基于 MPEG-4 的人脸表情图像变形研究

戴振龙 朱海一 张 申 贾 珈 蔡莲红
(清华大学计算机系,北京 100084)

摘 要 为了实时地生成自然真实的人脸表情,提出了一种基于 MPEG-4 人脸动画框架的人脸表情图像变形方法。该方法首先采用 face alignment 工具提取人脸照片中的 88 个特征点;接着在此基础上,对标准人脸网格进行校准变形,以进一步生成特定人脸的三角网格;然后根据人脸动画参数(FAP)移动相应的面部关键特征点及其附近的关联特征点,并在移动过程中保证在多个 FAP 的作用下的人脸三角网格拓扑结构不变;最后对发生形变的所有三角网格区域通过仿射变换进行面部纹理填充,生成了由 FAP 所定义的人脸表情图像。该方法的输入是一张中性人脸照片和一组人脸动画参数,输出是对应的人脸表情图像。为了实现细微表情动作和虚拟说话人的合成,还设计了一种眼神表情动作和口内细节纹理的生成算法。基于 5 分制(MOS)的主观评测实验表明,利用该人脸图像变形方法生成的表情脸像自然度得分为 3.67。虚拟说话人合成的实验表明,该方法具有很好的实时性,在普通 PC 机上的平均处理速度为 66.67 fps,适用于实时的视频处理和人脸动画的生成。

关键词 图像变形 人脸表情 MPEG-4
中图法分类号: TP391.41 **文献标识码**: A **文章编号**: 1006-8961(2009)05-782-10

MPEG-4 Based Facial Expression Image Morphing

DAI Zhen-long, ZHU Hai-yi, ZHANG Shen, JIA Jia, CAI Lian-hong
(*Department of Computer Science, Tsinghua University, Beijing* 100084)

Abstract Human face morphing is the foundation of facial expression synthesis and talking avatar animation. In this paper, an MPEG-4 based method for human face morphing and expression synthesis is proposed. The method uses a picture of neutral human face and a group of face animation parameters (FAP) as input, and the output is a corresponding facial expression image. There are four stages: facial feature point extraction, mesh generation for specific human face, mesh points movement driven by FAPs, and face texture mapping. After these four stages, photos of various facial expressions are created. Novel algorithms are also implemented for eyeball movement and texture mapping inside the mouth. Perceptual evaluation shows that the face morphing method can synthesize realistic and natural facial expressions for various human face models of different genders, ages and races. Meanwhile, this method is real-time, so it can be used in the areas of video processing and facial animation.

Keywords face morphing, facial expression, MPEG-4

1 引 言

目前人脸动画技术在电影制作、游戏娱乐、3 维虚拟交互等领域都有着非常广泛的应用。人脸动画技术的一些高级应用,如虚拟说话人、表情脸像合成等在底层实现上都需要对原始人脸图像或人脸网格模型进行变形,以取得逼真自然的动画效果。由于人脸器官结构和表情动作的复杂性和适应不同人脸以及满足动画的实时性要求,需要对人脸变形进行

基金项目:国家自然科学基金项目(60433030, 60805008);国家重点基础研究发展计划(973)项目(2006CB303101)
收稿日期:2008-10-15;改回日期:2008-12-12
第一作者简介:戴振龙(1988 ~),男。清华大学计算机科学与技术系本科生。研究方向为情感计算等。E-mail:zhenlong.dai@gmail.com

万方数据

参数化和标准化。

文献[1]中提出,MPEG-4 采用基于对象(object-based)的编码方式,通过专门定义的人脸动画框架来定义量化人脸动作。MPEG-4 使用以下 2 个参数集合来定义人脸:FDP 和 FAP。脸部定义参数(FDP)用来定义人脸的几何结构(图 1);脸部动画参数(FAP),其表示了一个完整的基本脸部动作集合。脸部动画参数单位(FAPU),表示 FAP 值的度量(图 1)。

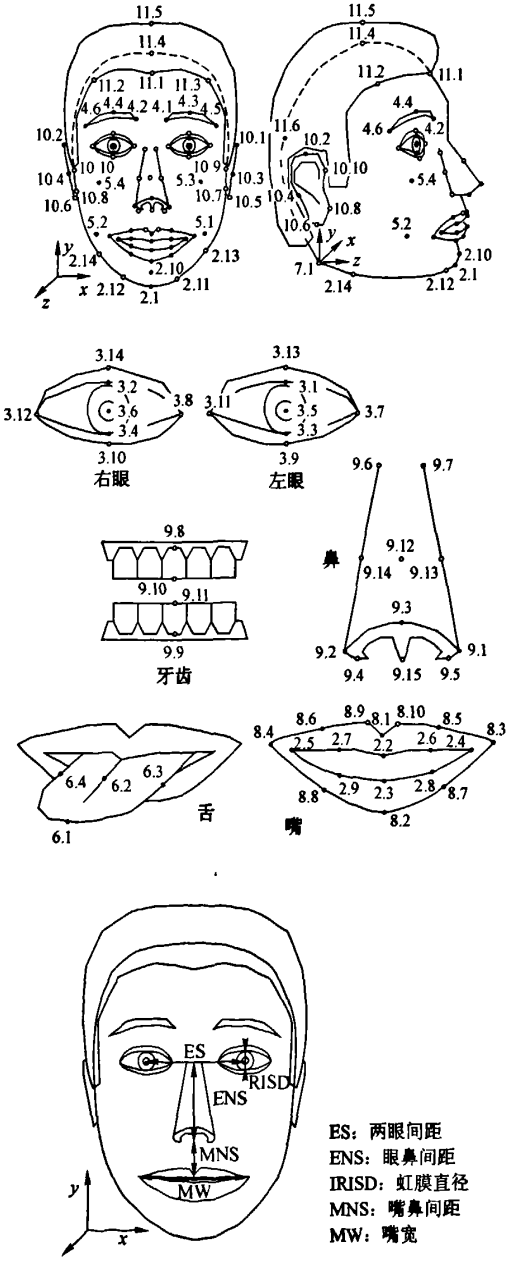

图 1　MPEG-4 定义的脸部特征点及脸部动画参数单位

Fig. 1　Facial feature points and FAPU defined by MPEG-4

文献[2]应用 FAP 驱动 3 维人脸模型来生成表情脸像,取得了很好的效果。3 维人脸模型一般都具有确定的网格结构,其不但面部特征点与固定网格点一一对应,便于生成人脸动画,同时也比较容易得到真实感很好的表情脸像,但是由于 3 维人脸模型往往结构复杂,且更换不同人脸模型及定义人脸动画规则往往费时费力,无法满足虚拟说话人及人机交互系统中对不同风格脸像的应用需求,因此本文在 3 维人脸表情生成研究的基础上,将 FAP 驱动的表情脸像生成方法应用于 2 维人脸照片,实现了基于 FAP 的人脸表情图像生成算法。在模型建立方面,基于 2 维人脸照片的表情脸像生成相对比较灵活,不仅可以从照片或者摄像头视频中直接获取人脸图像,甚至对于卡通化的人脸图片,都能够进行相应的表情脸像编辑和实现不同风格不同个体的表情脸像生成。

在 MPEG-4 定义的人脸动画框架中,虽然给出了与人脸肌肉动作相对应的动画参数描述,然而在表情脸像生成、虚拟说话人等高级应用中还需要对自然细微的人脸变形动作进行模拟。本文在 FAP 参数驱动的人脸网格变形的基础上,对眼神表情动作和口内的牙齿、舌头等细节纹理生成进行了特别处理,从而使得表情脸像生成更为丰富自然,而且虚拟说话人的言语动作中也更为自然逼真。

2　人脸变形系统架构

本文提出的人脸图像变形框架共分为以下 3 个主要模块:

(1)人脸特征点提取　输入为一张正面人脸的中性状态照片。人脸中性状态的定义见文献[1]。该模块能够自动在输入照片中找到与表情动作紧密相关的面部器官特征点以及其他的辅助特征点,可将其作为个性化人脸网格生成的控制点。

(2)个性化人脸网格生成　根据从人脸照片中提取出的特征点,对标准人脸网格进行校准调整,首先生成一张与输入的人脸照片匹配对应个性化人脸网格;然后在个性化人脸三角网格上进行与表情生成相关的人脸器官变形。

(3)人脸网格变形　根据表情脸像所对应的 FAP,对相应的网格点进行移动,并对顶点坐标发生

万方数据

变化的三角形网格区域，通过仿射变换进行纹理填充。

整体的系统框架流程如图 2 所示：

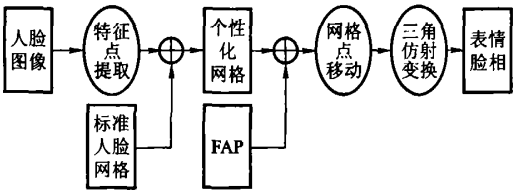

图 2 人脸变形算法基本流程

Fig. 2 Flow chart of face morphing system process

2.1 人脸特征点提取

人脸特征点的准确提取和面部器官定位是实现人脸图像变形的基础。文献[3]提出的改进 ASM (active shape model)算法与经典 ASM 和 Gabor ASM 相比，有更高的精确度和鲁棒性，其在性能上，对于 180×180 大小的人脸图像的变形速度平均可以达到 21.7 fps。因此本文采用基于这种算法所实现的 face alignment 程序提取出了面部的 88 个特征点。这些特征点包括了眼部、唇部、眉毛等器官的重要特征点，特征点的分布如图 3 所示，与各个面部器官(区域)对应的特征点个数如表 1 所示。

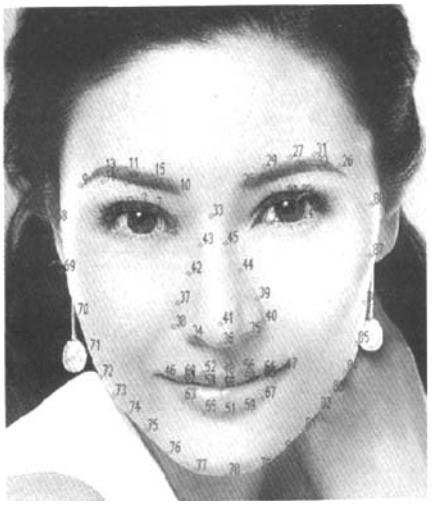

图 3 利用文献[3]中算法提取的人脸特征点

Fig. 3 Feature points extracted by advanced ASM algorithm

这 88 个点基本上完整准确地描述出了脸部轮廓、眉毛、眼睛、鼻子和嘴部(包括内唇和外唇)特征。

表 1 人脸特征点分布统计

Tab. 1 Feature points distribution table

区域	特征点个数
眉毛	16
眼睛	16
鼻子	13
嘴部	22
脸部轮廓	21

2.2 个性化人脸三角网格生成

这一模块的基本思想是先设计出符合基本人脸形状和器官分布的标准三角网格，并通过定义每个三角形的顶点序号来得到网格点的相对位置与三角网格面片之间的拓扑关系；然后用人脸特征点提取算法得到的控制点坐标来对标准网格进行校准变形，以便实现不同人脸照片的个性化面部网格生成。

2.2.1 标准网格

标准人脸网格如图 4 所示，该人脸网格包含 218 个特征点，361 个三角面片。网格的设计主要基于人脸器官分布和肌肉运动方向，以便于表情器官动作的控制生成。如眉毛、眼睛和嘴部三角网格划分较细，而额头和脸颊部分网格则相对稀疏。网格点与面片的基本分布统计以及各个面部器官(区域)对应的网格点数和所涉及到的 FAP 如表 2 所示。

表 2 网格点分布与关联 FAP 表

Tab. 2 Mesh points distribution and related FAPs table

区域	网格点个数	相关联的 FAP 编号	FAP 个数
眼眶	20	19～22	4
眼球	0	23～28	6
眉毛	20	31～38	8
嘴部	34	4～13,16,17,51～60	22
鼻子	9	61～64	4
脸颊	55	18,39～42	5
下颚	45	3,14,15	3
额头	25	N/A	0

2.2.2 网格点生成

根据 2.1 节中的人脸特征点提取算法得到的网格控制点，即可对标准人脸网格进行校准变形，进而生成个性化的人脸网格。这些网格点的生成是根据不同的面部器官分区域进行的，具体的生成方法描

万方数据

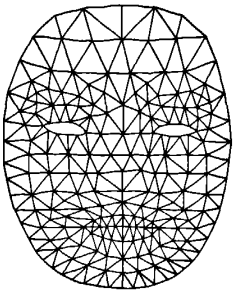

图 4　标准人脸网格

Fig. 4　Standard face mesh

述如下:

(1)眼睛轮廓

88 个特征点中,关于眼睛轮廓的有 16 个点,而在标准网格之中则需要对 20 个点进行校准定位。校准时,首先根据左眼角点、右眼角点和上边中点生成抛物线 5 和 7(见图 5);然后通过左眼角点、右眼角点和下边中点生成抛物线 6 和 8。最后在 4 条抛物线上的等水平距离取点即可获得所有 20 个点。

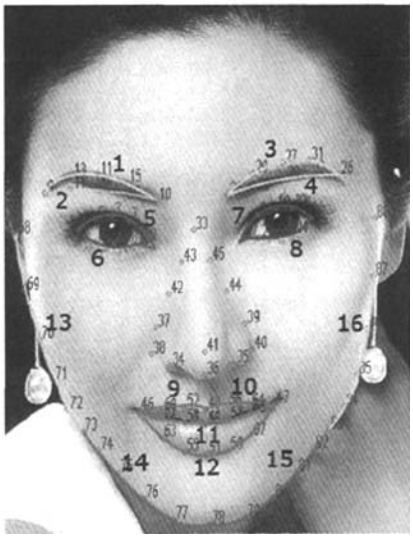

图 5　面部器官抛物线拟合

Fig. 5　Fitting of parabolas for face regions

(2)嘴部

88 个特征点中,嘴部轮廓有 22 个点,而在标准网格中则需要对 34 个点进行校准定位。通过对生成的抛物线 9 ~ 12 进行拟合,即可获取所有 34 个点。

(3)眉毛

88 个特征点中,眉毛区域有 16 个点,而标准网格中则需要对 20 个点进行校准定位。可通过对生成的抛物线 1、2、3 和 4 进行拟合来获取所有 20 个点。

(4)脸部轮廓

88 个特征点中有 21 个点表示脸部轮廓线,而在网格图中共有 33 个点表示轮廓线。可将轮廓线分为 4 段,分别用抛物线 13 ~ 16 进行拟合。

(5)额头

根据人脸结构的经验比例,从发际中点至眉心与眉心至下巴的比例大致为 1 ∶ 2。根据这个比例就可以算出发际中点的坐标。由于额头部分在人脸表情动作中起到的作用较少,因此额头部分的网格采用了等比例伸缩的方法进行近似生成。

(6)其他区域

如额头、脸颊、嘴部外围等处的点,它们的坐标可根据已经定好位置的网格点按比例算出。

最终生成的个性化人脸的三角网格如图 6 所示。由图 6 可以看到,对于不同的人脸照片,本文在标准网格的基础上,进行的个性化变形结果都能较为精确地与各个面部器官和脸部轮廓相匹配。

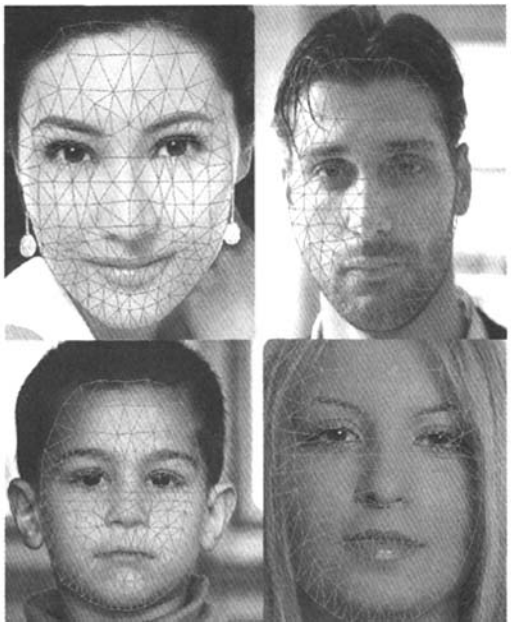

图 6　个性化的人脸网格生成

Fig. 6　Mesh morphing applied in different faces

同时,为了能够应用 FAP 生成表情动作,还计算了与个性化人脸网格对应的 FAPU 参数值。FAPU 的计算用到了 7 个特征点的坐标,这 7 个特

万方数据

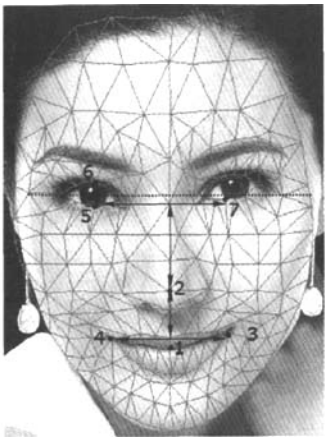

图 7 真实人脸网格 FAPU 定义

Fig. 7 Definition of FAPUs in real face mesh

征点的分布如图 7 所示，各 FAPU 参数值的具体计算公式如下：

$$\begin{cases} d_{\text{mouth-nose}} = y_1 - y_2 \\ w_{\text{mouth}} = x_3 - x_4 \\ D_{\text{IRIS}} = y_5 - y_6 \\ d_{\text{eye-nose}} = y_2 - (y_5 + y_6)/2 \\ d_{\text{eye}} = x_7 - x_5 \end{cases} \tag{1}$$

2.3 基于 FAP 的网格变形算法

2.3.1 网格点移动

在文献[1]中，每个 FAP 只定义了一个对应的 FDP 特征点在一个方向上的运动，但在实际应用时，为了取得自然的表情动作效果，特征点附近关联的其他的网格点需要根据特征点的运动进行相应的变形移动，才能实现对真实人脸肌肉运动的模拟。网格点变形算法的好坏将直接影响到人脸表情动作的真实感和自然度。针对与每一个 FAP 对应的 FDP 特征点，本文设置该特征点的影响范围为以该点为中心的矩形。同时将由 FDP 特征点运动引起的矩形内部网格点的关联位移分解为水平和垂直两个方向，然后分别计算两个方向上的关联位移。具体的位移坐标计算描述如下：

对 FAP 所对应的 FDP 特征点 p，不妨设其坐标为 (x_0, y_0)，设点 p 的影响范围由两对角顶点的坐标分别为 (x_1, y_1) 和 (x_2, y_2) 的矩形所确定。假设对应点 p 的某一方向运动的 FAP 为 F_1，则当点 p 做 s_1 个单位的 F_1 运动时，对于任意周围的网格点 $\hat{P} = (\hat{x}, \hat{y})$，其在该方向的移动距离 S 由式(2)确定：

$$S = \begin{cases} \dfrac{|\hat{y} - y_1| \cdot |\hat{x} - x_1| \cdot S_1}{|y_0 - y_1| \cdot |x_0 - x_1|} & (x_1 < \hat{x} < x_0 \text{ 和 } y_1 < \hat{y} < y_0) \\ \dfrac{|\hat{y} - y_2| \cdot |\hat{x} - x_1| \cdot S_1}{|y_0 - y_2| \cdot |x_0 - x_1|} & (x_1 < \hat{x} < x_0 \text{ 和 } y_0 \leqslant \hat{y} < y_1) \\ \dfrac{|\hat{y} - y_1| \cdot |\hat{x} - x_2| \cdot S_1}{|y_0 - y_1| \cdot |x_0 - x_2|} & (x_0 \leqslant \hat{x} < x_2 \text{ 和 } y_1 < \hat{y} < y_0) \\ \dfrac{|\hat{y} - y_2| \cdot |\hat{x} - x_2| \cdot S_1}{|y_0 - y_2| \cdot |x_0 - x_2|} & (x_0 \leqslant \hat{x} < x_2 \text{ 和 } y_0 \leqslant \hat{y} < y_1) \\ 0 & (\text{其他情况}) \end{cases} \tag{2}$$

上式的物理意义即特征点 p 与 FAP 对应的特征点的距离越近，则移动距离大，反之移动距离小。在影响范围的边界以及边界以外的区域，移动距离为 0。

当有多个 FAP 导致同一个网格点运动时，正如同多个力同时作用一样，只需将各个 FAP 造成的移动距离数值相叠加，即可算出该网格点最终需要移动的距离。

以 52 号 FAP(*raise_b_midlip_o*，即下外唇的中点向上垂直移动)为例。52 号 FAP 对应的特征点是外唇中点，也就是图 8 中标出的黑点；对应的单位是 MNS。下外唇中点的影响范围是以 ($x_{\text{左外唇点}}$, $y_{\text{左外唇点}}$) 和 ($x_{\text{右外唇点}}$, $y_{\text{下唇点}}$) 为两个对角顶点的矩形，如图 8 中标出的矩形方框。当有一个大小为 -1 单位的 52 号 FAP 输入时，下唇中点移动 -1 个单位(向下移动 1 个 MNS)，在影响范围中的其他网格点根据式(1)计算移动距离。如图 8 所示，箭头长度越长，表示移动距离越大，越短表示移动距离越小。

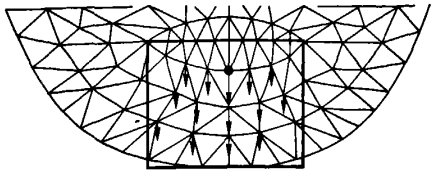

图 8 52 号 FAP 网格点移动示意图

Fig. 8 Mesh points movement driven by No. 52 FAP

2.3.2 网格拓扑结构的保持

网格变形的关键是保持整个网格的拓扑结构不变，具体到每个三角面片而言，就是要保持三角形中各个顶点与其对边的位置关系不变。如图 9 所示，直线 BC 将平面划分为两部分，经过移动后，顶点 A 不能变形到 BC 的另一半平面去，否则会与网格中其他三角形相互重叠，从而导致网格拓扑结构的

万方数据

变化。

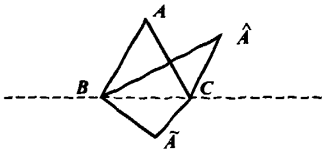

图 9　三角形变形示意图

Fig. 9　An example of triangle transform

因此,本文设定了三角形在变形过程中,各个顶点移动的最大距离,只要顶点移动距离不超过该最大值,就可以保证拓扑结构不变。本文借用向量积来表示这一条件。若三角形的 3 个顶点分别为 $\boldsymbol{A}=(x_A,y_A)$、$\boldsymbol{B}=(x_B,y_B)$、$\boldsymbol{C}=(x_C,y_C)$,则有

$$\begin{aligned}\boldsymbol{AC}\times\boldsymbol{BC}&=|\boldsymbol{AC}||\boldsymbol{BC}|\sin\angle ACB\\&=(x_C-x_A)(y_C-y_B)-(y_C-y_A)(x_C-x_B)\end{aligned}\tag{3}$$

若要保证点 $\boldsymbol{A}$ 与边 BC 位置不变,则需要使式(3)中,值的符号不变。因此,假设变形后三角形的 3 个顶点分别为 $\hat{\boldsymbol{A}}=(x_{\hat{A}},y_{\hat{A}})$、$\hat{\boldsymbol{B}}=(x_{\hat{B}},y_{\hat{B}})$、$\hat{\boldsymbol{C}}=(x_{\hat{C}},y_{\hat{C}})$,若要保持三角形拓扑不变,则要满足以下 3 个不等式:

$$\begin{cases}((x_C-x_A)(y_C-y_B)-(y_C-y_A)(x_C-x_B))\times\\\quad((x_{\hat{C}}-x_{\hat{A}})(y_{\hat{C}}-y_{\hat{B}})-(y_{\hat{C}}-y_{\hat{A}})(x_{\hat{C}}-x_{\hat{B}}))>0\\((x_B-x_A)(y_B-y_C)-(y_B-y_A)(x_B-x_C))\times\\\quad((x_{\hat{B}}-x_{\hat{A}})(y_{\hat{B}}-y_{\hat{C}})-(y_{\hat{B}}-y_{\hat{A}})(x_{\hat{B}}-x_{\hat{C}}))>0\\((x_A-x_B)(y_A-y_C)-(y_A-y_B)(x_A-x_C))\times\\\quad((x_{\hat{A}}-x_{\hat{B}})(y_{\hat{A}}-y_{\hat{C}})-(y_{\hat{A}}-y_{\hat{B}})(x_{\hat{A}}-x_{\hat{C}}))>0\end{cases}\tag{4}$$

2.3.3　面部纹理填充

表情动作生成是通过面部网格点的移动来引起网格三角面片的形变实现的,与此同时,与网格面片对应的面部纹理像素也会相应移动变化。本文采用仿射变换,通过对网格三角形内的像素进行坐标轴变换处理来保证图像纹理的自然过渡。仿射变换的目标是实现从 2 维坐标到 2 维坐标之间的线性变换,且保持 2 维图形的"平直性"和"平行性"。仿射变换可以通过一系列的源自变换的复合来实现,包括平移、缩放、翻转、旋转和剪切。

这类变换可以用以下一个 3×3 的矩阵来表示:

$$\begin{bmatrix}\hat{x}\\\hat{y}\\1\end{bmatrix}=\begin{bmatrix}m_{0,0}&m_{0,1}&m_{0,2}\\m_{1,0}&m_{1,1}&m_{1,2}\\0&0&1\end{bmatrix}\begin{bmatrix}x\\y\\1\end{bmatrix}=\begin{bmatrix}m_{0,0}x+m_{0,1}y+m_{0,2}\\m_{1,0}x+m_{1,1}y+m_{1,2}\\1\end{bmatrix}\tag{5}$$

该变换矩阵用于将原坐标(x,y)变换为新坐标$(\hat{x},\hat{y})$。根据式(5)就可以先得到仿射变换的系数 $m_{i,j}(i=0,1;j=0,1,2)$,然后遍历新三角形区域所有点,并用与其对应的原坐标点的像素进行填充,进而实现网格变形后人脸图像纹理的自然填充。

3　表情纹理细节处理

3.1　口内贴图

人们在表现言语口型动作和某些表情动作(如大笑、惊讶等)时,嘴部区域常常是不闭合的,会露出口腔内的舌头牙齿等器官,为了取得自然真实的表情动作效果,就需要对 2 维人脸照片进行口内贴图,以满足真实性。而且为了在不同的人脸模型上都能够取得较好的匹配效果,需要有适应性好的贴图方法。

进行口内贴图时,首先为口腔内的舌、牙等器官制作了 3 张纹理图片(如图 10 所示)。

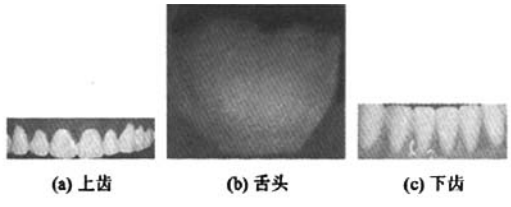

图 10　原始口内贴图

Fig. 10　Original texture mapping inside the mouth

在生成人脸变形图像时,先设置舌头图片的深度为 3(相当于置于底层),然后分别放置下牙和上牙图片的深度为 2 和 1,最后设置变形后的人脸图像深度为 0,以便实现各个器官贴图的正确覆盖关系。此时,口内空洞区域则正好被牙齿和舌头填充。

然而,由于需要处理不同的人脸照片,因此如果都采用同样的牙齿、舌头模板,那么就会导致如下两个问题:(1)不同的人脸有不同的尺寸,其对应的牙齿也应该有不同的尺寸,如果直接应用同一套的贴图模板,则会产生明显的不协调感;(2)由于正常人在说话或做表情动作时,上下齿的位置不是固定不变的,而是跟随口型动作实时移动的.因此在进行脸

万方数据

像网格变形的同时，必须同时对牙齿进行同步的移动，否则会出现牙齿错位的现象。

3.1.1 口内贴图尺寸调整

为了解决不同人脸的口内贴图的匹配问题，可首先根据个性化人脸网格中的 FAPU 标准参数对牙齿和舌头图片进行缩放。本文采用了嘴和鼻尖间距(MNS)和嘴宽度(MW)参数分别对口内贴图的高度和宽度进行调整。缩放比例为

$$\begin{cases} H_{\text{teeth}} = d_{\text{mouth-nose}}/2 \\ W_{\text{tongue}} = W_{\text{mouth}} \end{cases} \tag{6}$$

式中，$d_{\text{mouth-nose}}$ 是嘴与鼻尖间的距离，W_{mouth} 是嘴的宽度。

笔者之前曾尝试将牙齿宽度与嘴宽度进行关联，但由于嘴角位置可变性太大，致使同一个人脸在不同表情图片中，MW 可能会有很大差异。而 MNS 则相对稳定，因此本文把牙齿高度 H_{teeth} 和 MNS 相关联。

3.1.2 口内贴图位置动态调整

为了实现口腔器官随面部网格变形同步移动的效果，本文还设计了口内贴图位置动态调整的算法。根据人体生理结构，在表情动作过程中，由于上齿相对位置固定，因此可选取面部网格中，距离上齿位置较近，且位置基本保持固定的鼻尖点(FDP 9.3)作为上齿的参考控制点，并以 MNS 作为单位。设向下为 Y 轴正方向，上齿最上端的 Y 坐标为 y_{ut}，鼻尖点的 Y 坐标为 y_{nose}。上齿位置的计算公式如下：

$$y_{\text{ut}} = y_{\text{nose}} + 0.25 \times d_{\text{mouth-nose}} \tag{7}$$

而下齿的位置变化则比较灵活，但考虑到它的移动和下颚骨骼相关，因此可选取下颚点作为其参考控制点。下齿最下端的 Y 坐标为 y_{lt}，下颚点的 Y 坐标为 y_{jaw}，下内中唇点的 Y 坐标为 y_{midlip}。下齿位置的计算公式如下：

$$\begin{aligned} y_{\text{lt}} &= y_{\text{jaw}} - 0.75 \times (y_{\text{jaw}} - y_{\text{midlip}}) \\ &= 0.25 \times y_{\text{jaw}} + 0.75 \times y_{\text{midlip}} \end{aligned} \tag{8}$$

在水平方向上，上下齿的中点应分别与其控制点的水平坐标相同。

图 11 是不同口型下生成的人脸效果。

图 11 口内贴图效果

Fig. 11 The illustration of texture mapping inside the mouth

3.2 眼神表情

眼神是人们表达情感的重要手段。虽然 MPEG-4 中也定义了眼球移动的 FAP，但是在 2 维人脸图像中，眼球 FAP 的实现与其他 FAP 相比，有较大的不同之处。这是因为眼球的形状在移动的过程中是保持固定的，而不是像其他器官那样可以发生形变。如果将眼睛区域也简单分成三角网格，而且单纯采用前文所述的网格变形和仿射变换方法的话，那么移动后的眼球就会变形，不再是圆形的眼珠；而如果只将眼睛内部的所有像素进行整体移动的话，则又会造成眼角边缘的过渡不自然和空洞现象。本文采用的方法是以上两种方法的结合，简而言之就是对与眼球对应的像素进行整体平移，而对与“眼白”对应的像素则进行拉伸变形。

如图 12 所示，眼睛内部的是没有网格的，而眼眶的上下轮廓是靠 2.2.2 节中提到的算法获得的。笔者同时观察到，当眼睛正视前方时，由左上眼眶五分之一点(点 a)、左下眼眶五分之一点(点 b)与左眼角所组成的三角形和由右上眼眶五分之一点(点 c)、右下眼眶五分之一点(点 d)与右眼角组成的三角形是基本不包含眼球像素的。

图 12 眼部区域细节

Fig. 12 Detail of eye area

当眼珠向右移动时，将点 a 和点 b 连线的右侧部分整体向右移动一个单位，而将该连线以左的部分则根据点 a 和点 b 的位移进行关联移动，具体的位移计算按 2.3.1～2.3.3 节所述算法进行移动。当眼珠向左移动时，点 c 和点 d 的连线就成为分割线，分割线左面的点整体移动，而分割线右面的点则按 2.3.1～2.3.3 节所述算法根据点 c 和点 d 进行关联移动。

眼神移动的生成效果如图 13 所示，同时生成了

万方数据

与抛媚眼和蔑视对应的表情动作,可见其中眼神的处理为表情生成增色不少。

假设原始照片的眼睛没有正视的话,则无法采用五分之一点这样的近似方法。因此,笔者对算法进行了改进,即采用 Canny 边缘检测方法来获得眼球的左右边缘点(如图 14 所示),以取代上述算法中的五分之一点。

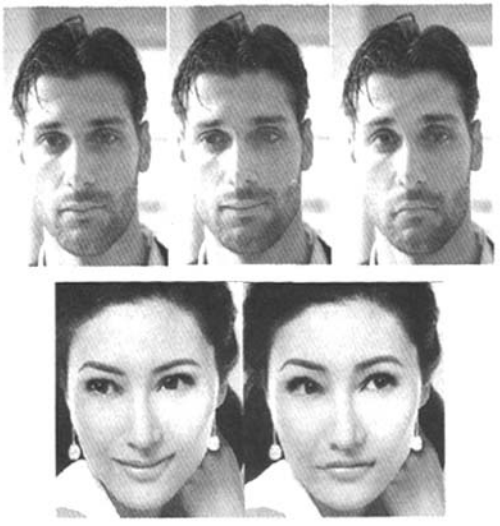

图 13　有眼球移动参与的表情效果

Fig. 13　The illustration of eyeballs movement

图 14　眼球 Canny 边缘检测效果

Fig. 14　The result of Canny edge detection

4　实验结果

本文在上述 2 维表情脸像生成框架的基础上,进行了多种实验,包括表情脸像合成、虚拟说话人动画生成等。

4.1　表情脸像合成

4.1.1　表情 FAP 参数确定

高兴、悲伤、生气、害怕、厌恶、惊讶 6 种基本感情都需要结合多种 FAP 动作进行表现。文献[4]给出了这 6 种感情的 FAP 对应表(如表 3 所示)。

本文基本采用了文献[4]的分类方法,并进行了简化,设计了简化的表情动作对照表,同时用手工设定了 FAP 动画规则。

表 3　表情动作对照表

Tab. 3　Expression-Movement mapping table

表情	动作描述
高兴	嘴角上扬,下唇往下,下巴向下,眉稍上扬
悲伤	嘴角向下,嘴唇向外拉,眉毛向下
惊讶	上唇向上,下唇往下,眉毛整体向上
生气	嘴唇紧闭内缩,眉稍向上,眉间紧蹙
害怕	下唇微张,嘴角向下,眉稍向下

图 15 是对几组不同年龄、不同性别、不同种族的人脸合成图片效果,其中,从左到右、从上到下分别为原始照片以及高兴、悲伤、害怕、生气和惊讶等不同表情的图像。

4.1.2　主观评测实验

为了验证提出的 2 维人脸图像变形算法在生成人脸表情方面的效果,本文还进行了主观评测实验,对 FAP 驱动生成的 2 维人脸表情脸像的自然度和表情表达的一致性进行了评测。

本文以调查问卷的形式对由 4 个不同的人脸图像生成的表情脸像进行了评测。每个人脸图像生成 5 种表情,分别是"高兴"、"悲伤"、"愤怒"、"害怕"和"惊讶",共计 20 张表情图片。被试者需要对每张表情图片进行"贴标签"式分类,即从如下 6 个表

表 4　表情脸像评测总表

Tab. 4　Evaluation result of synthetic facial expression image

表情脸相	正确率(%)	平均自然度(5 分评分制)
Male1-高兴	100	3.9
Male1-悲伤	60	4
Male1-生气	40	4
Male1-害怕	40	3.4
Male1-惊讶	100	3.6
Female2-高兴	100	4.2
Female2-悲伤	40	3.6
Female2-生气	90	3.3
Female2-害怕	50	3.5
Female2-惊讶	60	3.7
Female3-高兴	100	3.9
Female3-悲伤	70	3.9
Female3-生气	60	3.3
Female3-害怕	30	2.9
Female3-惊讶	100	3.4
Child4-高兴	100	3.4
Child4-悲伤	70	4
Child4-生气	70	3.9
Child4-害怕	30	3.7
Child4-惊讶	90	3.8

万方数据

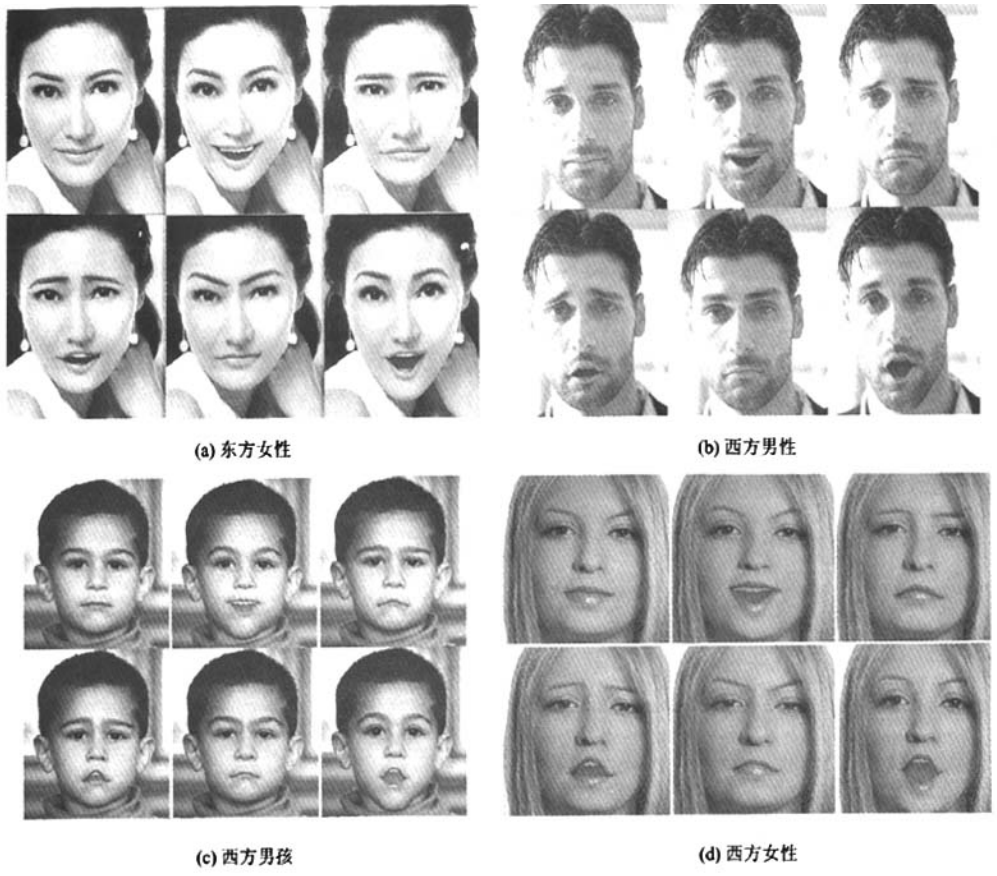

图 15 在不同人脸模型的表情生成效果

Fig. 15 The illustration of synthetic facial expression on different face models

情关键词中选取最为恰当的一个用来描述当前表情图片所表现出的情感,这 6 个表情关键词分别是“高兴”、“悲伤”、“生气”、“害怕”、“惊讶”、“无法判断”。同时,评测者还要给出每张表情脸像的自然度,用 1 ~5 分来进行量化打分,分别是“1 分:极不自然”、“2 分:较不自然”、“3 分:可以接受”、“4 分:比较自然”、“5 分:非常逼真”。

表 5 表情脸像评测结果(按表情分类)

Tab. 5 Evaluation results (classified by expression)

表情类别	正确率(%)	平均自然度
高兴	100	3.85
悲伤	60	3.85
生气	65	3.625
害怕	37.5	3.375
惊讶	87.5	3.625

表 6 表情脸像评测结果(按人脸分类)

Tab. 6 Evaluation results (classified by face)

人脸模型	正确率(%)	平均自然度
Male-1	68	3.76
Female-2	68	3.66
Female-3	72	3.48
Child-4	72	3.76

实验时,共有 10 名被试者参与评测。统计结果见表 4、表 5、表 6。从上面的实验结果可以得到如下结论:

(1)整体表情脸像的平均自然度较高,在 3.67 左右,90% 的脸像评价都等于或大于“3 分:可以接受”,其中 23.5% 为“5 分:非常逼真”。而且不同表情和不同人脸模型的平均自然度都在 3 ~4 之间,没有显著差别。

万方数据

(2)各表情的评测正确率的差别比较大,高兴的评测正确率为 100%,惊讶为 87.5%,生气为 65%,悲伤为 60%,而害怕只有 37.5%。

4.2 虚拟说话人

本文将 2 维人脸网格变形算法集成在虚拟说话人的动画合成系统中,利用口型动画的 FAP 驱动 2 维人脸图片开口说话。虚拟时,先通过读入口型动作的 FAP 序列文件,然后按照给定的速率显示图片,就可以生成自然的虚拟说话人效果。实验证明,对于 25 fps(即每帧 40 ms)的播放速率,就可以很好地实现音视频同步。

在普通的 PC 机器上(CPU2GHz,内存 1 G),实验证明对 431×591 大小的图片,平均每个表情或口型的运算时间只需要 10~20 ms,对于动作幅度较小的表情,甚至在 10 ms 以下,这是因为与动作幅度小的表情对应的网格变形涉及到的网格点的数目较少。实验证明,本文提出的 2 维人脸网格变形算法能够应用在实时的视频流处理和实时的动画生成领域。

5 结　论

本文提出了一种基于 MPEG-4 人脸动画框架的 2 维人脸表情图像的实时生成方法。该方法实现了个性化的人脸网格的生成,同时提出了 FAP 驱动的网格移动变形算法,还采用仿射变换实现面部纹理填充,并且在眼球移动和口内贴图等表情纹理细节处理上进行了创新。主观评测实验证明,该算法不仅在不同的人脸图像上都能取得真实、自然的效果,而且该算法实时性好,能够应用于实时的视频处理和动画生成。

同时,主观评测实验的结果也显示出一些问题,例如各表情的评测正确率的差别比较大,"害怕"等表情的识别率非常低。这一结果一方面表明,对表情的 FAP 定义还需要进一步调整;另一方面也表明,对于"害怕"这类表情进行参数量化还比较困难,因为这类表情没有显著的共同特征,不同人做不同表情会有不同效果。另外,不同人看到的同一面部表情也会有不同的理解。个性化表达是表情生成的难点之一,也是今后工作的改进方向。

致　谢　文中所用到的人脸图片来自 flickr.com。

参考文献(References)

1 Murat T, Ostermann J. Face and 2-D mesh animation in MPEG-4 [J]. Signal Processing: Image Communication, 2000,15: 387-342.

2 Zhang Shen, Wu Zhi-yong, Meng Helen M, *et al*. Facial expression synthesis using PAD emotional parameters for a Chinese expressive avatar[A]. In: Proceedings of Affective Computing and Intelligent Interaction [C], Lisbon, Portugal, 2007: 24-35.

3 Zhang Li, Ai Hai-zhou, Xin Sheng-jun, *et al*. Robust face alignment based on local texture classifiers[A]. In: Proceedings of the IEEE International Conference on Image Processing [C], Genoa, Italy, 2005: 354-357.

4 Raouzaion N, Tsapatsoulis N, Karpouzis K, *et al*. Parameterized Facial Expression Synthesis Based on MPEG-4 [J]. EURASIP Journal on Applied Signal Processing, 2002,10: 1021-1038.

万方数据