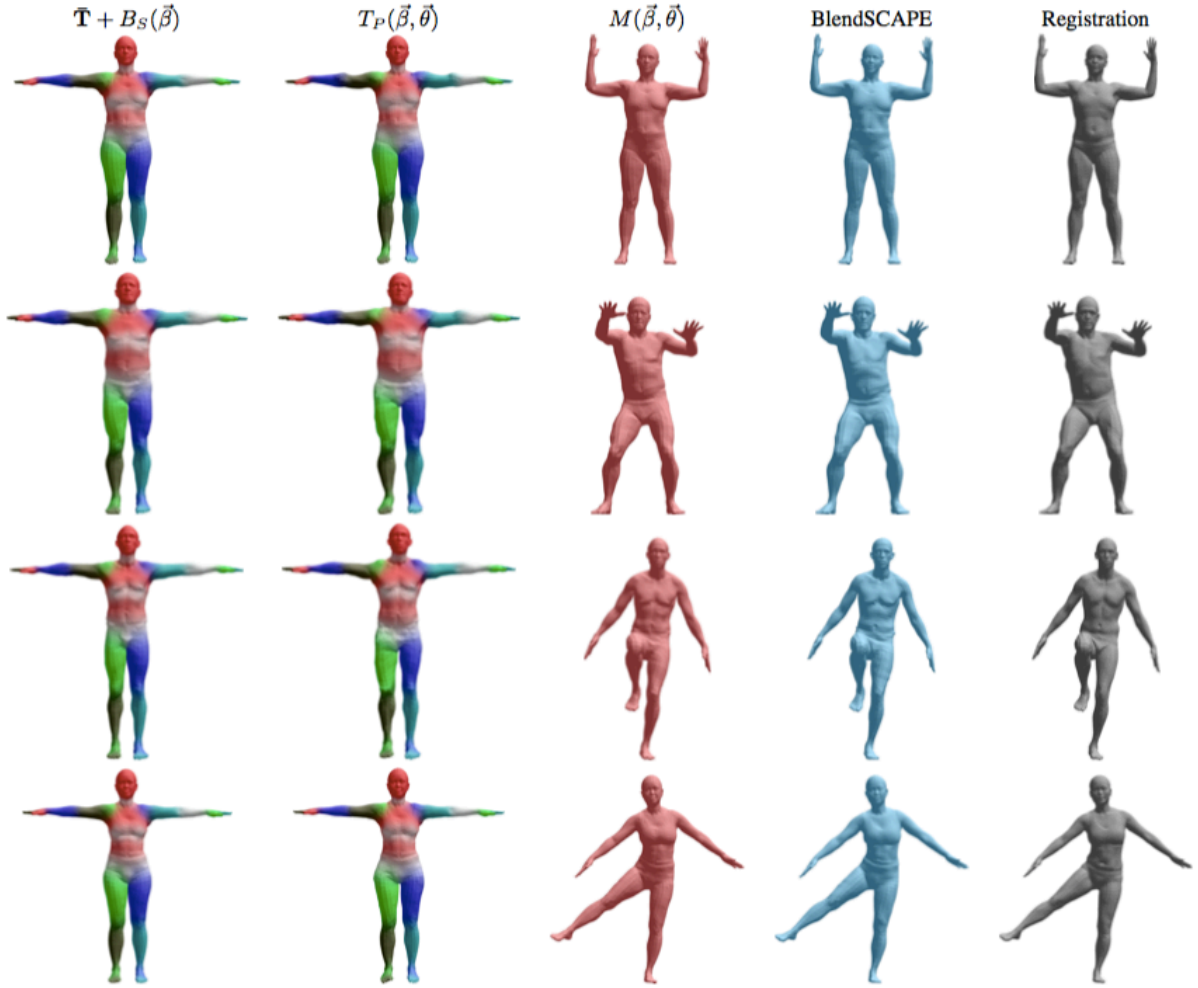


Figure 10: *Model fitting with intermediate stages. We fit both BlendSCAPE (blue) and SMPL-LBS, $M(\vec{\beta},\vec{\theta})$, (red) to registered meshes by optimizing pose and shape. $\bar{\mathbf{T}} + B_S(\vec{\beta})$ shows the estimated body shape and $T_P(\vec{\beta},\vec{\theta})$ shows the effects of pose-dependent blend shapes. Here we show SMPL-LBS, because T_P shows more variation due to pose than SMPL-DQBS.*

为了评估 Model Generalization 的表现，作者先根据 ground truth 生成一个模型，然后用生成模型与 ground truth 顶点的平均欧氏距离作为误差。

至于 Pose Generalization 表现的评估，我们可以认为 pose blend shape 是不依赖于 shape blend shape 的。因此，对于某一个 subject，如果我们已经获得了一组 shape 参数，那么这一组 shape 参数理应适用于这个 subject 的所有 pose blend shape。以此为前提，作者以 subject 为单位进行考虑：作者先从一个 subject 的所有 registration 中任意挑选一个，然后对此 registration 执行正常的模型生成流程；然后，我们就可以获得一组 shape 参数与一组 pose 参数；接下来，我们把这组 shape 参数应用在该 subject 的其余 registration 对应的模型生成中；最后，我们将生成模型与 ground truth 顶点的平均欧氏距离作为误差。

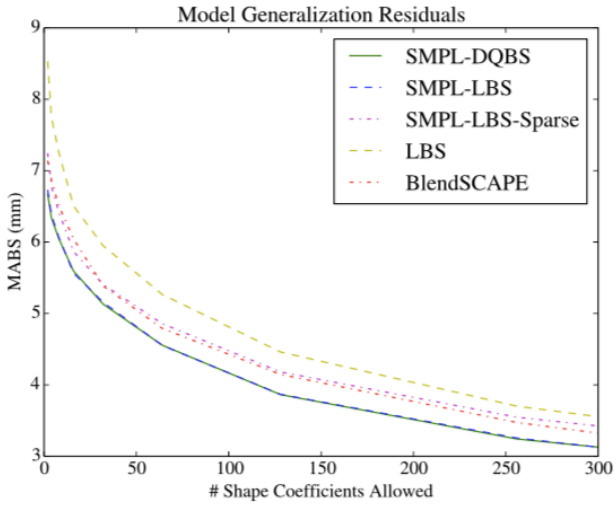


Figure 11: *Model generalization indicates how well we can fit an independent registration. Mean absolute vertex error versus the number of shape coefficients used.*

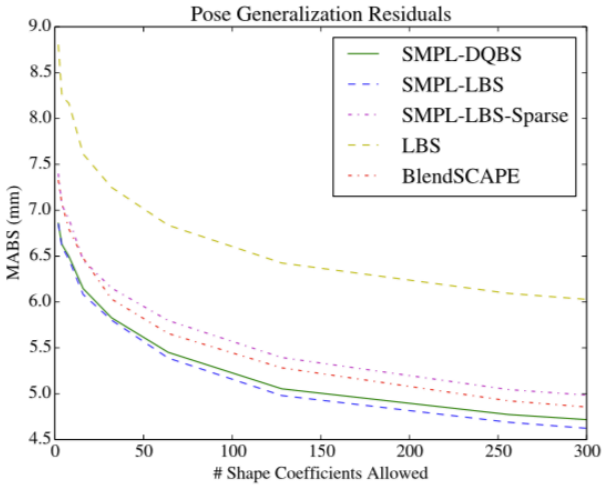


Figure 12: *Pose generalization error indicates how well a fitted shape generalizes to new poses.*

Quantitative Evaluation 的结果如上，我们可以看到 SMPL-LBS 与 SMPL-DQBS 的效果是最好的，且随着 shape 参数个数的增多，误差会趋向于 0。

Sparse SMPL。在这里，sparse 指的是一个顶点的变形只由个别关节点控制。在 SMPL 中，pose blend shape 是 dense 的，一个顶点由所有的关节点共同控制，因此，$\mathcal{P} \in 3\mathrm{N} \times |\theta|$ (即 3 × 顶点个数 × 9 × 关节点个数)；在 Sparse SMPL 中，作者限制一个顶点最多由 4 个关节点共同控制 (防止一个顶点被与其距离较远的关节点控制)，因此，$\mathcal{P} \in 3\mathrm{N} \times 4$。但结果证明，这个处理不仅复杂了训练过程，而且不能使模型有更好的表现 (主要原因估计是 4 个关节点的限制过于严格了，如果增加这个数字的话，表现应该会提升)。

Run-time Performance。SMPL 的开销取决于 blend skinning 与 blend shape。作者在 CPU 上分别测试 SMPL 与 BlendSCAPE 的运行速度，其中：SMPL 使用单核运行，BlendSCAPE 使用多核运行。作者在测试过程中，改变 shape 参数的个数，我们可以发现，无论 shape 参数的个数为何值，SMPL 的表现都要明显优于 BlendSCAPE。

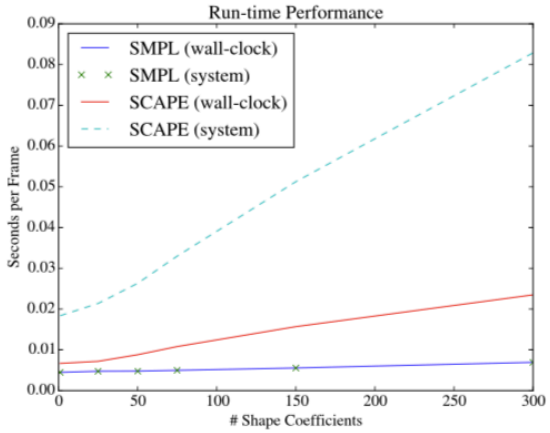


Figure 14: *Performance of SMPL and BlendSCAPE vary with the number of body shape coefficients used. Performance shown here is from a 2014 Macbook Pro.*

Visual Evaluation。下图中，水平轴代表不同的 subject (同一 subject 的 shape 相同)，竖直轴代表不同的姿势。

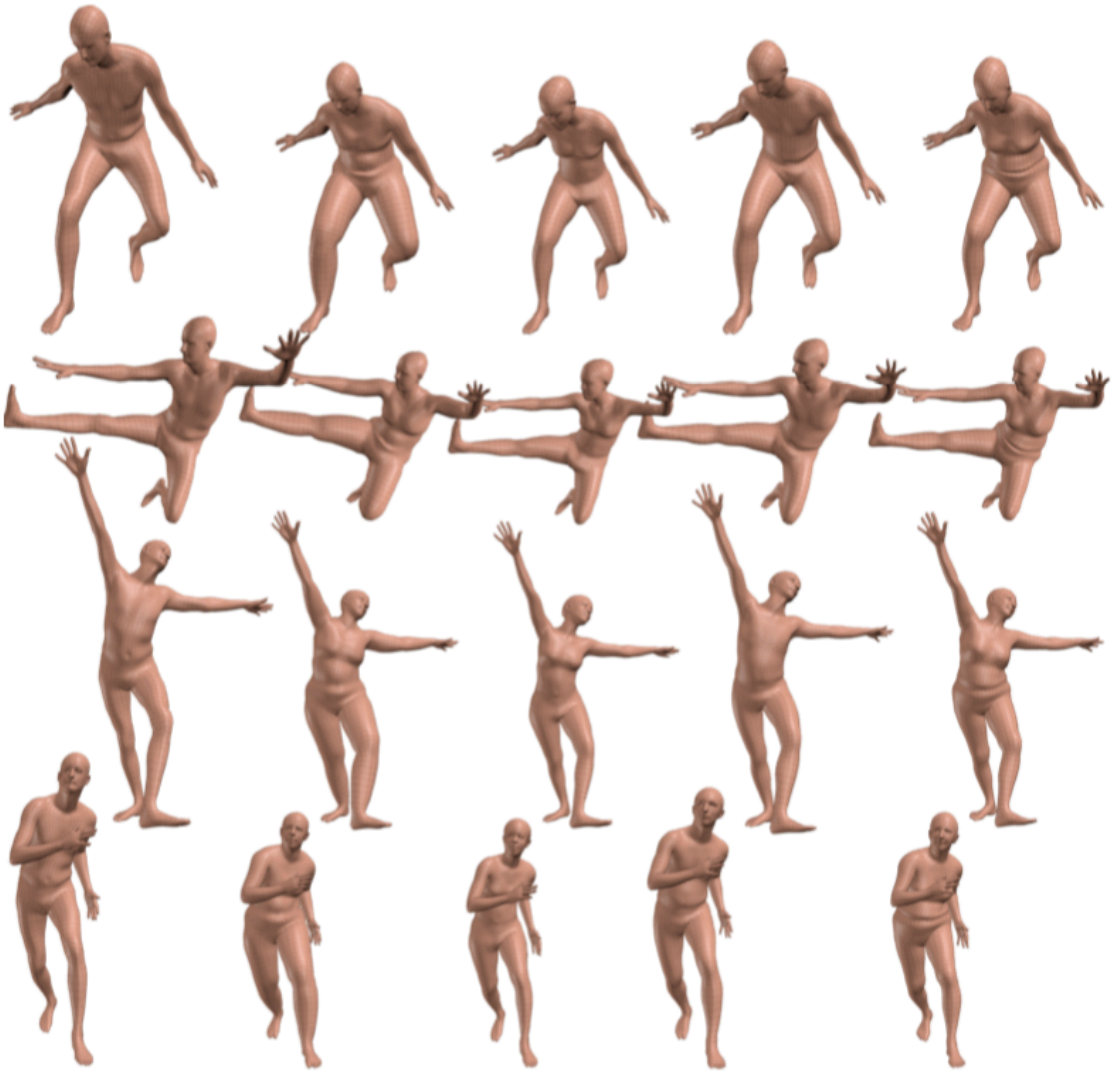


Figure 13: Animating SMPL. *Decomposition of SMPL parameters into pose and shape: Shape parameters, $\vec{\beta}$, vary across different subjects from left to right, while pose parameters, $\vec{\theta}$, vary from top to bottom for each subject.*

Discussion

Advantage

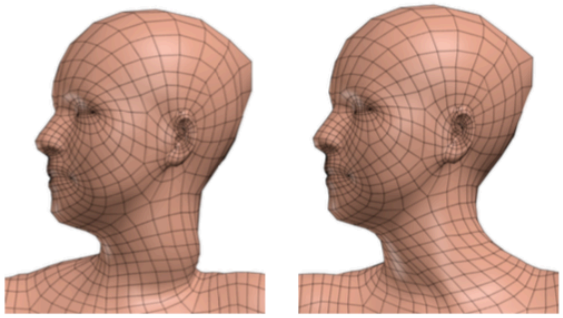
SMPL 的良好表现主要得益于以下要素：拥高质量的训练数据，训练数据覆盖大范围的体型；简单的模型定义使得 SMPL 能够使用大量的数据学习多个参数。

在实现 blend shape 的过程中，与传统的散乱点插值算法 (scattered-data interpolation methods) 相比，SMPL 通过学习的方式，从大量的训练数据中学习到了多种姿势。且外，SMPL 的 blend shape 函数是关于旋转矩阵的线性函数，复杂度要比传统方法要小。使用关于旋转矩阵的线性函数不仅能生成更多不同的姿势，还能在加快渲染速度的同时，防止模型的不自然扭曲 (因为旋转矩阵的元素是有界的)。

尽管 SCAPE 方法也和 SMPL 一样，同时考虑了 shape blend shape 与 pose blend shape，SCAPE 比较依赖三角变形。使用 SCAPE 的话，体格较大的人在发生姿势变形时，变形幅度会比较大。且外，在遇到没学习过的体格时，模型的表现会下降。

在 BlendSCAPE 的训练过程中，我们的目标并不是直接再现出 registration，而是训练出一个能够再现 registration 的变形。因此，BlendSCAPE 无法支持关节点对与其距离较远的顶点的控制，且无法表示顶点间的相互关系。

在考虑颈部的扭转时 (会导致关于竖直轴的正角度旋转或负角度旋转)，标准的 LBS 会令模型的颈部往某一个方向收缩。为了解决这个问题，我们需要使用 blend shape 来增大颈部扭转时，模型颈部的体积。而使用欧拉角作为姿势参数的话，由于欧拉角的范围只有 $[-\pi, \pi]$，模型将无法同时解决右向扭转问题与左向扭转问题 (SMPL 可以解决这个问题，同时，作者推测使用四元数旋转也有可能解决这个问题，但并没有进行试验)。



(a) *Euler-angles* (b) *SMPL parameterization*

Figure 16: Parameterizing pose blend shapes. *(a) Pose blend shapes parameterized by Euler angles cause significant problems. (b) our proposed parameterization allows the head to rotate in either direction with natural deformations.*

对比起 weighted pose-space deformation，SMPL 的运算速度要更快。在 WPSD 中，我们使用具体数目的 template pose，在运算时使用 RBFs 来插值完成 blend skinning 的纠正。在实际应用过程中，这样的操作需要搜索大量的 template pose，从而导致渲染的速度变慢。

Limitation

在 SMPL 中，与姿势变形有关的偏移量是不依赖于体格的。因此，当 SMPL 在处理一些不真实的人体模型时，通常没法获得较好的效果 (比如一个人人体的各部分的 scale 相差较大时)。(其实作者还提到了另一种情况，但是我没有看懂那句话的意思：or a space of humans that includes infants and adults)

事实上，我们可以在训练 pose blend shape 相关参数时，同时考虑旋转矩阵与 shape 参数 β。借此，我们有可能在不会大幅增大 run-time cost 的同时，解决上述问题。

且外，SMPL 是仅依赖于关节点与 shape 参数的模型 — 它无法模拟出呼吸、面部运动等不依赖于关节点的运动。但是，我们可以仿照 DMPL 使用的方法 (原论文中有介绍)，添加一些另外的 blend shape 来解决这个问题。

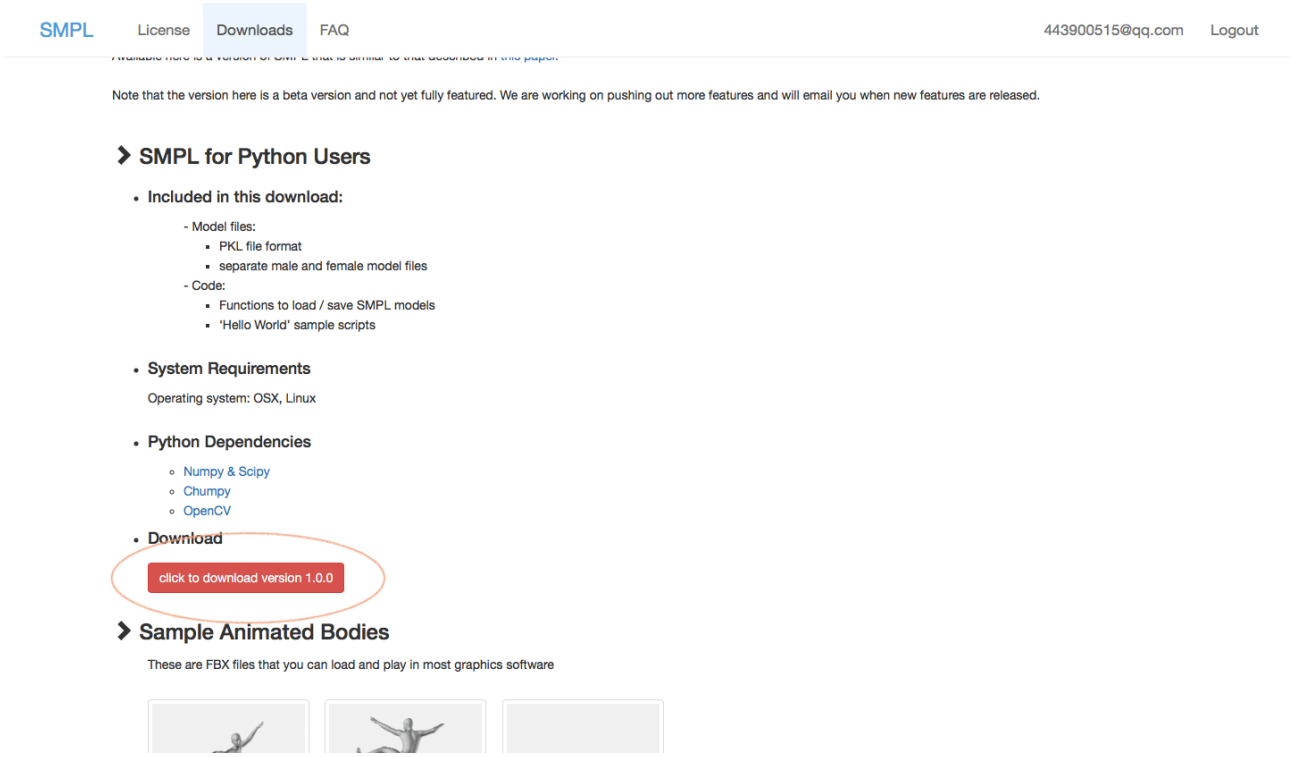
在 Shape Parameter Training 的过程中，作者使用了 PCA 来提升模型效率。实际上，在 Pose Parameter Training 中，也同样能使用 PCA 来提升效率。

最后，SMPL 并没有通过学习的方式获得所有的模型参数。在 SMPL 中，网格的拓扑结构、rest pose、人体区域的划分 (共 24 个区域) 都是手工定义的。原理上，这些参数也可以通过学习的方式获得，但考虑到对结果的提升可能并不大，为了简单起见，在此采用了手工定义的方式。

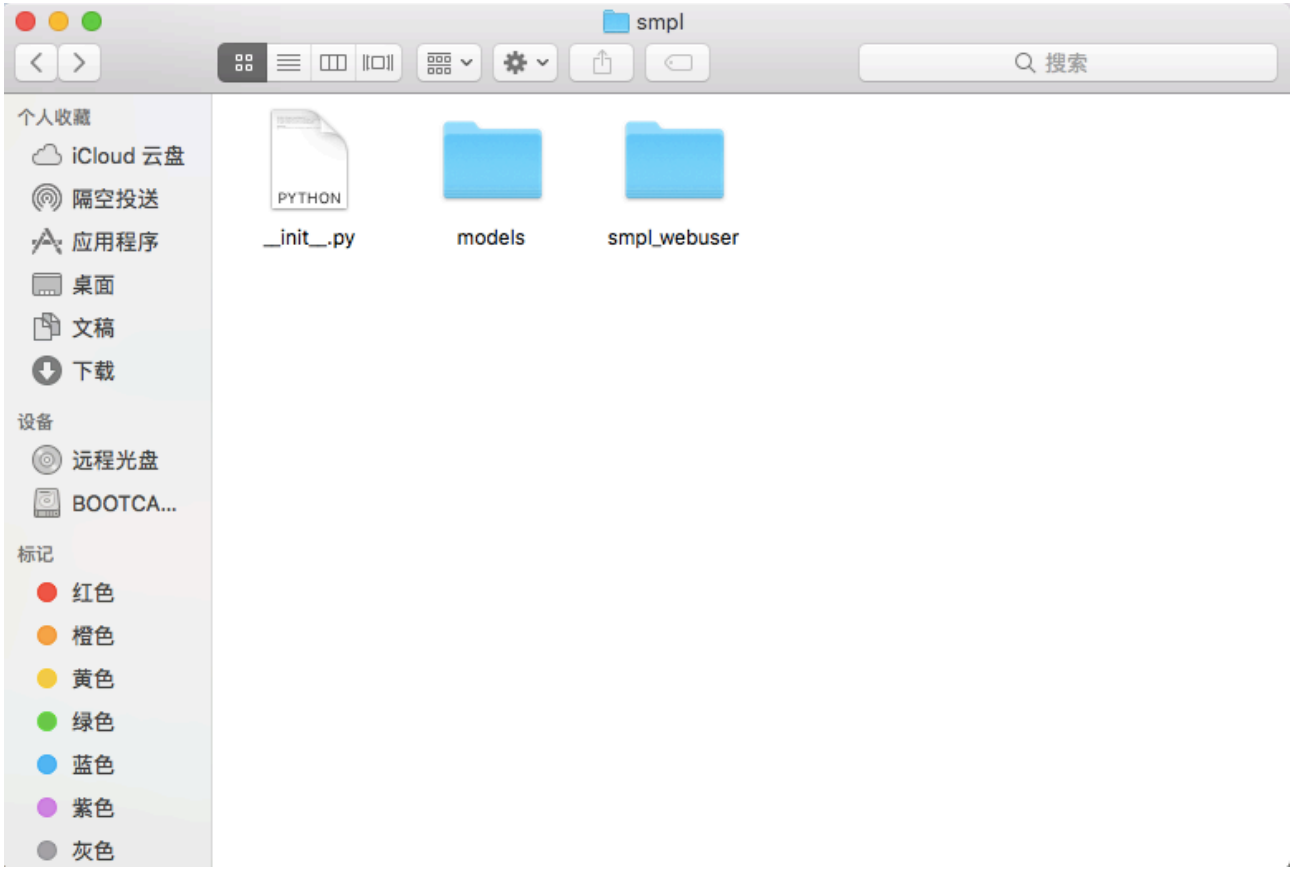
· Appendix

如果我们想要创建一个新的 SMPL 模型，可以使用 Maya、Unity、Python 来完成这一任务。

首先，我们在官网上注册一个账号，并下载 SMPL for Python Users。



将下载好的文件解压后，我们可以看到如下的目录结构。



其中，models 文件夹下存放的是 SMPL 的 template 模型 (包括 female 和 male 两个模型)；smpl_webuser 文件夹下存放的是代码文件。

SMPL 的 Python 代码是使用 Python 2.x 编写的，它主要存在以下问题：Python 2.x 与目前主流的 Python 3.x 不兼容，且当中用到的不少库已经停止了维护 (如 chumpy、opendr 等)；SMPL 的 Python 代码不支持 GPU 运算，导致运行效率差。对于代码方面的问题，我们可以使用基于 Python 3.x 的非官方的 CalciferZh/SMPL 作为替代，或参考我之前写的一篇博客，手动将代码修改成 Python 3.x 的语法。(这里需要注意，使用 CalciferZh/SMPL 的话能够改进代码效率且令代码支持 GPU 运算；而手动修改的话，只能让代码兼容 Python 3.x，不会对运行有任何影响)

接下来，我们把重点放在模型部分。models 文件夹下包括 basicModel_f_lbs_10_207_0_v1.0.0.pkl 与 basicmodel_m_lbs_10_207_0_v1.0.0.pkl 两个文件，这两个文件的储存格式为 pickle。在这里，pickle 的作用主要是实现 dict 变量的序列化与反序列化。当我们使用 Python 打开官方提供的 template model，我们能得到一个拥有下列 key 的 dict 变量。

dict_keys(['J_regressor_prior', 'posedirs', 'J_regressor', 'weights', 'vert_sym_idxs', 'J', 'v_template', 'kintree_table', 'bs_type', 'shapedirs', 'f', 'bs_style', 'weights_prior', 'pose_training_info'])

如果我们不打算重新训练 SMPL 的话，我们无须关注上述的 key，因此，我只简略介绍一下上述的 key。但是，由于文档中没有介绍这部分的内容，我只能介绍一下自己在研究代码后的见解。J_regressor_prior，24 × 6890 的矩阵，可能与上文提到的 J 有关 (J 表示体型参数到关节点的映射关系)，在代码中并没有使用到这些数据；J_regressor，24 × 6890 的矩阵，即 J；v_template，即 $\bar{\mathrm{T}}$，表示三维顶点参数；J，即关节点坐标；bs_style，即 blend skinning 使用的函数 (LBS 或 DQBS)；bs_type，实际上只支持 lrotmin 一种 pose mapping 方法，通过 Rodrigues 公式，将姿势参数 θ 转换成旋转矩阵；weights，即 $\mathcal{W}$，包含 6890 × 24 个参数，定义了顶点受每个旋转矩阵影响的权重；weights_prior，可能与 $\mathcal{W}$ 有关，但代码中没有使用到这些数据；posedirs，即 B_P，表示姿势参数到 pose blend shape 的映射关系；shapedirs，即 B_S，表示体型参数到 shape blend shape 的映射关系；kintree_table，即 kinematic tree；pose_training_info，即模型的一些相关信息，包括性别等；f，即对应 obj 文件的 face 参数；vert_sym_idxs，定义顶点的对称关系。

一般来说，在创建新的模型时，我们只需要读取 template model，然后改变 β 与 θ 即可，它们分别与键值 'beta' 和 'pose' 相对应。这两个键值需要我们在读取 template model 后，自行添加进 dict 变量中。对于 pose，我们是比较好理解的，即每个关节点的旋转角；而对于 beta，作者的定义是比较模糊的。beta 代表使用 PCA 获得的 10 个代表人体体型的参数，在翻阅了不少资料后，我在官方的 Unity 教程中找到了以下描述。

> The shape parameters are the principal components of the shape variation learned from 3D scans of thousands of people. So there is no specific definition of what body part each of the shape parameters changes. Here's a rough layout for the first few:
> - Shape000: height / overall scale of the model
> - Shape001: weight
> - Shape002: torso height + shoulder width
> - Shape003: chest breadth + neck height
> - ...

这代表 beta 没有明确的实际意义，在实际建模时，我们需要依赖经验来取值。而依靠经验的做法显然是不适用于研究中的，因此，我们可以给出模型的关节点，然后采用回归的方式，计算出 pose 与 beta。

• SMPLify

在我之前的一篇博文中有提到，我们可以使用 SMPLify 来把一幅 2D 人像拟合成 SMPL 模型。详细的过程可以参照我之前的博文，具体原理是：先获得 2D 关节点，再使用回归的方式拟合出模型。首先，我们初始化一组 β 与 θ，并获得一个 SMPL 模型；然后，通过之前提到的 J，我们可以获得模型对应的预测关节点；之后，基于预先确定好的 SMPL 关节点与 2D 关节点的对应关系，我们计算预测关节点和 2D 关节点的欧氏距离来优化 β 与 θ，最终获得一个拟合模型。