

清华 大学

# 综合 论文 训 练

题目：人体三维姿态估计算法研究

系 别：自动化系

专 业：自动化

姓 名：梁鼎

指导教师：刘烨斌副研究员

2013年6月7日

# 关于学位论文使用授权的说明

本人完全了解清华大学有关保留、使用学位论文的规定，即：学校有权保留学位论文的复印件，允许该论文被查阅和借阅；学校可以公布该论文的全部或部分内容，可以采用影印、缩印或其他复制手段保存该论文。

(涉密的学位论文在解密后应遵守此规定)

签 名：\_\_\_\_\_ 导师签名：\_\_\_\_\_ 日 期：\_\_\_\_\_

## 中文摘要

人体姿态估计一直以来是计算机视觉研究领域的重点研究方向之一。人体姿态估计在体育运动分析、人机交互等领域有着广阔的应用前景和重大的应用价值。如今的研究主要集中于从单目图像中估计二维人体姿态信息，存在的问题有准确率低、无法完整刻画人体的运动姿态等。本文旨在通过从三幅不同角度拍摄的照片中估计出人体三维骨架信息。

本文的创新点有：

- 提出了一种有效的背景分割算法。原有的分割算法会产生很多空洞及毛边，这在HoG描述特征中会产生很大噪声，本文提出了一种有效的图像分割方法，使人物能从背景中完整分离；
- 实现了一种HoG描述特征。原有的HoG特征维数过多，且会根据人物在图像中的尺寸改变每个元胞的大小，本文通过固定长宽比，保持了原有图像的信息；
- 提出了一种有效的含有皮肤信息的特征。新增的皮肤信息对姿态识别有重要帮助。

综上所述，本文提出实现了一种新的图像特征描述，该特征能有效地应用于人体三维姿态估计中，尤其适合识别非数据库中人物的姿态，增强了鲁棒性。

**关键词：**三维，姿态，估计

## ABSTRACT

**Keywords:** 3D; pose; estimation

# 目 录

|                              |           |
|------------------------------|-----------|
| <b>第1章 绪论 .....</b>          | <b>1</b>  |
| 1.1 研究背景 .....               | 1         |
| 1.2 研究现状 .....               | 1         |
| 1.2.1 综述 .....               | 1         |
| 1.2.2 二维姿态估计算法 .....         | 2         |
| 1.2.2.1 基于生成模型的姿态估计算法 .....  | 2         |
| 1.2.2.2 基于判别模型的姿态估计算法 .....  | 2         |
| 1.2.2.3 基于肢体的姿态估计算法 .....    | 2         |
| 1.2.2.4 其他算法 .....           | 3         |
| 1.2.3 三维姿态估计算法 .....         | 3         |
| 1.2.3.1 基于特征的姿态估计算法 .....    | 3         |
| 1.2.3.2 基于二维姿态估计的算法 .....    | 3         |
| 1.2.3.3 半自动姿态估计算法 .....      | 3         |
| 1.2.4 研究难点 .....             | 3         |
| 1.3 问题描述 .....               | 4         |
| 1.4 文章结构 .....               | 4         |
| <b>第2章 构建三维人体姿态数据库 .....</b> | <b>6</b>  |
| 2.1 HumanEva数据库 .....        | 6         |
| 2.1.1 数据来源 .....             | 6         |
| 2.1.2 数据统计 .....             | 8         |
| 2.2 人体三维骨架表示 .....           | 10        |
| <b>第3章 提取特征 .....</b>        | <b>12</b> |
| 3.1 图片预处理 .....              | 12        |
| 3.1.1 计算背景数据 .....           | 13        |
| 3.1.2 局部优化 .....             | 13        |

|                                 |           |
|---------------------------------|-----------|
| 3.2 HOG特征 .....                 | 17        |
| 3.2.1 概述 .....                  | 17        |
| 3.2.2 计算HOG描述子 .....            | 17        |
| 3.3 皮肤特征 .....                  | 21        |
| 3.4 特征表示 .....                  | 21        |
| <b>第4章 基于双高斯过程的姿态估计算法 .....</b> | <b>22</b> |
| 4.1 概述 .....                    | 22        |
| 4.2 高斯过程回归 .....                | 22        |
| 4.3 双高斯过程回归 .....               | 23        |
| 4.3.1 概述 .....                  | 23        |
| 4.3.2 KL散度 .....                | 24        |
| 4.3.3 求解优化问题 .....              | 24        |
| 4.4 TGPKN .....                 | 25        |
| <b>第5章 结果比较与分析 .....</b>        | <b>26</b> |
| 5.1 运行环境 .....                  | 26        |
| 5.2 测试集 .....                   | 26        |
| 5.3 误差度量 .....                  | 27        |
| 5.4 测试结果 .....                  | 28        |
| 5.4.1 HumanEva测试集 .....         | 28        |
| 5.4.2 其他数据 .....                | 28        |
| 5.5 计算时间 .....                  | 32        |
| 5.6 结果分析 .....                  | 32        |
| <b>第6章 总结与展望 .....</b>          | <b>33</b> |
| 6.1 总结 .....                    | 33        |
| 6.2 收获与体会 .....                 | 34        |
| <b>插图索引 .....</b>               | <b>35</b> |
| <b>表格索引 .....</b>               | <b>36</b> |
| <b>公式索引 .....</b>               | <b>37</b> |

|                  |    |
|------------------|----|
| 致 谢 .....        | 38 |
| 声 明 .....        | 39 |
| 附录A 外文资料翻译 ..... | 40 |

## 主要符号对照表

|            |   |
|------------|---|
| HPC        | 高性能计算(High Performance Computing)   |
| cluster    | 集群  |
| Itanium    | 安腾  |
| SMP        | 对称多处理   |
| API        | 应用程序编程接口  |
| PI         | 聚酰亚胺  |
| MPI        | 聚酰亚胺模型化合物, N-苯基邻苯酰亚胺  |
| PBI        | 聚苯并咪唑   |
| MPBI       | 聚苯并咪唑模型化合物, N-苯基苯并咪唑  |
| PY         | 聚吡咯   |
| PMDA-BDA   | 均苯四酸二酐与联苯四胺合成的聚吡咯薄膜   |
| $\Delta G$ | 活化自由能 (Activation Free Energy)  |
| $\chi$     | 传输系数 (Transmission Coefficient)   |
| $E$        | 能量  |
| $m$        | 质量  |
| $c$        | 光速  |
| $P$        | 概率  |
| $T$        | 时间  |
| $v$        | 速度  |
| 劝学         | 君子曰：学不可以已。青，取之于蓝，而青于蓝；冰，水为之，而寒于水。木直中绳。（车柔）以为轮，其曲中规。虽有槁暴，不复挺者，（车柔）使之然也。故木受绳则直，金就砺则利，君子博学而日参省乎己，则知明而行无过矣。吾尝终日而思矣，不如须臾之所学也；吾尝（足齐）而望矣，不如登高之博见也。登高而招，臂非加长也，而见者远；顺风而呼，声非加疾也，而闻者彰。假舆马者，非利足也，而致千里；假舟楫者，非能水也，而绝江河，君子生非异也，善假于物也。积土成山，风雨兴焉；积水成渊，蛟龙生焉；积善成德，而神明自得，圣心 |

备焉。故不积跬步，无以至千里；不积小流，无以成江海。骐  
骥一跃，不能十步；驽马十驾，功在不舍。锲而舍之，朽木不  
折；锲而不舍，金石可镂。蚓无爪牙之利，筋骨之强，上食埃  
土，下饮黄泉，用心一也。蟹六跪而二螯，非蛇鳝之穴无可寄  
托者，用心躁也。——荀况

# 第1章 绪论

本章首先阐述了本文研究问题提出的背景，并总结了前人的工作以及该研究方向的难点和重点，以此提出了自己研究的具体任务，最后简要介绍了本文的内容框架。

## 1.1 研究背景

过去的十到二十年以来，在计算机视觉领域很大比例的文章都与人物相关，包括人体检测与跟踪、人脸识别、人体姿态或动作的识别等等。前两个研究方向相对较为成熟，现在的算法已经可以很容易做到实时同时正确率超过90%，然而人体姿态的识别却一直未能攻克。人们在图像中表达的信息很大程度上是由其姿态（即动作）决定的，因此人体姿态识别具有很广阔的应用前景，在人机交互、影视动画、临床运动诊断、体育运动动作分析等方面都有重要应用价值。例如微软公司设计的kinect传感器就可以实时捕捉识别人体的三维姿态，成功地应用于人机交互游戏中。

目前，最为流行且实用的人体姿态捕捉方法是利用多台摄像机、在目标人物身上附加标记点或传感器，这种方法在电影特效、游戏动画等领域应用十分普遍，被公认为是最有效的方法。这种方法的最大优点就是姿态能够准确获得，但它的问题有很多，使得该方法的应用受到了限制。基于标记点的方法对硬件设备依赖较大，标记点处理复杂，传感器价格昂贵，且无法对已有的图像或视频进行分析。因此，目前的研究都是为了摆脱标记点，用人工智能的方法自动检测识别人体姿态。本文也顺应该研究方向，力求提高识别精度。

## 1.2 研究现状

### 1.2.1 综述

人体姿态识别的研究内容有很多种分类方式，首先我们根据姿态描述信息的多少分为姿态估计和动作分类两种，动作分类的任务是给定一张静止图像或图像序列，判定这张图片中人物的动作类比，例如wang等人<sup>[?]</sup>只需要分辨图像

中人的动作是拳击、跑步或鼓掌等动作即可，而姿态估计则需要给出人物的关键点的具体位置来描述姿态，包含了更丰富的信息，以下将着重介绍姿态估计的相关研究。还有一种分类方式是依据是否需要人工参与分为全自动和半自动估计，wei等人<sup>[?][?]</sup>的工作就是在人工标定一些关键帧后由算法估计出其他视频帧中的任务姿态，以下的介绍将集中于全自动估计方法。此外还可以根据使用的图像视角的数量分为单目识别和多目识别，在二维姿态估计中只需要单目图像信息，三维姿态估计则通常需要多目信息，如<sup>[?][?][?][?]</sup>，但也有单目估计的算法，如<sup>[?][?][?]</sup>。此外还可以分为单人和多人识别、二维和三维估计等。

## 1.2.2 二维姿态估计算法

### 1.2.2.1 基于生成模型的姿态估计算法

理想的生成模型应该能考虑到姿态、衣着的多样性，从而能生成更真实的样本，然而由于复杂度限制，这是不可能实现的。因此，人们做了很多假设，将场景限定在一定的范围内，缩小了样本空间。这类方法事先通过统计等方法给出一个人体姿态分布的先验函数，然后用似然函数描述观察到的图片所含姿态与先验函数姿态的一致性。这种方法的效果好坏很大程度上依赖于先验函数的优劣。

### 1.2.2.2 基于判别模型的姿态估计算法

判别模型方法不是事先给定先验函数，而是直接学习后验分布，这种方法与生成模型相比更加快捷、易于实现<sup>[?]</sup>。

### 1.2.2.3 基于肢体的姿态估计算法

这种方法将身体分为各个部分，所有的肢体（如头、胳膊、腿等）构成一个集合，因此对完整姿态的估计从一个高维模型化简为若干个低维的推断问题，同样，为了使数学推断更行之有效，<sup>[?][?][?]</sup>做了一些简化和假设。多数基于肢体的方法本质上也是生成模型，但通常也包含了判别模型<sup>[?]</sup>。<sup>[?]</sup>取得了非常好的结果。

#### 1.2.2.4 其他算法

Wang等人<sup>[?]</sup>使用了一种隐含树模型，通过学习各肢体间的关系，构建出一个树结构的图模型，在LSP<sup>[?]</sup>和PARSE<sup>[?]</sup>数据集上得到了目前最优秀的结果。Tian等人<sup>[?]</sup>也使用了一种类似的树形结构，将人体分为三层，并用离散的类别刻画局部动作，取得了几乎是最好的结果。

### 1.2.3 三维姿态估计算法

#### 1.2.3.1 基于特征的姿态估计算法

基于特征的方法本质上是基于判别模型的方法<sup>[?][?][?][?]</sup>，输入图像被抽象成特征描述，经过学习得到模型，从而进行预测。代表性的几种方法有：Sminchisescu等人的SSLVM<sup>[?]</sup>、条件混合专家预测<sup>[?]</sup>、双高斯过程<sup>[?]</sup>。

#### 1.2.3.2 基于二维姿态估计的算法

这种算法<sup>[?]</sup>不关心二维姿态的估计，算法假定多视角的二维姿态已知，通过数据融合，推算出三维姿态，其强调的是如何将已有的二维姿态融合推广至三维情况。

#### 1.2.3.3 半自动姿态估计算法

该算法<sup>[?][?]</sup>已在前文介绍，不再赘述。

### 1.2.4 研究难点

相关研究之所以未能取得很大突破，究其原因主要有如下几点，这些因素被公认为人体姿态估计领域的重要挑战。

- (1) 图像中人物外貌、衣着的多样性
- (2) 光照条件的多样性
- (3) 遮挡问题，包括自身肢体遮挡、他人或场景中物品对人物的遮挡
- (4) 人体骨架的复杂性
- (5) 人体姿态是一个高维描述
- (6) 图片中丢失了3D信息
- (7) 复杂的背景环境

其中(6)在多目估计中不存在，因此多目图像估计能够更好地克服遮挡等问题。

### 1.3 问题描述

本文将研究内容聚焦于人体三维姿态的估计，研究工作建立在Sminchisescu等人的双高斯过程<sup>[?]</sup>工作基础之上，具体说来，输入与输出如下：

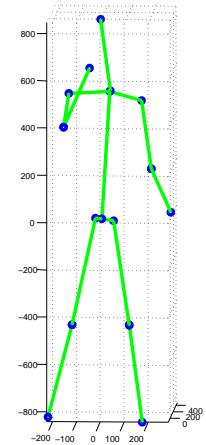
**输入** 无标记的单目或多目静止图像（可以推广至图像序列）

**输出** 用关节点表示的三维人体骨架（关于骨架表示法将在2.2节阐述）

可以参照图1.1。



(a) 输入



(b) 输出

图 1.1 本文研究问题的输入与输出

### 1.4 文章结构

本文共分为六章，第二章将介绍本人研究工作所依赖的数据库，包括数据库的内容、统计信息以及如何利用数据库等，其次介绍了本文所用的人体骨架模型的表示方法。第三章将介绍特征提取的一些方法，以及本文采用的特征以及其表示方法。第四章详细介绍了姿态估计算法，该算法基于双高斯过程，较好地考虑了输入输出变量之间的耦合关系。第五章主要以图表的方式整理了所做工作的结果，通过与前人工作对比，证明了本文工作的意义，并从结果中发

现总结问题，分析可能的原因。第六章对全文做以总结，最后针对还存在的问题提出一些可能的解决方法和未来的努力方向。

## 第2章 构建三维人体姿态数据库

由于人体姿态估计算法大部分为监督学习，因此研究者制作了很多用于训练和测试的数据库，这些数据库有些是从互联网、电影中获得的图片，并人工标记出骨架作为ground-truth，有些是通过标记点或传感器等手段在拍摄影像的同时记录人体姿态的ground-truth。表2.1整理了目前常用的人体数据库，本文采用了HumanEva-I<sup>[?]</sup>数据库，将在2.1节介绍。

### 2.1 HumanEva数据库

由于本文所做工作为人体三维姿态估计，不仅需要多个视角的视频数据，还需要对人物进行运动捕捉，记录关节点的位置，这项工作十分繁琐，由于经历和时间有限，也为了能和已有算法进行更好对比，最后选择了HumanEva数据库作为训练和测试的依据。

#### 2.1.1 数据来源

尽管近年来姿态估计的算法百花齐放，但是由于不同算法基于不同的数据库，度量标准也有所差别，而且很多数据只是二维图片和骨架，缺少三维骨架信息，因此算法能利用的数据库十分有限，且不同算法之间很难比较，为了解决如上问题，美国布朗大学于2006年在本田研究所和英特尔公司的支持下设计制作了HumanEva数据库。该数据库用多台相机同步记录了多视点的人体动作视频，并用运动捕捉的方法记录了三维姿态信息，同时各相机也进行了标定，因此也可以计算得到二维姿态信息，在给出数据本身的同时，项目组还为姿态估计领域的研究者提供了一个简单的示例程序，演示了如何读取数据库，此外项目组还给出了一套计算误差的标准和网络提交平台，供研究者上传自己的算法结果，与他人结果进行对比。因此，该数据库的出现，不仅让研究者有了易用的大量数据，而且促进了该领域的交流与发展。

表 2.1 常用姿态估计数据库汇总

|             | 数据集                                       | 数据格式   | 说明               | 误差度量             | 帧数                   | 复杂度      |
|-------------|---|--------|------------------|------------------|----------------------|----------|
| 维<br>一<br>单 | Agarwal/Triggs <sup>[?]</sup>             | 图片     | 3D运动捕捉           | AJA <sup>①</sup> | 训练: 1927<br>测试: 418  | 简单       |
|             | Buffy Stickmen <sup>[?]</sup>             | 图片     | 2D上身姿势           | PCP <sup>②</sup> | 748                  | 日常       |
|             | ETHZ PASCAL Stickmen <sup>[?]</sup>       | 图片     | 2D上身姿势           | PCP              | 549                  | 日常       |
|             | UIUC Stickmen <sup>[?]</sup>              | 图片     | 2D上身姿势           | PCP              | 训练: 346<br>测试: 247   | 日常       |
|             | PARSE <sup>[?]</sup>                      | 图片     | 2D完整姿势           | PCP              | 训练: 100<br>测试: 205   | 复杂       |
|             | Leeds Sports Poses Dataset <sup>[?]</sup> | 图片     | 2D完整姿势           | PCP              | 训练: 1000<br>测试: 1000 | 复杂       |
|             | Human-Object Interaction <sup>[?]</sup>   | 图片     | 2D完整姿势<br>2D运动   | PCP              | 训练: 180<br>测试: 120   | 复杂       |
| 维<br>三<br>多 | HumanEva <sup>[?]</sup>                   | 多目视频   | 三维运动捕捉           | AJP <sup>③</sup> | 约80000               | 日常       |
|             | MPI08 <sup>[?]</sup>                      | 多目视频   | 内部传感器<br>三维激光扫描  | AJA              | 约24000               | 日常<br>复杂 |
|             | Stanford ToF <sup>[?]</sup>               | ToF 深度 | 三维运动捕捉<br>三维激光扫描 | AMP <sup>④</sup> | 27 个序列               | 日常<br>复杂 |

① Average Joint Angel Error<sup>[?][?]</sup>② Percentage of Correctly Estimated Body Parts<sup>[?][?][?][?][?][?]</sup>③ Average Joint Position Error<sup>[?][?][?]</sup>

④ Average Marker Position Error

### 2.1.2 数据统计

HumanEva数据库一共包含两部分，分别为HumanEva-I和HumanEva-II，I代表数据制作的批次，两者区别见表2.2。本文采用的是HumanEva-I，用七台同步相机以每秒60帧的速度记录了四位演员分别表现的六种动作，共计13.5G视频资料，具体数据详见表2.3，相机的摆放位置参照图2.1，图2.2展示了数据库的样例。

表 2.2 HumanEva-I vs HumanEva-II

|         | HumanEva-I | HumanEva-II |
|---------|------------|-------------|
| 同步方式    | 软件         | 硬件          |
| 相机数量    | 7          | 4           |
| 相机种类    | 3彩色+4黑白    | 4彩色         |
| 运动捕捉相机数 | 6          | 8           |
| 数据类别    | 训练、验证、测试   | 测试          |

表 2.3 HumanEva-I训练+验证帧数统计

| 动作 | Subject1 | Subject2 | Subject3 | 合计    |
|----|----------|----------|----------|-------|
| 走路 | 1315     | 1047     | 1020     | 3382  |
| 慢跑 | 820      | 952      | 908      | 2680  |
| 投掷 | 1052     | 1256     | 1111     | 3419  |
| 挥手 | 872      | 1052     | 1199     | 3123  |
| 拳击 | 863      | 916      | 1084     | 2863  |
| 合计 | 4922     | 5223     | 5322     | 15467 |

然而，数据库只对部分视频资料给出了运动姿态的ground-truth，四号演员的所有视频都只能用来测试，一到三号演员也有一半视频都没有ground-truth。此外，由于运动捕捉系统本身的问题，并不是对所有视频帧都能给出正确的三维姿态，因此经过筛选后<sup>[?][?]</sup>文章所使用的训练数据如表2.4所示。在本文中，训练使用的帧数比<sup>[?][?]</sup>文章较多，具体见表2.5。

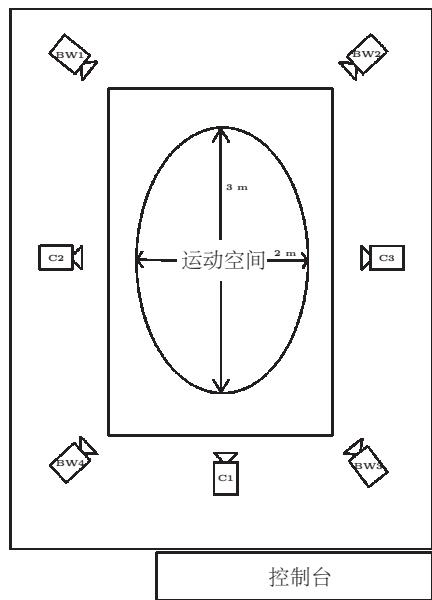


图 2.1 相机摆放位置

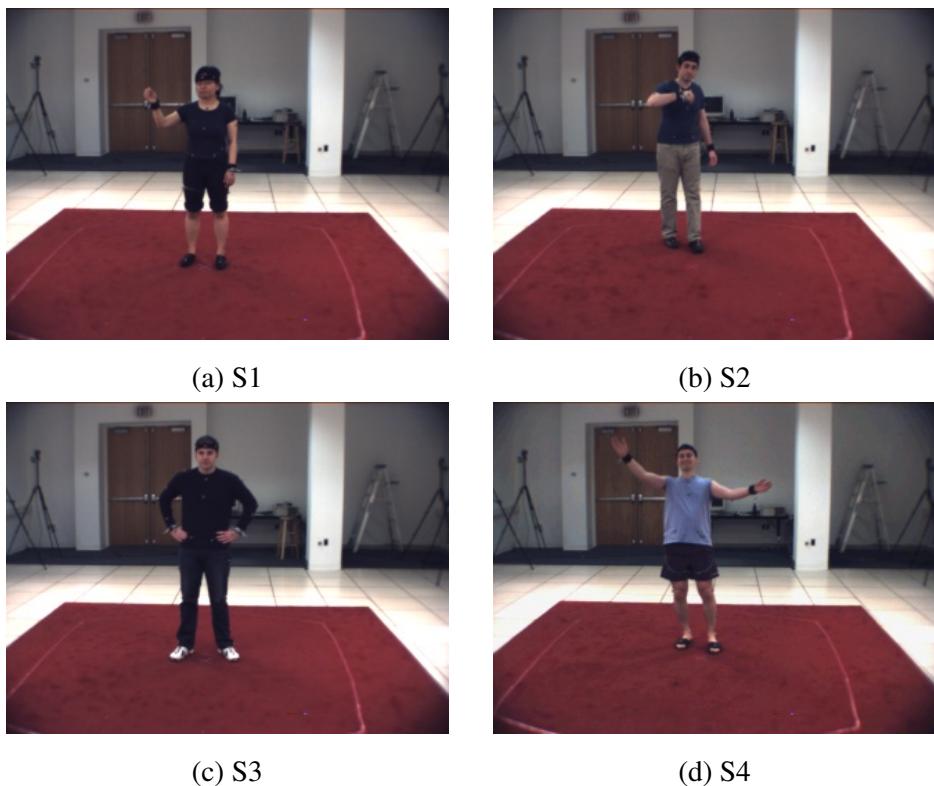


图 2.2 HumanEva-I样例

表 2.4 [?][?] 文章所用帧数统计

| 动作 | Subject1 | Subject2 | Subject3 | Total |
|----|----------|----------|----------|-------|
| 走路 | 1176     | 876      | 895      | 2947  |
| 慢跑 | 439      | 795      | 831      | 2065  |
| 投掷 | 217      | 806      | 0        | 1023  |
| 挥手 | 801      | 681      | 214      | 1696  |
| 拳击 | 502      | 464      | 933      | 1889  |
| 合计 | 3135     | 3622     | 2873     | 9630  |

表 2.5 本文所用帧数统计

| 动作 | Subject1 | Subject2 | Subject3 | Total |
|----|----------|----------|----------|-------|
| 走路 | 1220     | 913      | 976      | 3109  |
| 慢跑 | 506      | 824      | 897      | 2227  |
| 投掷 | 217      | 815      | 0        | 1032  |
| 挥手 | 872      | 690      | 260      | 1822  |
| 拳击 | 576      | 470      | 996      | 2042  |
| 合计 | 3391     | 3712     | 3129     | 10232 |

## 2.2 人体三维骨架表示

本文采用的人体三维骨架表示方法同[?], 将人体用20个关节点来描述, 每个关节点用 $\mathbf{P} = (x, y, z)$ 三维坐标表示, 串联起来构成 $20 \times 3 = 60$ 维向量。在二维图像上表示骨架如图2.3, 三维表示如图2.4, 20个关节点的具体含义参见表2.6。为了使每个骨架对齐, 我们把躯干远端作为三维坐标的原点, 所有其他点坐标都根据原点做平移变换。

| 躯干 | 头部 | 胳膊 |    |    |    | 腿  |    |    |    |
|----|----|----|----|----|----|----|----|----|----|
|    |    | 左前 | 左后 | 右前 | 右后 | 左前 | 左后 | 右前 | 右后 |
| 近端 | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  |
| 远端 | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  |

表 2.6 关节点构成

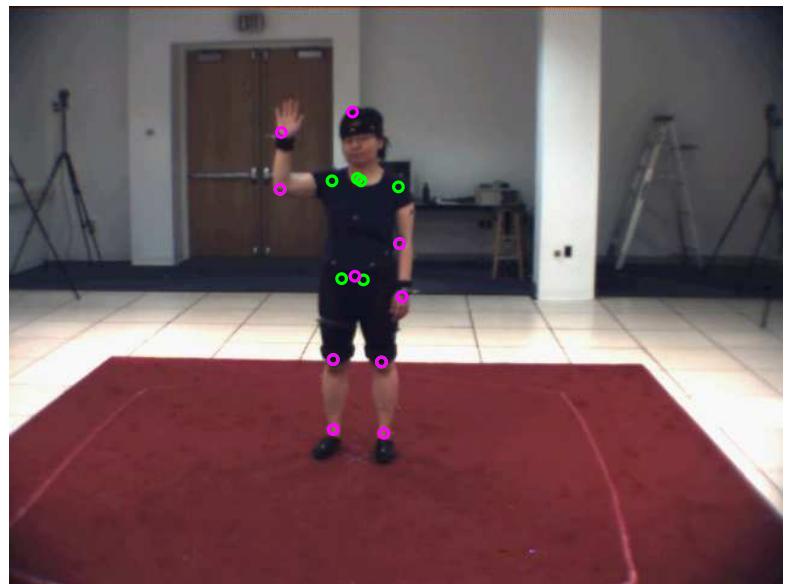


图 2.3 二维骨架表示

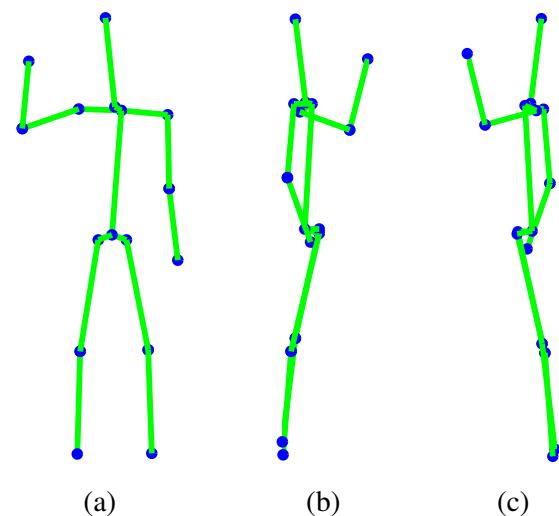


图 2.4 三维骨架表示

## 第3章 提取特征

本文采用的是基于特征的方法，如何用合适的特征描述图像显得至关重要。首先，选取的特征需要能很好地刻画人物的姿态信息，即相似姿态具有相似的特征，不同姿态的特征的区别也要足够明显，尽可能让特征只包含姿态信息，不会因为光照、图片尺寸、衣服、人物体型的改变受到剧烈变化。其次，特征的维数不能太高，否则会增加学习和预测过程中的计算量。同时提取特征本身的计算量也要加以限制。经过调查发现，目前最好的姿态估计算法<sup>[?][?][?]</sup>大都采用了HOG（Histogram of Oriented Gradient，梯度直方图）的特征，也有一些文献采用了其他的特征，如<sup>[?]</sup>采用了SIFT和HistoSC的特征，<sup>[?]</sup>采用了Lab颜色空间统计和皮肤检测的特征，本文同多数算法一样采用了HOG特征，并做了一些改进，此外，还提出了一种能表示皮肤信息的特征，取得了较好的结果。

### 3.1 图片预处理

为了剔除环境对提取特征的影响，在提取特征前首先要提取ROI（Region Of Interests）。在本文中，ROI即为图片中的人物，提取ROI即背景分离，任务是把人物从图片中分割出来，并裁剪图片。由图2.2可以看出HumanEva数据库的环境是非常复杂的，好在数据库提供了纯背景的图像序列3.1，因此可以在已知背景情况下分割人物。尽管这项任务是计算机视觉领域中一个非常久远且已经攻克的问题，但由于这不是本文重点，因此没有花太多时间和精力调查文献和实现算法，用一系列简单的操作实现了比<sup>[?]</sup>更好的分割效果。

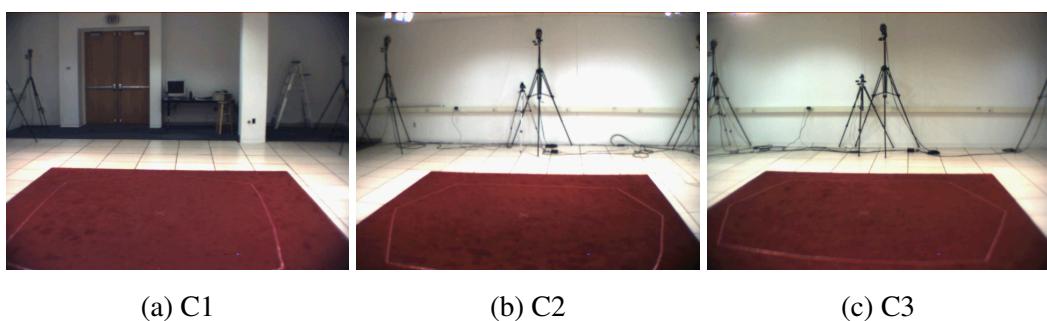


图 3.1 HumanEva 背景

### 3.1.1 计算背景数据

给出的背景不是单张图片，而是一段视频，因此可利用的信息更多。本文假定视频中的 $N$ 帧图像每个像素点都满足高斯分布，记像素点 $\mathbf{c} = (r, g, b)$ ，则 $\mathbf{c} \sim \mathcal{N}(\mu, \sigma)$ ，通过统计所有视频帧可以算出 $\mu$ 和 $\sigma$ ，用待分割图片中的每一个像素点代入高斯分布，即可算出该点是否为背景的概率。给定一个阈值，即可产生一个粗糙的二值图分割，见图3.2。

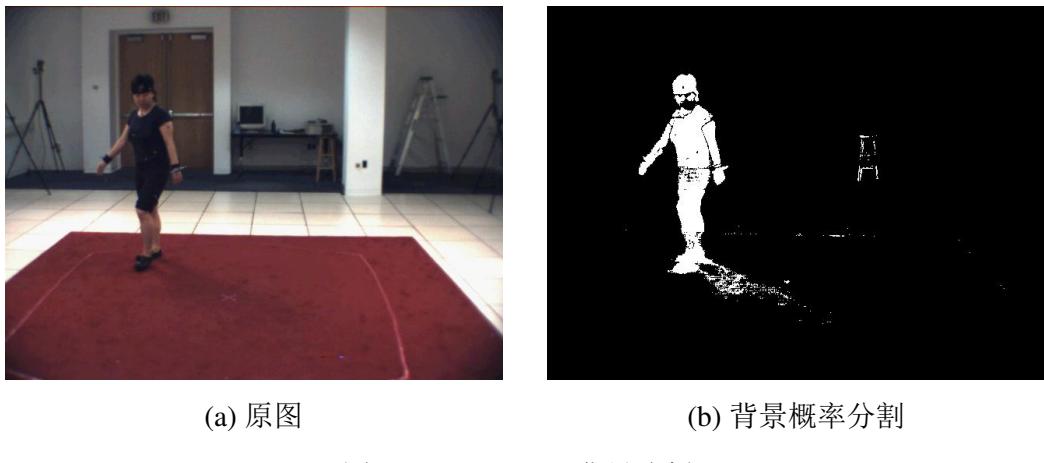
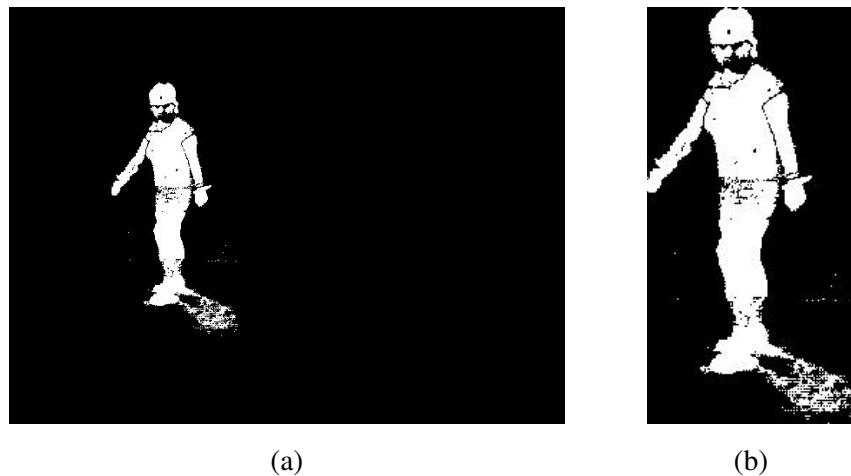


图 3.2 HumanEva 背景分割 I

### 3.1.2 局部优化

可以看出，用简单的阈值分割无法保证正确性，部分背景被认为是前景（即所需人物），而前景中夹杂的与背景相似的颜色（尤其是脸部，和背景中的门颜色过于相近，残缺很多）则被认为是背景，于是又做了很多优化。首先，需要把明显不是前景的背景去掉，这里考虑用距离来度量，即偏离人物较远的均认为是背景，先找到图中最大面积区域的位置，该区域被近似为前景位置，将偏离该区域较远的区域都认为是背景（因为数据库中人物的动作范围不大，长宽均有限，这里把阈值选为80 像素），可以从图3.3看出偏远大块的背景已经被去掉。

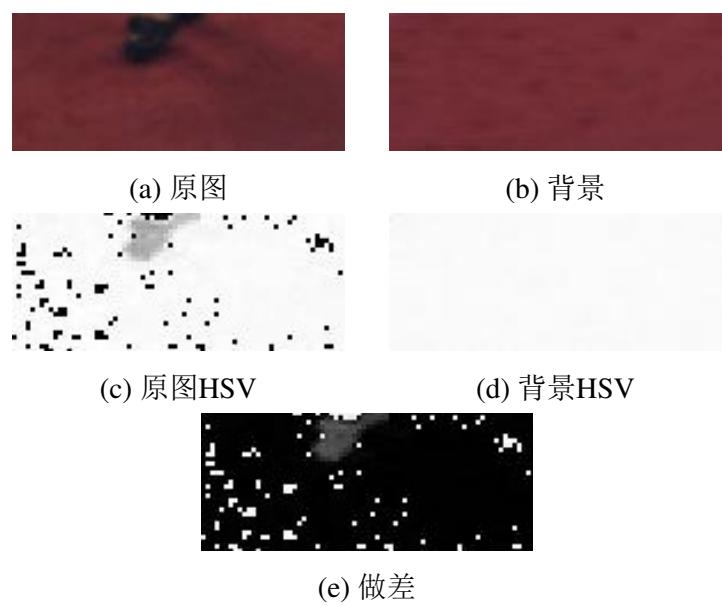
然而由于人物遮挡光线产生的阴影还存在，因此需要考虑去掉人物产生的阴影。去除阴影用了最简单的方法，即把原图先裁剪至最小前景区域，只考虑该区域底部20%的区域，将这小部分区域转换到HSV空间，同样在纯背景中对相同区域做HSV变换，仅对色调通道做差，手工选定阈值，再与已有分割对应部分做与运算，即可将阴影去除，阴影去除后的效果见图3.5(a)。



(a)

(b)

图 3.3 HumanEva背景分割2



(a) 原图

(b) 背景

(c) 原图HSV

(d) 背景HSV

(e) 做差

图 3.4 去除阴影

此时还有一些背景噪音，接着去掉了前景中面积较小的部分，面积的阈值可以选的小一些，在这里选择为40像素（图像为 $480 \times 640$ 像素），这些部分大多都是未能剔除的环境噪音，此时结果参见图3.5(b)。

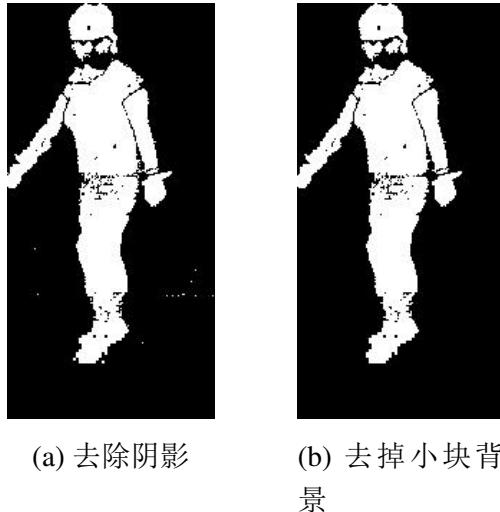


图 3.5 HumanEva背景分割3

由于人物的脸部等地方有较大缺失，于是接着做了8次膨胀<sup>①</sup>（半径为1的圆形，见图3.6(a))，和8次腐蚀（同样半径为1，见图3.6(b))，最后做一次背景抹除（面积小于400像素的都被消去），此时只剩下了单一连通的前景。

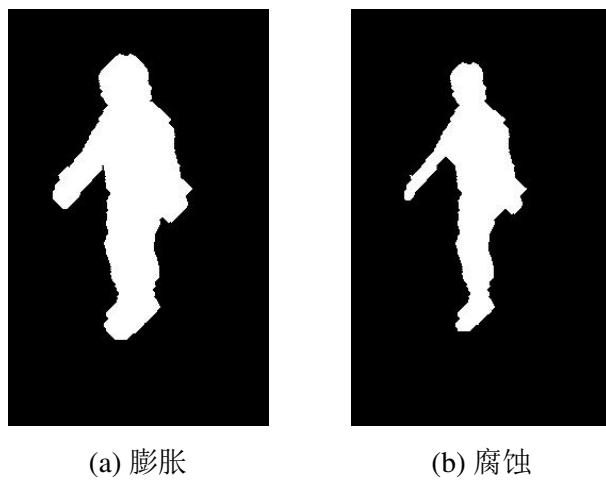


图 3.6 HumanEva背景分割4

---

① 在膨胀前先对图像加框，防止越界

与<sup>[?]</sup>对比可以看出，该文献的前景（见图3.7(a)）有较多空洞，这对于HoG的计算会带来很大误差，而本文提出的分割方法（见图3.7(b)）没有小块噪音，整个前景连为一体，虽然在边缘轮廓不是特别准确，但是对于HoG特征来说影响不大。

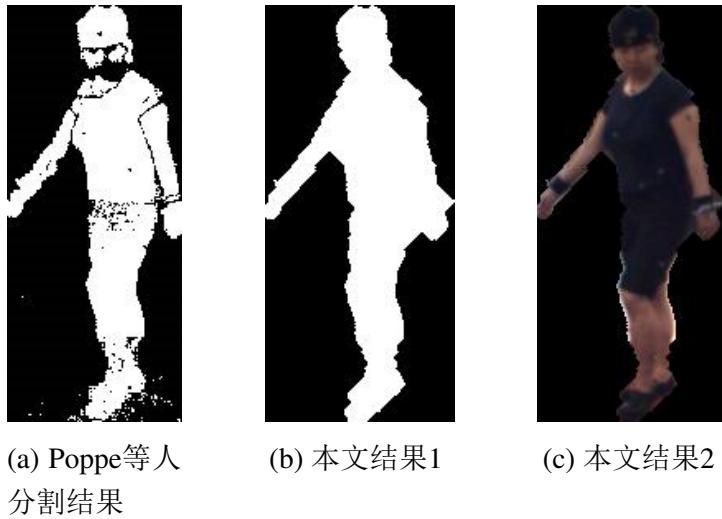


图 3.7 背景分割对比

此外，在处理演员2的时候由于衣服颜色和背景过于相像，出现了图3.8 所示的错误（上身部分缺失），针对这种特殊情况，先找到前景所占矩形区域，然后填充矩形区域中上部60%范围内的空洞。这一操作虽然不至于填充两腿缝隙，但对于胳膊与腰造成的空洞也会一并填充，鉴于这种情况出现的帧数极少，故忽略此问题。



图 3.8 背景分割错误

## 3.2 HOG特征

### 3.2.1 概述

HOG特征最早由Navneet Dalal和Bill Triggs在2005年提出<sup>[?]</sup>，被成功地用于目标检测，尤其是行人检测。该方法是对图像的局部区域出现的方向梯度进行统计，和SIFT（scale-invariant feature transform）相似，不同的是HOG用了重叠的局部对比度归一化（overlapping local contrast normalization）技术。HOG描述子的主要贡献是很好地描述了局部区域的边缘方向密度。对图像进行归一化后，能够很好地保证照射和阴影不变性。与其他方法相比，HOG有很好的几何和光学转化不变性，因此十分适合行人检测和姿态估计。

### 3.2.2 计算HOG描述子

HOG特征的提取方法主要步骤如下：

1. 转化为灰度图像
2. 划分成若干子区域（cell）
3. 计算每个像素的梯度
4. 统计每个子区域内梯度直方图

具体说来，本文将 $r, g, b$ 三个通道认为是三张灰度图，分别求取HOG特征，最后取每一维最大值，最后做归一化处理。伪代码见代码1。

---

#### Algorithm 1 MAX-HOG

---

```
1: for  $i = 0$  to  $image.NumberOfChannels$  do
2:    $H(i) \leftarrow GetHOG(image.channel(i));$ 
3: end for
4:  $H \leftarrow MAX(H(i));$ 
5:  $H \leftarrow L1Norm(H);$ 
6: return  $H$ 
```

---

针对每一张灰度图求取HOG特征的算法伪代码见代码2。

此处需要强调一下，Poppe等人对于区域数量的选取是 $5 \times 6 = 30$ ，由于每张图片人物大小不一、长宽比例不固定，因此每个子区域的长宽比也不固定，这会导致一些问题，比如在不同长宽比的图片中，头部在细长的图片中会出现

---

**Algorithm 2** GetHOG

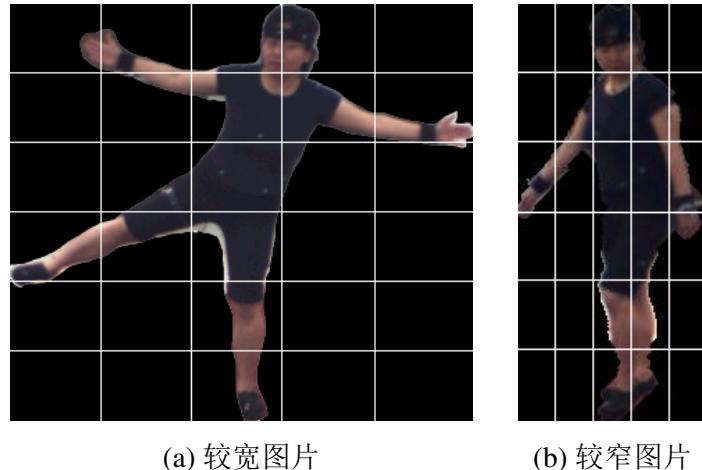
---

```
1: \\\设置区域数量, 横向6个, 纵向9个
2: Cell_Num_x ← 6;
3: Cell_Num_y ← 9;
4: \\\计算梯度
5: gradx ← CalGrad(image, hx);
6: grady ← CalGrad(image, hy);
7: angle ← GetAngle(gradx, grady);\\计算梯度方向
8: magnit ←  $\sqrt{(grad_y u^2 + grad_x r^2)}$ ;\\计算梯度大小
9: \\\计算直方图
10: for i = 1 to Cell_Num_x × Cell_Num_y do
11:   for j = 0 to 180, interval=20 do
12:     H(i, j) ←  $\sum magnit(j \leq angle \leq j+interval)$ ;
13:   end for
14: end for
15: H ← L1Norm(H);
16: return H
```

---

在多个子区域中, 而较宽的图片中则只有一个区域有头部, 图片本身的比例信息已经丢失, 见图3.9。为了解决这个问题, 本文在求取HOG特征之前先将图片填充到3:2 的比例 (见图) 3.10, 这样求取的特征就不会变形, 虽然有较多区域是无用信息, 占用了维数, 但是经过测试, 这并不会增加额外的存储占用, 因为数据本身是稀疏的。

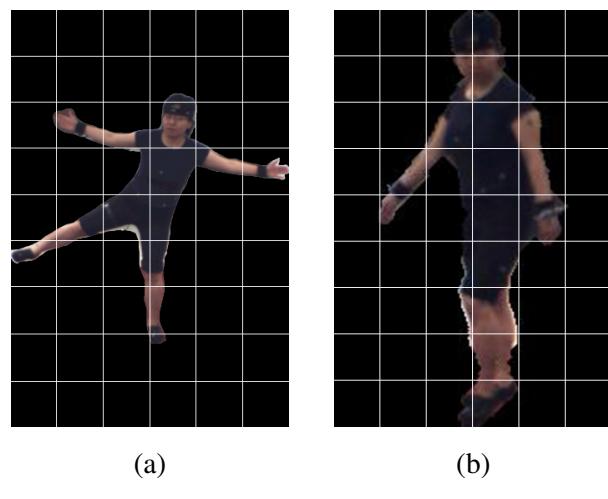
每个像素点根据周围像素信息都可以算得一个梯度信息, 包括梯度的方向和大小。计算直方图指的是对于每一个子区域, 对梯度的方向做直方图, 方向范围是 $0^\circ \sim 180^\circ$ , 将直方图分为9块, 每块含 $20^\circ$ , 对在该角度范围的梯度的大小求和作为直方图中这一块的值。子区域的直方图统计可以用图3.11(c)来表示, 每一个方向用该方向上的矩形表示, 颜色越亮表示梯度越大。最后做L1 规范化, 得到最终的HOG 特征向量。



(a) 较宽图片

(b) 较窄图片

图 3.9 图片比例影响示意



(a)

(b)

图 3.10 图片固定比例示意

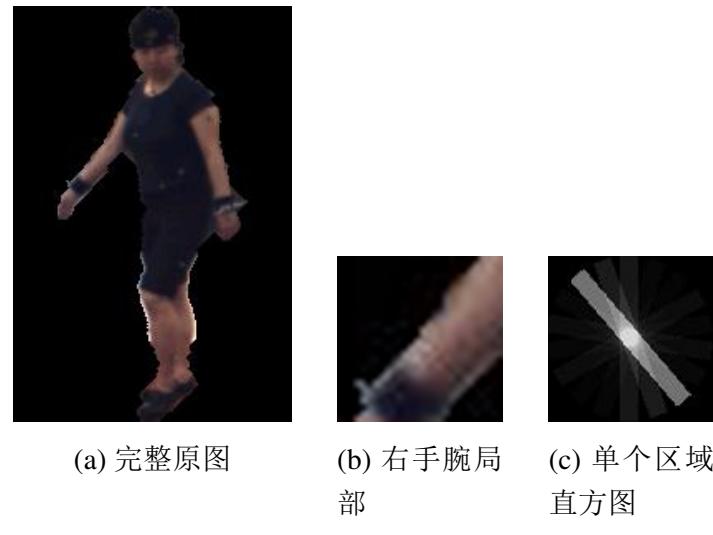


图 3.11 HOG特征子区域直方图表示

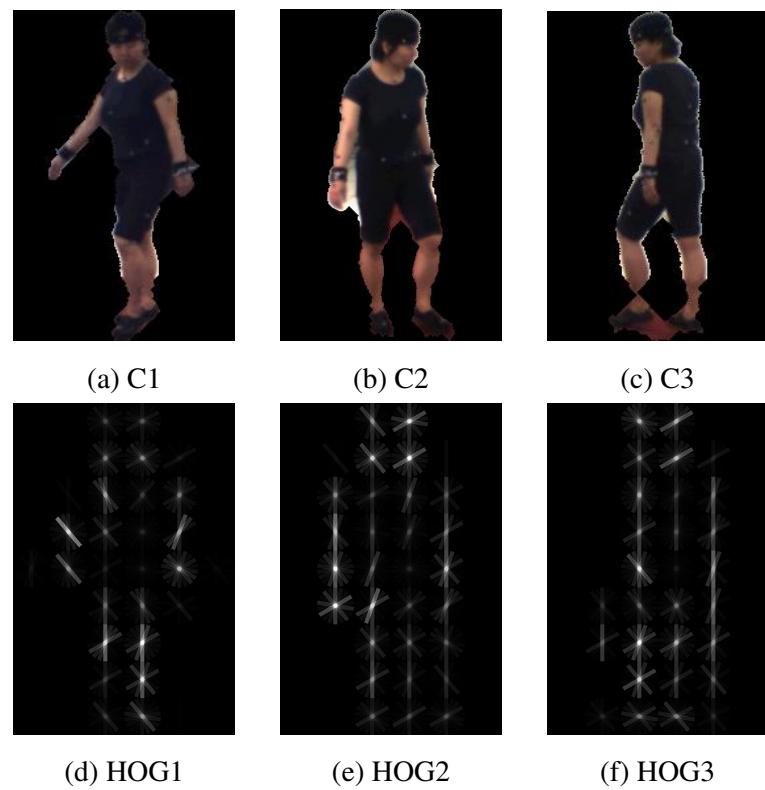


图 3.12 HOG特征

### 3.3 皮肤特征

由于人的皮肤是一个不变的特征，而且胳膊、头的位置对于姿态估计有重要作用，因此充分利用皮肤特征是十分必要的。对于姿态估计，我们只关心胳膊、头的位置，因此本文用位置、面积信息来描述皮肤特征。首先，需要判断每个像素属于皮肤的概率，为了简化问题，本文假定皮肤颜色在YCbCr空间服从高斯分布，均值和方差均为手工给出。由于人们往往是穿着长裤，因此只能检测到头和左右臂这三个部分，而在穿长袖衣服时则只能检测到头和左右手。因此本文只选取概率最大的三个区域<sup>①</sup>，分别认为是头、左臂和右臂。按面积由大到小排序，每一个区域用 $\mathbf{r} = (x, y, s)$ 表示， $(x, y)$ 为区域的重心坐标， $s$ 为区域面积。检测结果见图3.13。

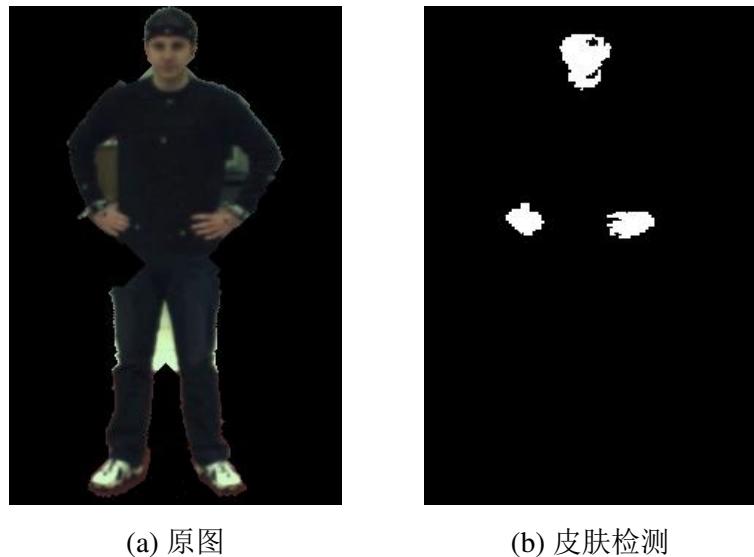


图 3.13 皮肤检测

### 3.4 特征表示

HOG特征有 $6 \times 9 = 36$ 个子区域，每个子区域有9维直方图，因此共计 $36 \times 9 = 486$ 维。皮肤特征有3个区域，每个区域 $(x, y, s)$ 三维信息，共计 $3 \times 3 = 9$ 维。两种特征相连共计 $486 + 9 = 495$ 维。如果用到三个视角图片，则会有 $495 \times 3 = 1485$ 维。

<sup>①</sup> 只考虑ROI中的上半部分。有时候会小于三个区域，因为手和头部相连。

## 第4章 基于双高斯过程的姿态估计算法

### 4.1 概述

由第3.4节和第2.2节可知，本文研究问题的输入输出已经刻画成两个一维向量，记输入向量（特征向量）为 $\mathbf{R}$ ，输出向量（骨架向量）为 $\mathbf{X}$ ，则 $\mathbf{X} = \mathbf{F}(\mathbf{R})$ ，可以归结为一个多输出多元回归问题。显然，输出向量的每个元素之间是有关联的，用若干简单的回归方法（如线性回归、SVM等方法）独立预测各元素是不合理的。此外，我们知道相似的特征应该表示相似的人体骨架，因此，自然想到了KNN ( $k$  nearest neighbor regression) 算法，但该算法效果并不理想，因此本文采用了一种能更好描述向量之间相关性的算法——双高斯过程回归 (Twin Gaussian Process Regression，以下简称TGP)。在介绍该算法前，首先有必要介绍其前身和基础——高斯过程回归 (Gaussian Process Regression，以下简称GPR)。

### 4.2 高斯过程回归

高斯过程回归是基于贝叶斯理论和统计学习理论发展起来的一种机器学习方法，对于高维数、小样本和非线性等复杂回归问题行之有效。记输入向量为 $\mathbf{R} = (\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_N)$ ,  $\dim(\mathbf{r}_i) = W$ ，对应的输出向量为 $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N)$ ,  $\dim(\mathbf{x}_i) = D$ ,  $N$ 为样例的个数。简单来说，GPR就是假定输入向量的协方差满足高斯分布。先定义 $f(\mathbf{r})$ 是输入 $\mathbf{r}$ 的一个函数，用公式4-1表示输入向量的协方差，协方差函数可以有多种取法，公式4-2即为一种取法。

$$\text{cov}(f(\mathbf{r}_i), f(\mathbf{r}_j)) = K_R(\mathbf{r}_i, \mathbf{r}_j) \quad (4-1)$$

$$K_R(\mathbf{r}_i, \mathbf{r}_j) = \exp(-\gamma_r \|\mathbf{r}_i - \mathbf{r}_j\|^2) + \lambda_r \delta_{ij} \quad (4-2)$$

其中 $\lambda_r$ 用来描述噪声，

$$\delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases} \quad (4-3)$$

又因为 $\mathbf{X} = \mathbf{F}(\mathbf{R})$ , 所以有公式4-4

$$\begin{bmatrix} (\mathbf{X}^{(d)})^\top \\ \mathbf{x}^{(d)} \end{bmatrix} \sim \mathcal{N}_R \left( \mathbf{0}, \begin{bmatrix} \mathbf{K}_R & \mathbf{K}_R^r \\ (\mathbf{K}_R^r)^\top & K_R(\mathbf{r}, \mathbf{r}) \end{bmatrix} \right) \quad (4-4)$$

其中 $\mathbf{X}^{(d)}$ 是 $\mathbf{X}$ 的第 $d$ 列,  $\mathbf{x}^{(d)}$ 是 $\mathbf{x}$ 的第 $d$ 个元素;  $\mathbf{K}_R$ 是 $N \times N$ 矩阵,  $(\mathbf{K}_R)_{ij} = K_R(\mathbf{r}_i, \mathbf{r}_j)$ ;  $\mathbf{K}_R^r$ 是 $N \times 1$ 的向量,  $(\mathbf{K}_R^r)_{ij} = K_R(\mathbf{r}_i, \mathbf{r})$ , 用来描述输入向量的相关性;  $K_R(\mathbf{r}, \mathbf{r})$ 是 $\mathbf{r}$ 自身的协方差。

用 $\mathbf{X}^{(d)}$ 表示观测数据,  $\mathbf{x}^{(d)}$ 表示预测数据, 因为二者联合分布符合高斯分布, 因此后验概率 $p(\mathbf{x}^{(d)} | (\mathbf{X}^{(d)})^\top)$ 也满足高斯分布, 均值和方差分别见公式4-5、4-6, 即可用均值表示输入 $\mathbf{r}$ 的预测结果。

$$\text{mean}(\mathbf{x}^{(d)}) = \mathbf{X}^{(d)} \mathbf{K}_R^{-1} \mathbf{K}_R^r \quad (4-5)$$

$$\sigma^2(\mathbf{x}^{(d)}) = K_R(\mathbf{r}, \mathbf{r}) - (\mathbf{K}_R^r)^\top \mathbf{K}_R^{-1} \mathbf{K}_R^r \quad (4-6)$$

以上阐述了如何利用GPR进行预测, 关于GPR的参数如何学习计算, 可以采用共轭梯度法、牛顿法等优化方法求得参数的最优解, 本文不再详述。

由于GPR只关心单个输出的预测, 没有考虑每个输出之间的关系, 而且对于从二维平面恢复三维姿态这样一个多解问题GPR显得无能为力, 因此本文采用了双高斯过程回归, 将在4.3节介绍。

## 4.3 双高斯过程回归

### 4.3.1 概述

为了考虑各输出变量之间的相关性, 借鉴GPR的思路, 我们假定输出变量之间也符合关于输出的高斯分布:

$$\begin{bmatrix} (\mathbf{X}^{(d)})^\top \\ \mathbf{x}^{(d)} \end{bmatrix} \sim \mathcal{N}_X(\mathbf{0}, \mathbf{K}_{\mathbf{X} \cup \mathbf{x}}) \quad (4-7)$$

其中相关性矩阵为:

$$\mathbf{K}_{\mathbf{X} \cup \mathbf{x}} = \begin{bmatrix} (\mathbf{X}^{(d)})^\top \mathbf{X}^{(d)} & (\mathbf{X}^{(d)})^\top \mathbf{x}^{(d)} \\ \mathbf{X}^{(d)} \mathbf{x}^{(d)} & \mathbf{x}^{(d)} \mathbf{x}^{(d)} \end{bmatrix} \quad (4-8)$$

于是  $\begin{bmatrix} (\mathbf{X}^{(d)})^\top \\ \mathbf{x}^{(d)} \end{bmatrix}$  服从  $N_X$  和  $N_R$  两个正态分布。我们希望这两个分布尽量一致， $N_X$  的参数是需要估计的，通过优化调整  $\mathbf{x}^{(d)}$  使两个分布尽量接近。

### 4.3.2 KL散度

为了刻画相同事件空间里的两个概率分布的相似度，在此引入KL散度（Kullback-Leibler Divergence），其物理意义是：在相同事件空间里，概率分布  $P(x)$  的事件空间，若用概率分布  $Q(x)$  编码时，平均每个基本事件（符号）编码长度增加了多少比特<sup>①</sup>。计算公式为：

$$D(P\|Q) = \sum P(x) \log \frac{P(x)}{Q(x)} \quad (4-9)$$

具体在这里  $N_X$  和  $N_R$  的KL散度可表示为：

$$\begin{aligned} D_{KL}(N_X\|N_R) &= -\frac{N}{2} - \frac{1}{2} \log |\mathbf{K}_{\mathbf{X} \cup \mathbf{x}}| \\ &\quad + \frac{1}{2} \text{Tr} \left\{ \mathbf{K}_{\mathbf{X} \cup \mathbf{x}} \begin{bmatrix} \mathbf{K}_R & \mathbf{K}_R^r \\ (\mathbf{K}_R^r)^\top & K_R(\mathbf{r}, \mathbf{r}) \end{bmatrix}^{-1} \right\} \\ &\quad + \frac{1}{2} \log \left\| \begin{bmatrix} \mathbf{K}_R & \mathbf{K}_R^r \\ (\mathbf{K}_R^r)^\top & K_R(\mathbf{r}, \mathbf{r}) \end{bmatrix} \right\| \end{aligned} \quad (4-10)$$

于是预测三维姿态  $\mathbf{x}$  转化成一个优化问题：

$$\mathbf{x}^* = \underset{\mathbf{x}^{(d)}}{\operatorname{argmin}} \left[ L(\mathbf{x}^{(d)}) \equiv D_{KL}(N_X\|N_R) \right] \quad (4-11)$$

其中  $\text{Tr}$  表示求矩阵的迹。

### 4.3.3 求解优化问题

通过去掉公式4-10中和  $\mathbf{x}$  的无关项，加以简化可以得到式4-12

$$\begin{aligned} L(\mathbf{x}^{(d)}) &= \mathbf{x}^{(d)} \mathbf{x}^{(d)} - 2 \mathbf{x}^{(d)} \mathbf{X}^{(d)} \mathbf{K}_R^{-1} \mathbf{K}_R^r \\ &\quad - [K_R(\mathbf{r}, \mathbf{r}) - (\mathbf{K}_R^r)^\top \mathbf{K}_R^{-1} \mathbf{K}_R^r] \\ &\quad \times \log \left\{ \mathbf{x}^{(d)} \mathbf{x}^{(d)} - \mathbf{x}^{(d)} \mathbf{X}^{(d)} \left[ (\mathbf{X}^{(d)})^\top \mathbf{X}^{(d)} \right]^{-1} (\mathbf{X}^{(d)})^\top \mathbf{X}^{(d)} \right\} \end{aligned} \quad (4-12)$$

---

<sup>①</sup> url:[http://en.wikipedia.org/wiki/Kullback-Leibler\\_divergence](http://en.wikipedia.org/wiki/Kullback-Leibler_divergence)

如果预测三维姿态变量各维之间是独立的，那么可以写成公式4-13

$$\begin{aligned} L(\mathbf{x}) = & \mathbf{x}^\top \mathbf{x} - 2\mathbf{x}^\top \mathbf{X} \mathbf{K}_R^{-1} \mathbf{K}_R^r \\ & - \left[ K_R(\mathbf{r}, \mathbf{r}) - (\mathbf{K}_R^r)^\top \mathbf{K}_R^{-1} \mathbf{K}_R^r \right] \\ & \times \log \left[ \mathbf{x}^\top \mathbf{x} - \mathbf{x}^\top \mathbf{X} (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{x} \right] \end{aligned} \quad (4-13)$$

然而显然变量各维之间是不独立的，因此同对待输入变量一样，我们为输出变量各维之间也定义相关函数：

$$\text{cov}(f(\mathbf{x}_i), f(\mathbf{x}_j)) = K_R(\mathbf{x}_i, \mathbf{x}_j) \quad (4-14)$$

同样相关函数也可以选择和公式4-2类似的高斯函数。于是我们把输出变量的相关矩阵表示为式4-15

$$\mathbf{K}_{\mathbf{X} \cup \mathbf{x}} = \begin{bmatrix} \mathbf{K}_X & \mathbf{K}_X^x \\ (\mathbf{K}_X^x)^\top & K_X(\mathbf{x}, \mathbf{x}) \end{bmatrix} \quad (4-15)$$

于是最终优化问题变为式4-16

$$\begin{aligned} L(\mathbf{x}) = & K_X(\mathbf{x}, \mathbf{x}) - 2(\mathbf{K}_X^x)^\top \mathbf{K}_R^{-1} \mathbf{K}_R^r \\ & - \left[ K_R(\mathbf{r}, \mathbf{r}) - (\mathbf{K}_R^r)^\top \mathbf{K}_R^{-1} \mathbf{K}_R^r \right] \log \left[ K_X(\mathbf{x}, \mathbf{x}) - (\mathbf{K}_X^x)^\top \mathbf{K}_X^{-1} \mathbf{K}_X^x \right] \end{aligned} \quad (4-16)$$

通过求取梯度，令梯度为0，即可求得最优解。

#### 4.4 TGP-KNN

由于TGP方法需要求 $N \times N$ 矩阵的逆，当 $N$ 很大时复杂度会很高，因此需要做一定的近似来降低复杂度，在这里考虑用KNN ( $k$  nearest neighbors) 方法先在 $N$ 个数据中选择 $k$ 个和待预测数据最接近的，然后再用TGP来实现。

## 第5章 结果比较与分析

### 5.1 运行环境

表 5.1 程序运行环境

|      |                           |
|------|---------------------------|
| CPU  | Intel Core2 Q9550 2.83Ghz |
| 内存   | 6G                        |
| 操作系统 | Windows8 企业版 64位          |
| 程序环境 | MATLAB R2013a             |
| 备注   | 4核并行                      |

### 5.2 测试集

测试集和训练集（见表2.3）拍摄条件相同，由同样的三位演员表现了相同的五类姿态，但除此之外，还增加了额外的动作以及新增了一位演员，因此测试集共有四位演员六类姿态，具体每段视频所含帧数统计见表5.2，该表给出了用于可用于提取特征的测试帧的数目统计。由于这些测试集只给出了视频，没有给出ground-truth，因此只能通过网络提交<sup>①</sup>的方法进行测试。将预测姿态结果写成xml格式，通过网络提交后，经过一段时间（理论速度为500 帧/20 分钟）即可收到邮件报告，下载xml文件后经过解析即可获得结果，该结果包含了预测误差。同样由于数据库在制作时由于运动捕捉系统的错误出现了一些无效数据，此时网络系统会返回N/A。剔除之后有效测试数据统计见表5.3。由于Subject1的两个结果（投掷、混合）系统没有返回，无法获得有效帧数。

① utl:[http://vision.cs.brown.edu/humaneva/submit\\_results.html](http://vision.cs.brown.edu/humaneva/submit_results.html)

表 5.2 HumanEva-I 测试帧数统计

| 动作 | Subject1 | Subject2 | Subject3 | Subject4 | 合计    |
|----|----------|----------|----------|----------|-------|
| 走路 | 999      | 1088     | 800      | 662      | 3549  |
| 慢跑 | 869      | 722      | 859      | 585      | 3035  |
| 投掷 | 946      | 1394     | 995      | 768      | 4103  |
| 挥手 | 1065     | 1057     | 548      | 454      | 3124  |
| 拳击 | 601      | 984      | 748      | 569      | 2902  |
| 混合 | 2597     | 1800     | 1793     | 1097     | 7287  |
| 合计 | 7077     | 7045     | 5743     | 4135     | 24000 |

表 5.3 HumanEva-I 有效测试帧数统计

| 动作 | Subject1 | Subject2 | Subject3 | Subject4 | 合计    |
|----|----------|----------|----------|----------|-------|
| 走路 | 762      | 1082     | 383      | 640      | 2867  |
| 慢跑 | 506      | 722      | 844      | 443      | 2515  |
| 投掷 | -        | 1106     | 785      | 613      | 1891  |
| 挥手 | 1052     | 908      | 474      | 441      | 2875  |
| 拳击 | 441      | 700      | 702      | 540      | 2383  |
| 混合 | -        |          | 1742     | 1027     | 3472  |
| 合计 | 2761     | 6248     | 4930     | 3704     | 16003 |

### 5.3 误差度量

对于单个姿态的误差计算，本文采用 AJP (Average Joint Position) 度量标准，即根据式5-1计算

$$Err(\tilde{\mathbf{x}}, \mathbf{x}) = \frac{1}{M} \sum_{i=1}^M \|f_i(\tilde{\mathbf{x}}) - f_i(\mathbf{x})\|_2 \quad (5-1)$$

其中  $f_i(\mathbf{x})$  表示从  $\mathbf{x}$  提取出第  $i$  个关节点的三维坐标。对于一个有  $T$  帧的视频序列，用所有帧的平均误差来表示。

$$Err_{seq} = \frac{1}{T} \sum_{i=1}^T Err(\tilde{\mathbf{x}}_i, \mathbf{x}_i) \quad (5-2)$$

以下结果的单位均为毫米 (mm)。

## 5.4 测试结果

### 5.4.1 HumanEva测试集

这里用TGP方法对表5.3所述数据集进行了测试，并与原文进行了对比，结果请见表5.4，可以看出，本文的结果和原文<sup>[?]</sup>有较大差距，原因分析参见第5.6节。

表 5.4 TGP测试结果

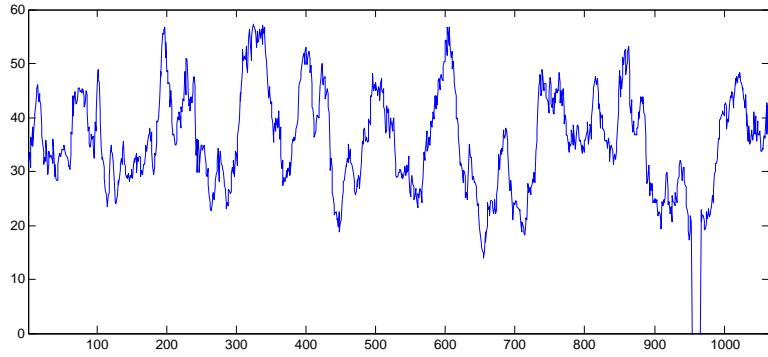
| 动作 | Subject1 | Subject2 | Subject3 | Subject4 | 平均    |
|----|----------|----------|----------|----------|-------|
| 走路 | 49.7     | 70.0     | 169.5    | 286.8    | 126.3 |
| 慢跑 | 68.7     | 215.5    | 131.5    | 278.5    | 162.8 |
| 投掷 | -        | 124.0    | 138.2    | 249.7    | 210.8 |
| 挥手 | 35.9     | 171.0    | 74.7     | 257.4    | 118.9 |
| 拳击 | 81.7     | 208.0    | 193.4    | 250.6    | 190.0 |
| 混合 | -        | 186.1    | 173.8    | 288.1    | 265.1 |
| 平均 | 53.0     | 158.7    | 150.7    | 271.2    | 163.6 |

表 5.5 TGPKNN测试结果

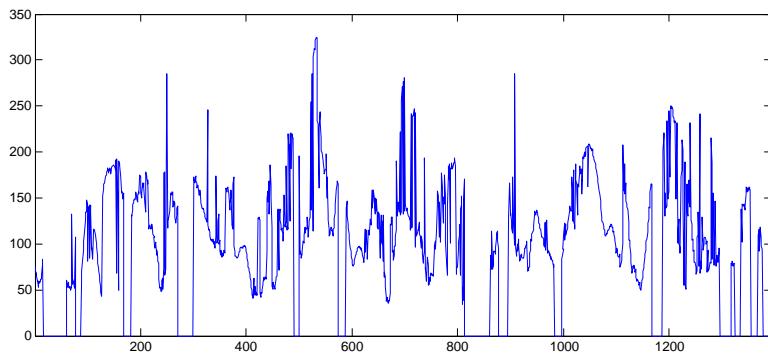
| 动作 | Subject1 |      | Subject2 |      | Subject3 |      | Subject4 |
|----|----------|------|----------|------|----------|------|----------|
|    | 本文       | 原文   | 本文       | 原文   | 本文       | 原文   | 本文       |
| 走路 | 53.4     | 26.6 | 73.2     | 25.2 | 172      | 31.0 | 282.7    |
| 慢跑 | 70.5     | 32.2 | 219.5    | 26.9 | 117.1    | 32.4 | 274.2    |
| 投掷 | -        | -    | 128.6    | 4.05 | 143.3    | 74.1 | 242.5    |
| 挥手 | 36.1     | 19.2 | 176.1    | 50.2 | 68.8     | 50.9 | 254.9    |
| 拳击 | 84       | 57.7 | 201.3    | 72.5 | 198.0    | 75.5 | 255.2    |
| 混合 | -        | -    | 184.2    | 51.9 | 171.9    | 64.6 | 282.1    |
| 平均 | 54.8     | 33.9 | 160.0    | 44.5 | 151.8    | 54.8 | 267.5    |

### 5.4.2 其他数据

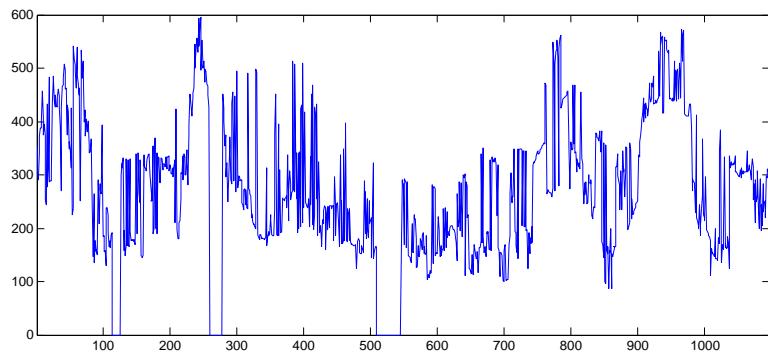
本文将算法用于其他数据，发现无论是<sup>[?]</sup>的特征还是本文所提特征均表现不是很好，不过本文所提特征对于其他数据稍好一点。具体可以参考图5.2，这



(a) S1 Gestures



(b) S2 ThrowCatch



(c) S4 Combo

图 5.1 单个视频逐帧误差

是本人模仿数据库中动作所拍照片，对比结果见图，图5.3。可以看出本文结果更加接近真实情况，尤其是右臂部分。

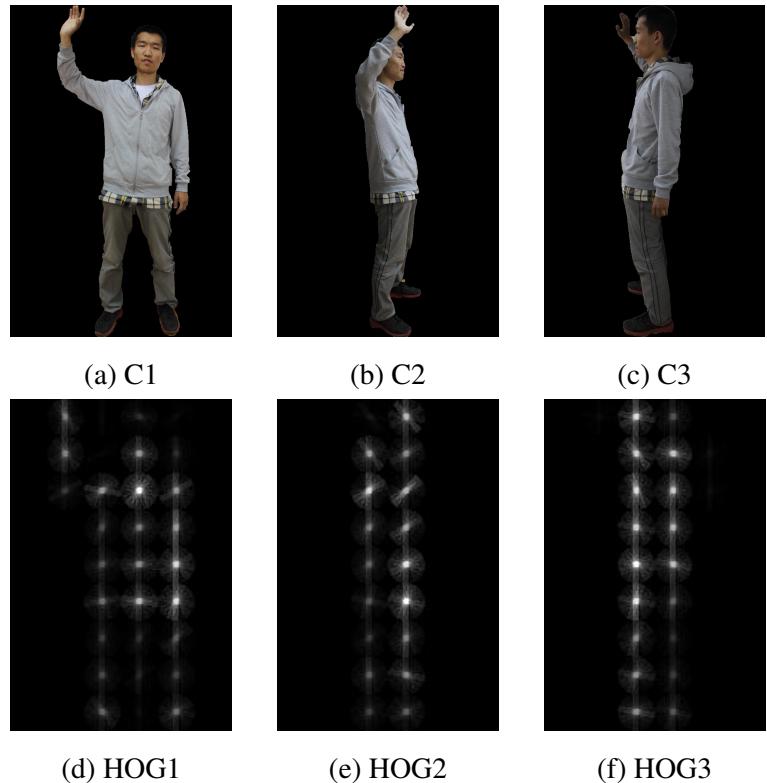


图 5.2 其他数据测试示意1

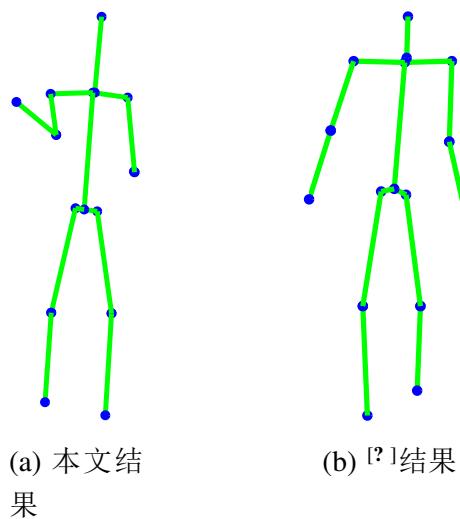


图 5.3 其他数据测试结果对比1

图5.4是一位女性所做的简单动作，结果对比见图5.5。可以看出，两者结果差异不大，左臂都是错误的，但本文结果的右臂更加接近原图。

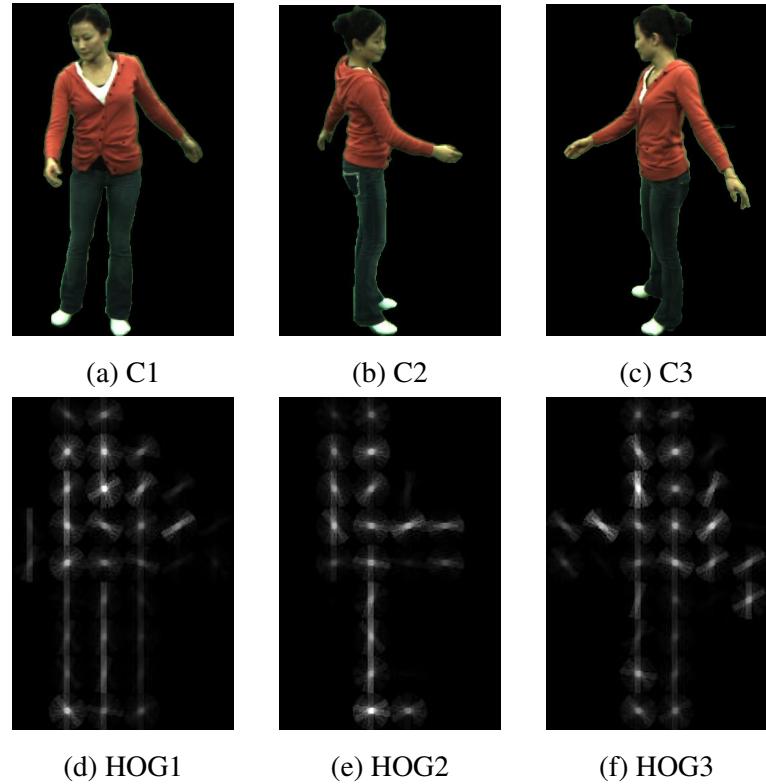


图 5.4 其他数据测试示意2

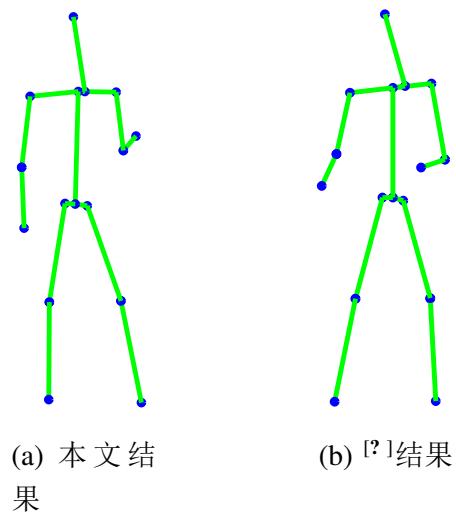


图 5.5 其他数据测试结果对比2

## 5.5 计算时间

本文所做工作非常耗时，具体可以从表5.6看出。

表 5.6 测试各步骤耗时统计

| 步骤                | 帧数    | 耗时                       | 单帧耗时   |
|-------------------|-------|--------------------------|--------|
| 提取训练集特征           | 10232 | 9629 s $\approx$ 2.7 h   | 0.94 s |
| 提取训练集ground-truth | 10232 | 1459 s $\approx$ 0.4 h   | 0.14 s |
| 提取测试集特征           | 24000 | 22778 s $\approx$ 6.3 h  | 0.95s  |
| 训练TGP             | -     | 119 s                    | -      |
| 预测测试集             | 24000 | 37781 s $\approx$ 10.5 h | 0.6 s  |
| 网络测试              | 24000 | $\approx$ 12 h           | 0.6 s  |

## 5.6 结果分析

从以上结果可以看出本文所提特征在测试集上明显不如Bo等人<sup>[?]</sup>的结果，可能的原因如下：

1. 本文的背景分割方法在某些图片上效果不佳
2. 本文所提HoG特征虽然没有改变图片比例，但改变了图片中人物的尺度
3. 本文所用方法的参数选取不合适

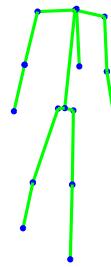
## 第6章 总结与展望

### 6.1 总结

本文实现了一种可以估计人体三维姿态的算法，在大量数据上进行了测试。测试结果并不理想，和原文有较大差距。在实验的过程中，我发现了一些问题，比如即便ground-truth告诉我们当前帧的运动捕捉到的姿态是有效的，实际可能是错误的，可以参照图6.1。



(a) 原图



(b) 错误骨架

图 6.1 错误的ground-truth

本文采用了不同的特征得到了较差的结果，说明了预测结果和特征关系极大，由于这次的实验十分耗时，我没能用更多的时间去选择合适的特征，我非常遗憾。在接下来的工作中，我希望能提出一种更有效的特征来解决这个问题。综合一些文献<sup>[?][?]</sup>的思路，我认为在解决人体姿态估计问题中需要分层处理，即先识别各个肢体的位置和姿态，然后再全局优化。用本文中所述将所有信息一起训练是冗余且低效的，即使用了多达10000帧的数据，并且动作简单到只有几类，效果仍然不是很好。

## 6.2 收获与体会

首先，我认为选题是非常重要的环节。在毕业设计初期，我一直在做很多尝试，调查了很多方向，也阅读了相当多的文献，但都没有找到合适的方向下手。当考虑做多人体三维建模时发现难度很大，于是不断缩小目标，最后落脚到单人体三维姿态估计，即便如此，经过调查我发现这是一个由来已久的难题，于是只好摸索着前进。前期调研花掉了较多的时间，导致后期做实验的时间被大大缩短，也导致了结果的不理想。

其次，我体会到了做图像处理实验的难处。第一，数据库很重要，巧妇难为无米之炊，也正如文献<sup>[?]</sup>所说那样：

We give the last word to Sherlock Holmes:

“Data! Data! Data!” he cried impatiently. “I can’t make bricks without clay.”

*The Adventure of the Copper Beeches*

没有数据就没办法实验，好在本文可以利用HumanEva数据库，在此十分感谢布朗大学。第二点难处是时间，做实验非常漫长，需要很好的耐心，不能轻易放弃。在我没有结果的时候，我的朋友给了我鼓励和帮助，让我坚持了下去，我很感激他们。

当然，我收获了很多知识，不仅仅是人体姿态识别这个领域我了解了最流行最有效的方法，我还在前期调研中探索了很多其他领域，比如图片拼接、3D视频质量评价、去模糊、三维重建、霍夫树等。收获的也不止是知识，还有学习的方法，熟悉了如何调研、查阅文献，怎么去网上找到可以利用的代码和数据库等等。

## 插图索引

|        |                       |    |
|--------|-----------------------|----|
| 图 1.1  | 本文研究问题的输入与输出 .....    | 4  |
| 图 2.1  | 相机摆放位置 .....          | 9  |
| 图 2.2  | HumanEva-I样例 .....    | 9  |
| 图 2.3  | 二维骨架表示 .....          | 11 |
| 图 2.4  | 三维骨架表示 .....          | 11 |
| 图 3.1  | HumanEva背景 .....      | 12 |
| 图 3.2  | HumanEva背景分割1 .....   | 13 |
| 图 3.3  | HumanEva背景分割2 .....   | 14 |
| 图 3.4  | 去除阴影 .....            | 14 |
| 图 3.5  | HumanEva背景分割3 .....   | 15 |
| 图 3.6  | HumanEva背景分割4 .....   | 15 |
| 图 3.7  | 背景分割对比 .....          | 16 |
| 图 3.8  | 背景分割错误 .....          | 16 |
| 图 3.9  | 图片比例影响示意 .....        | 19 |
| 图 3.10 | 图片固定比例示意 .....        | 19 |
| 图 3.11 | HOG特征子区域直方图表示 .....   | 20 |
| 图 3.12 | HOG特征 .....           | 20 |
| 图 3.13 | 皮肤检测 .....            | 21 |
| 图 5.1  | 单个视频逐帧误差 .....        | 29 |
| 图 5.2  | 其他数据测试示意1 .....       | 30 |
| 图 5.3  | 其他数据测试结果对比1 .....     | 30 |
| 图 5.4  | 其他数据测试示意2 .....       | 31 |
| 图 5.5  | 其他数据测试结果对比2 .....     | 31 |
| 图 6.1  | 错误的ground-truth ..... | 33 |

## 表格索引

|       |                                 |    |
|-------|---------------------------------|----|
| 表 2.1 | 常用姿态估计数据库汇总 .....               | 7  |
| 表 2.2 | HumanEva-I vs HumanEva-II ..... | 8  |
| 表 2.3 | HumanEva-I训练+验证帧数统计 .....       | 8  |
| 表 2.4 | [?][?]文章所用帧数统计 .....            | 10 |
| 表 2.5 | 本文所用帧数统计 .....                  | 10 |
| 表 2.6 | 关节点构成.....                      | 10 |
| 表 5.1 | 程序运行环境.....                     | 26 |
| 表 5.2 | HumanEva-I测试帧数统计 .....          | 27 |
| 表 5.3 | HumanEva-I有效测试帧数统计 .....        | 27 |
| 表 5.4 | TGP测试结果 .....                   | 28 |
| 表 5.5 | TGPKNN测试结果 .....                | 28 |
| 表 5.6 | 测试各步骤耗时统计 .....                 | 32 |

## 公式索引

|               |    |
|---------------|----|
| 公式 4-1 .....  | 22 |
| 公式 4-2 .....  | 22 |
| 公式 4-3 .....  | 22 |
| 公式 4-4 .....  | 23 |
| 公式 4-5 .....  | 23 |
| 公式 4-6 .....  | 23 |
| 公式 4-7 .....  | 23 |
| 公式 4-8 .....  | 23 |
| 公式 4-9 .....  | 24 |
| 公式 4-10 ..... | 24 |
| 公式 4-11 ..... | 24 |
| 公式 4-12 ..... | 24 |
| 公式 4-13 ..... | 25 |
| 公式 4-14 ..... | 25 |
| 公式 4-15 ..... | 25 |
| 公式 4-16 ..... | 25 |
| 公式 5-1 .....  | 27 |
| 公式 5-2 .....  | 27 |

## 致 谢

感谢导师戴琼海教授、刘烨斌老师对本人的热情指导，他们帮助我迈出了科研的第一步，做科研的方法将终身受益。

承蒙王雁刚师兄的指导和帮助，我坚定了研究方向，梳理了研究思路，在做不下去的时候，王雁刚师兄一次次给予了我前进的方向和成功的信心。

感谢和我一起讨论问题、交流毕业论文心得的吴蒙蒙、刘金林、胡雪梅等同学，是你们，让毕业设计有了更多的欢声笑语。

最后感谢柯家琪、张洋师兄对我撰写论文给予的帮助，感谢THUTHESIS，它的存在让我的论文写作轻松自在了许多，让我的论文格式规整漂亮了许多。

## 声 明

本人郑重声明：所呈交的学位论文，是本人在导师指导下，独立进行研究工作所取得的成果。尽我所知，除文中已经注明引用的内容外，本学位论文的研究成果不包含任何他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确方式标明。

签 名: \_\_\_\_\_ 日 期: \_\_\_\_\_

## 附录 A 外文资料翻译

### 一种用于视频中表情生成的数据驱动方法

李凯<sup>1,2</sup> 徐枫<sup>1</sup> 王珏<sup>3</sup> 戴琼海<sup>1</sup> 刘烨斌<sup>1</sup>

<sup>1</sup>清华大学自动化系

<sup>2</sup>清华大学深圳研究院 <sup>3</sup>Adobe 系统

**摘要:** 本文提出了一种方法来用一个人的面部表情视频对目标人脸合成真实的面部动画。不同于以往的面部动画方法，我们的系统利用了现有的目标人物的面部表情数据库，并最终通过从数据库中获取含有与输入相似的表情的帧来生成最终视频。为此我们开发了一种表情相似度度量来准确地测量两个视频帧的表情差异。为了加强时间相干性，我们的系统从相似度度量决定的候选帧中，利用最短路径算法来选择最优的图片。最后，我们的系统采用一种表情映射的方法来进一步减小输入和检索得到的帧之间的表情差异。实验结果显示我们的系统可以利用所提出的数据驱动方法生成高质量面部动画。

#### A.1 简介

性能驱动的面部动画在20世纪80年代就已经走红。它指的问题是：将面部表情从一个人映射到另一个，目标是使渲染的目标面部动画和原表情相比真实且一致。

尽管在过去几十年内已经取得了巨大进步，但这个问题依然未解决。之前的方法主要集中于表情的真实度，也就是说，使目标面部的渲染表情主观上接近输入面部的表情。另一方面，逼真的渲染很大程度上被忽视，先前的方法通常使用3D面部模型作为目标头像。目前尚不清楚在给出一个人的表情后，怎样为一个真实人物的面部渲染逼真的动画。此外，许多以前的方法严重依赖诸如在源面部上标记<sup>[1,17]</sup>，或精细的人机交互做跟踪<sup>[15,28]</sup>等额外信息。这些方法的应用范围和效率因此比较有限。

在本文中，我们旨在开发一种自动化系统将一个脸部视频的表情转化到另一个人上，从而产生目标人物的自然表情的视频。受到最近在封闭人脸实现<sup>[8]</sup>和人体运动动画<sup>[29]</sup>上的数据驱动方法的启发，我们的系统基于现有的目标

人物的表情数据库来实现目标。由于数据库提供了目标人脸在不同表情下的自然视频帧，我们可以利用这些做参考来渲染和输入表情匹配的视频。然而，这个任务并不简单，有如下挑战：

1. 怎样测量两个不同人物视频帧的表情相似度；
2. 怎样高效地搜索数据库以确保生成的视频不仅接近输入表情，也具有时间相干性；
3. 由于数据库大小有限，不能覆盖所有输入的表情，怎样进一步调整目标视频帧的表情来提高表情准确性。

我们的系统采用一套技术来解决这些挑战。具体来说，我们提出一个新颖的测量视频中不同人物表情的相似度。为了在时间相干性和表情匹配精度上平衡，我们先从数据库找到K近邻作为每个输入帧的候选，用优化方法来求得最优输出序列。最后，考虑到每个输入和检索帧的细微表情差异，我们提出一种表情转移方法，用这个结果来进一步细化获得的帧的表情。实验结果显示我们的系统能合成时间上连贯且与输入匹配的逼真的面部表情动画。

## A.2 相关工作

这项工作与以前在面部表情匹配、面部表情重定向（映射）、视频到视频合成方面研究工作相关。

### A.2.1 面部表情匹配

我们的要求是找到最相似的苗青，而不是把面部运动分类到具体、事先定义的类别。在表情识别社区中使用的特征，比如Gabor小波<sup>[19]</sup>, LBP<sup>[21]</sup>和FACS<sup>[9]</sup>，或许能提供一种替代方法。然而，他们经常没能考虑身份的差异。比如，一个有胡子的笑脸与没有胡子的笑脸，就LBP特征而言是不同的。只有拥有足够的有胡子和没胡子的训练样本，分类器才能辨别两个笑脸是一样的。此外，这些度量也许不能推断出一个连续的实值距离测量，这意味着他们经常不足以精确地捕捉细微的表情差异。比如CERT<sup>[14]</sup>，仅仅能较好识别峰值表情的活动单元。还不清楚它分辨细微的AU运动能有多好。相反，这两个主要问题在我们提出的表情相似度测量中不存在。

### A.2.2 3D基于模型的面部表情重定位

在表情建模和重定位方面已有大量工作。在基于PCA的模型中，比如AAM<sup>[4]</sup>

/CLM<sup>[23]</sup>，3D形变模型<sup>[3]</sup>，多线性模型<sup>[7,25]</sup>，和变形模型<sup>[27]</sup>，通用基础通过保留主成分从大训练数据中学习来。他们以丢失精细的细节为代价努力换取鲁棒地跟踪所有表情。然而在我们的精炼方法中，在两个相似表情的图片中光流可以更好地捕捉细节表情的不同，从而获得更精确地重定位结果。有特殊特征的形状融合模型为实时动画而建立<sup>[26]</sup>。然而，形状融合变形器的数量是模型覆盖度和总适应性之间的矛盾。其他系统<sup>[2,20,22]</sup>努力建立纹理逼真的3D面部模型。然而，获得完全纹理的3D模型不容易。

### A.2.3 基于图片的表情映射

一些人脸合成系统直接在2D图片上操作来实现表情转移。Williams的系统<sup>[28]</sup>从源和目标图片提取面部特征，用特征差异引导扭曲。Liu等人<sup>[17]</sup>提出Expression Ratio Image (ERI)通过捕捉光照变化来加强表情映射。Zhang *et al.*<sup>[30]</sup>用几何元通过融合样例脸部图片来计算每个图片子区域的纹理。然而，这些方法通常不能处理两幅图片间大的拓扑变化。我们的方法通过从拥有和输入相似的表情的数据库中获得目标脸部克服了这个局限。此外，这些方法通常是劳动密集型的。

### A.2.4 视频到视频的合成

我们的工作涉及到之前视频到视频合成系统。和我们的目标相似，Kemelmacher-Shlizerman等人<sup>[10]</sup>利用数据库，在一个人联视频驱动下合成一个目标人物的面部视频。然而，他们的系统主要着眼于测量面部表情的相似度。最终视频只是简单地由独立的最相似的图片连接合成，这可能时间上不一致。视频面部替代系统<sup>[7]</sup>在保证时空一致性的基础上用源视频的面部代替目标视频中的人脸。然而，它假设输入和目标视频之间粗糙的语义对应和大致相近。

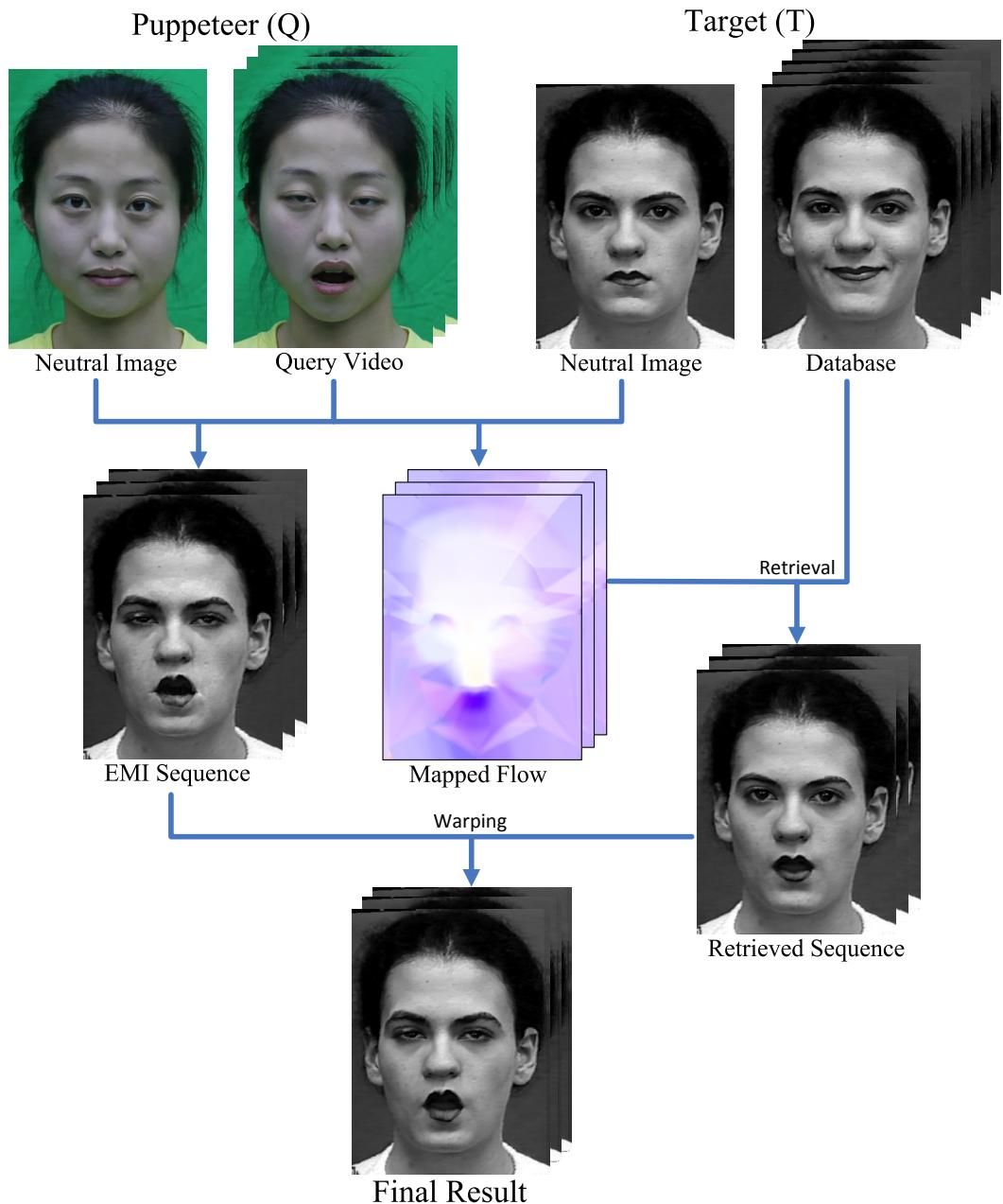


图 A.1 系统概览. 首先将每个查询帧和它的中性面部之间的光溜映射到目标人，用来从数据库中检索。同时，目标人的中性图像被扭曲以生成含有查询表情的EMI序列。最后，检索序列用EMI 精炼，来生成最终结果。

## A.3 系统概览

图 A.1展示了我们系统的概况。要为目标人生成逼真的表情，我们首先捕捉一段这个人展示基本表情的视频，比如生气、恐惧、惊奇、伤心、高兴、厌恶。有了另一个我们叫做人偶师(puppeteer)的人的面部表情，我们的方法尝试利用目标人的数据合成同样的表情。

具体而言，对于每个输入帧，我们使用第 A.4.1章描述的基于光流相似度度量方法查询数据库获得 $k$ 个与输入帧有最相近表情的目标人的视频帧。正如在第 A.4.2 章描述的那样，我们把这个任务认为是和最短路问题一样找最优连续帧，而不是像Kemelmacher-Shlizerman等人<sup>[10]</sup> 直接用最相似的帧生成一个匹配序列。获得的序列包含和人偶师相似且时间一致的表情。

然而，由于数据库大小有限，为每个输入帧找到一个完美的表情匹配几乎是不可能的，更何况，一些表情的人偶师具有独特的特点。为了考虑再输入和检索帧之间细微的表情差异，我们用一个表情映射技术来生成另一个候选面部，我们称之为EMI图像，如第 A.4.3章描述那样。EMI图片通常有比检索帧有更精确的面部表情，但她的面部外观可能有重大瑕疵。在最后一步，我们把EMI图片和检索帧结合起来从而生成有精确表情和逼真外观的最终输出帧，如第 A.4.3章所述。

## A.4 算法

### A.4.1 表情相似度度量

给出人偶师的面部图片 $Q_e$ ，我们的系统试图从目标人的数据库中找到对应面部图片 $T_e$ ，这幅图片有和 $Q_e$ 最接近的面部表情. 为此我们需要能准确测量 $Q_e$ 和 $T_e$ 之间表情差异的面部相似度度量，同时忽视两幅图片的外表差异。

为了找到这样一个度量，我们的系统使用人偶师和目标人的中性面，分别记作 $Q_n$  和 $T_n$ 。当我们建立数据库的时候 $T_n$  只需要标识一次，我们假设 $Q_n$  在输入视频中由用户标记。为了说明人偶师的面部如何从 $Q_n$  到 $Q_e$ 变化，我们能计算两幅图片中的光流场<sup>[16]</sup>记作 $\mathbf{F}_{Q_n \rightarrow Q_e} \in \mathbb{R}^{m \times 2}$ ，这里 $m$ 表示 $Q_n$ 中的所有人脸像素。为了从光场中去除全局头部运动，我们用不随表情变化的鼻子区域来估计2D相似度变化，在计算表情差异前先把 $Q_e$ 和 $Q_n$ 对齐。我们也用脸的宽度归

一化光流。类似的， $T_n$ 和 $T_e$ 之间的光流场 $\mathbf{F}_{T_n \rightarrow T_e} \in \mathbb{R}^{n \times 2}$ ，其中 $n \neq m$ ，也可以计算。然而，由于身份/外观不同我们不能直接对比这两个光场。为了在两个光场

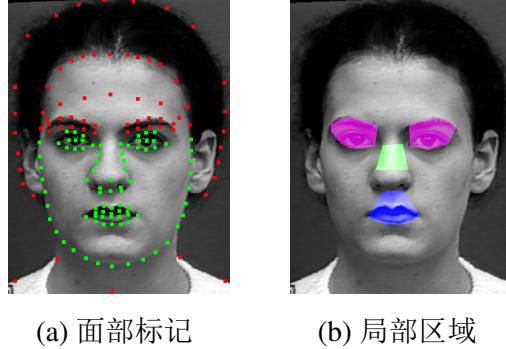


图 A.2 中性脸初始化(a) 绿色是ASM的标记，红色是手工标记(b) 眼睛、嘴巴、鼻子区域分别用品红、蓝色、绿色标记。

之间建立精确的对应，我们首先使用Active Shape Model (ASM)<sup>[5]</sup>只检测在中性面 $Q_n$ 和 $T_n$ 的面部标志物，该方法对于中性表情的正面人脸很有效。然而它无法覆盖我们算法在之后几步中需要的所有面部。因此我们在两个中性面上手工标注标志点，如图 ?? 所示。然后我们用Delauney三角网标出 $Q_n$  和 $T_n$  中的脸部区域，这引出一个只能像素注册函数 $g : Q_n \rightarrow T_n$ 。此外，由于两个身份之间的语义对应应该对不同的面部表情具有不变性，可以合理假设 $g' : Q_e \rightarrow T_e$ 两个表情图片的注册函数近似和 $g : Q_n \rightarrow T_n$ 相同。有了注册函数，对于一个点 $\vec{d} \in Q_n$ 转移到 $\vec{d}' \in Q_e$ ，它在 $T_n$ 对应的光流向量按如下计算：

$$\Delta \vec{b} = g(\vec{d}') - g(\vec{d}) \mathbf{f}_i \quad (\text{A-1})$$

其中 $\vec{b} = g(\vec{d})$ 是 $T_n$ 上 $\vec{d}$ 的对应点。

通过在 $Q_n$ 上的所有面部像素应用这个映射，我们获取了一种映射好的光流场 $\mathbf{F}'_{Q_n \rightarrow Q_e}$ ，可以通过与 $\mathbf{F}'_{Q_n \rightarrow Q_e}$ 对比测量两个表情有多接近。以往工作指出<sup>[10]</sup>，表情差异的主要来源是眼睛和嘴巴区域，因此我们只用这些区域的像素来计算表情相似度（参见图 ??）。直接的方法是通过绝对的光流差计算 $Q_e$  和 $T_e$ 之间的表情差异：

$$d_e(Q_e, T_e) = \alpha_e \sum_{i \in \text{eye}} \|\mathbf{F}'_{Q_n \rightarrow Q_e, i} - \mathbf{F}_{T_n \rightarrow T_e, i}\| + \alpha_m \sum_{i \in \text{mouth}} \|\mathbf{F}'_{Q_n \rightarrow Q_e, i} - \mathbf{F}_{T_n \rightarrow T_e, i}\| \mathbf{f}_i \quad (\text{A-2})$$

这里下标*i*表示光流矩阵 $\mathbf{F}$ 中第*i*行。 $\alpha_{\{e,m\}}$ 分别是眼睛和嘴巴区域的权重。

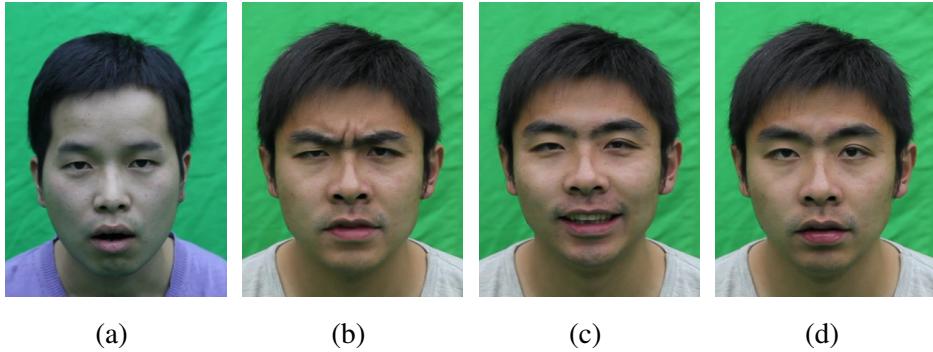


图 A.3 表情度量对比。 (a) 查询帧。 (b) 由LBP方法得到的最相似表情，它有一个不理想的皱眉。 (c) 由等式 A-2获得的最相似表情，有一个笑容而不是惊奇。 (d) 由等式 A-4获得的最相似表情，和查询帧匹配很好

等式 A-2中的距离度量在我们大多数实验中都很成功，但我们发现会偶尔产生如图 A.3所示的错误。这是因为在等式 A-2中，我们仅考虑了 $\mathbf{F}'_{Q_n \rightarrow Q_e} - \mathbf{F}_{T_n \rightarrow T_e}$ 的光流差异的大小。但光流的方向通常包含更多有关表情的重要信息。比如笑脸通常与嘴角的上翘相关，而哭脸和嘴角下弯相关。这表明在 $Q_e$ 中的表情与 $T_e$ 中的很不一样，如果 $\mathbf{F}'_{Q_n \rightarrow Q_e}$ 到 $\mathbf{F}_{T_n \rightarrow T_e}$ 的差异方向很不一样，即便差异大小很小。有了这个发现，我们重新设计表情距离度量如下：

$$d_b(\vec{u}, \vec{v}) = \beta_m |\vec{u} - \vec{v}| + \beta_o (-\vec{u} \cdot \vec{v} + |\vec{u}| |\vec{v}|), \quad (\text{A-3})$$

$$\begin{aligned} d_e(Q_e, T_e) = & \alpha_e \sum_{i \in \text{eye}} d_b(F'_{Q_n \rightarrow Q_e, i}, F_{T_n \rightarrow T_e, i}) \\ & + \alpha_m \sum_{i \in \text{mouth}} d_b(F'_{Q_n \rightarrow Q_e, i}, F_{T_n \rightarrow T_e, i}), \end{aligned} \quad (\text{A-4})$$

这里 $\beta_{\{m,o\}} \in [0, 1]$ 分别是大小和方向的权重，且 $\beta_m + \beta_o = 1$ 。当 $\beta_o$ 等于零，等式 A-4简化为等式 A-2。在等式 A-3 中的偏移量 $|\vec{u}| |\vec{v}|$ 确保方向项非负。注意此距离度量不保证对称性和三角不等式。为使之更有数学味，可以计算反向距离 $d_e(T_e, Q_e)$ ，用二者平均值作为最终距离度量。但实际上我们发现这并无必要，因为 $d_e(Q_e, T_e)$ 已很好描述不同身份两张图片的表情差异，对我们的应用而言已足够。

## A.4.2 基于检索的视频合成

使用如上定义的相似性度量，一个视频合成的简单方法是为每个输入帧，找到其在数据库中的最近邻表清，并叠在一起，以形成最终的输出视频。然而，我们发现这种方法并不很好，在最终视频中面部表情的时间相干性没有得到很好的保持，并且最终的视频经常出现抖动。补充材料包含了说明此问题的视频。我们的系统采用了一些额外的技术来解决的时间一致性问题，我们将在本小节详细介绍。

### A.4.2.1 结合表情速度

首先，在公式 A-4 中定义的距离度量只考虑了两个面部的表情相似度。然而在视频中，我们需要关心在每一帧表情变化的速度。最相似的帧应该是表情及其变化速度都与查询帧相符的。为了保证表情速度，我们在视频序列中简单地计算另一个在当前帧和下一帧之间的光流。记  $Q_e^{(q)}$  为第  $q$  个查询帧，表情速度计算如下：

$$d\mathbf{F}_{Q_e^{(q)}} = \mathbf{F}_{Q_e^{(q)} \rightarrow Q_e^{(q+1)}}. \quad (\text{A-5})$$

类似的，对于在数据库中的帧  $T_e^{(t)}$ ，我们计算表情速度为  $d\mathbf{F}_{T_e^{(t)}}$ 。同样，由于  $Q_e^{(q)}$  和  $T_e^{(t)}$  身份和表情差异，直接计算  $d\mathbf{F}_{Q_e^{(q)}}$  和  $d\mathbf{F}_{T_e^{(t)}}$  距离并不好。需要扭曲表情速度光流场来去除身份差异，如我们在第 A.4.1 章所做。我们还需要扭曲两者的速度光流场来把他们映射到中性表情，以去除他们的表情差异。

具体说，对于数据库帧，我们把  $\mathbf{F}_{T_n \rightarrow T_e^{(t)}}$  的反向光流用于  $d\mathbf{F}_{T_e^{(t)}}$ ，导出与中性表情  $T_n$  相符的扭曲的表情速度流  $d\mathbf{F}'_{T_e^{(t)}}$ 。对于查询帧  $Q_e^{(q)}$ ，我们把公式 A-4 计算的反向光流  $\mathbf{F}'_{Q_n \rightarrow Q_e^{(q)}}$  用于  $d\mathbf{F}_{Q_e^{(q)}}$ ，导出与中性表情  $T_n$  相符的扭曲的表情速度流  $d\mathbf{F}'_{Q_e^{(q)}}$ 。最后， $Q_e^{(q)}$  和  $T_e^{(t)}$  的表情速度差异计算如下：

$$\begin{aligned} d_v(Q_e^{(q)}, T_e^{(t)}) = & \alpha_e \sum_{i \in \text{eye}} d_b(d\mathbf{F}'_{Q_e^{(q)}, i}, d\mathbf{F}'_{T_e^{(t)}, i}) \\ & + \alpha_m \sum_{i \in \text{mouth}} d_b(d\mathbf{F}'_{Q_e^{(q)}, i}, d\mathbf{F}'_{T_e^{(t)}, i}), \end{aligned} \quad (\text{A-6})$$

此处函数  $d_b(\cdot, \cdot)$  在公式 A-3 中定义。结合公式 A-4 和公式 A-6，最终的视频表情距离度量定义如下：

$$\mathcal{D}(Q_e^{(q)}, T_e^{(t)}) = \gamma_e d_e(Q_e^{(q)}, T_e^{(t)}) + \gamma_v d_v(Q_e^{(q)}, T_e^{(t)}), \quad (\text{A-7})$$

这里  $\gamma_{\{e,v\}} \in [0, 1]$  分别是表情距离和表情速度距离的权重，满足  $\gamma_e + \gamma_v = 1$ 。

图 A.4 的例子表明，当表情很微妙，在度量中结合表情速度能帮助系统更好的捕捉表情变化。



图 A.4 使用表情速度的重要性阐述。(a) 红色表示带表情速度的当前查询帧。(b) 微笑的下一查询帧。(c) 由公式 A-4选择的最相似帧，有细淡淡的忧伤。(d)由公式 A-7选择的最相似帧，有正确的微笑。

#### A.4.2.2 基于优化的检索

改进的表情相似度度量不能独立完整地解决时间一致性的问题。因此我们的系统使用就与优化的检索方法进一步提升合成序列的时间一致性。

对于每个查询帧，我们首先使用公式 A-7定义的完整的距离度量从数据库不是抽取一个，而是  $k$  近邻，我们称之为候选帧。在每帧的时间戳放置一列  $k$  个候选帧，我们建立如图 A.5 所示的有向无环图。有向边只连接邻近候选帧。记  $V_i^{(q)}$  为时刻  $q$  的第  $i$  候选帧。我们定义有向弧  $r = (V_i^{(q)}, V_j^{(q+1)})$  的长度为：

$$\begin{aligned} \mathcal{L}(r) = & \mathcal{D}(V_i^{(q)}, Q_e^{(q)}) + \mathcal{D}(V_j^{(q+1)}, Q_e^{(q+1)}) \\ & + \lambda \exp(-(\mathcal{T}(V_j^{(q+1)}) - \mathcal{T}(V_i^{(q)}) - \mu)^2 / \sigma^2), \end{aligned} \quad (\text{A-8})$$

这里  $\mathcal{T}(\cdot)$  是输入帧的时间戳。通过最小化相邻帧的时间差，公式 A-8 的最后一项鼓励数据库中的连续帧选为匹配帧，以保证时间一致性。时间尺度变量  $\mu$  用于补偿查询和数据库序列之间的运动速度差。当查询帧和数据库序列运动速度大致相同， $\mu$  设为 1，当查询帧运动速度比数据库序列快则设为一个大数，否则相反。 $\sigma$  是带宽， $\lambda$  是时间项的权重。

在公式 A-8中的时间项是L2范数，允许晓得时间变化，但对大的变化有严惩。由于小的变化被允许，它也允许某些时间尺度的变化。这在我们的查询序列1中被证明，如图 A.7所示，它包含缓慢嘴巴张开的表情和快速撅嘴的表情。我们的系统对这二者都处理很好。此外， $\mu$ 也能在视频的不同时间根据查询运动的速度自动调整。

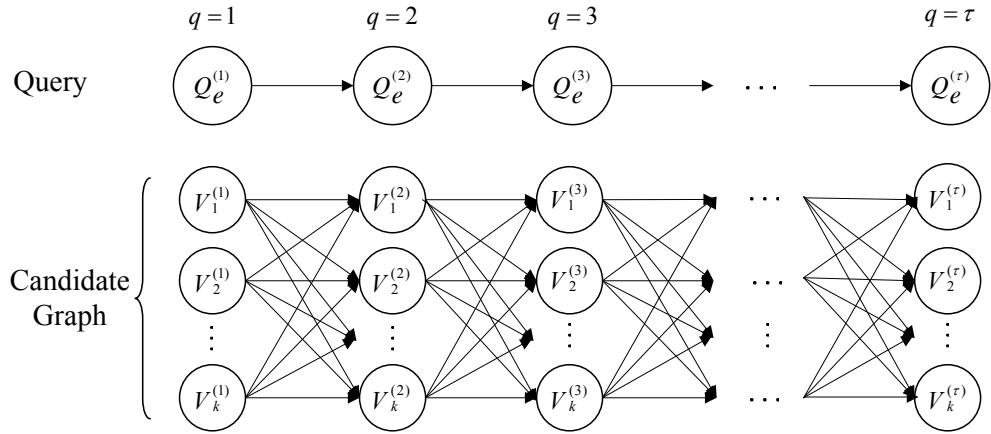


图 A.5 用于检索的有向图

令 $\mathcal{P}_{V_i^{(1)} \rightarrow V_j^{(\tau)}}$ 为连接起始节点 $V_i^{(1)}$ 和终止节点 $V_j^{(\tau)}$ 的路径，这里 $i, j \in \{1, 2, \dots, k\}$ 。在所有从第一到最后帧的可能路径中，我们找到最短路。优化目标被正式定义如下：

$$\mathcal{P}_{opt} = \arg \min_{i,j} \arg \min_{\mathcal{P}_{V_i^{(1)} \rightarrow V_j^{(\tau)}}} \sum_{r \in \mathcal{P}_{V_i^{(1)} \rightarrow V_j^{(\tau)}}} \mathcal{L}(r). \quad (\text{A-9})$$

该问题用Fibonacci堆<sup>[6]</sup>的Dijkstra算法很好解决。连接最优路径 $\mathcal{P}_{opt}$ 的所有帧构成检索序列。

我们的基于优化的方法借鉴了以往用于生成时间一致动画的工作<sup>[11-13]</sup>。我们的方法结合了时间一致性和语义对应。

#### A.4.3 表情精炼

前面的检索结果有两个缺点。首先，由于我们的数据库的大小是有限的，检索的帧可能不包含和输入序列完全相同的表情。其次，数据库中的帧没完全对齐，所以检索序列包含了一些少量的时间抖动。要删除这些错误，我们的系统采用了额外的表情细化组成部分。

表情精炼的主要思路是给定  $Q_n$  和  $Q_e$ , 中性帧和人偶师的表情帧, 还有  $T_n$ , 目标人的中性帧, 我们可以直接从两个源图片中提取表情, 并映射到  $T_n$  来合成新的面部  $T_{Q_e}$ 。该合成的脸部, 我们称之为expression mapping image (EMI), 有所需的表情, 但是也许没有逼真的纹理, 尤其当  $Q_n$  和  $Q_e$  表情差异很大的时候。另一方面, 检索帧有真实的外表, 但表情和  $Q_e$  并不完美匹配。结合EMI和检索帧, 我们可以生成最终图片, 有逼真的外表和精确匹配的表情。

具体说, 我们首先通过把  $Q_n$  和  $Q_e$  的光流转移到目标帧来扭曲  $T_n$ 。给定点  $\vec{d} \in Q_n$ ,  $\vec{d}' \in Q_e$  和  $\vec{b} \in T_n$  ( $\vec{b} = g(\vec{d})$  如公式 A-1), 我们计算点  $\vec{b}' \in T_{Q_e}$  的颜色如下:

$$c_{\vec{b}'} = c_{\vec{b}} \frac{c_{\vec{d}'}}{c_{\vec{d}}}, \quad (\text{A-10})$$

这里  $c_{\{\vec{d}, \vec{d}', \vec{b}, \vec{b}'\}}$  分别是点  $\vec{d}$ ,  $\vec{d}'$ ,  $\vec{b}$  and  $\vec{b}'$  的颜色值 (我们系统中使用YCrCb颜色空间)。这里我们用比值  $c_{\vec{d}'} / c_{\vec{d}}$  来表示颜色  $c_{\vec{b}'}$  如ERI方法<sup>[17]</sup>所做。在实际实施中, 为了避免  $\vec{b}'$  有非整数坐标, 我们用反向计算表情映射我们从一个整数像素  $\vec{b}' \in T_{Q_e}$  开始, 根据  $\vec{d}' = g^{-1}(\vec{b}')$  找到它的对应点  $\vec{d}' \in Q_e$ 。通过计算  $F_{Q_e \rightarrow Q_n}$  获得的光流  $\Delta \vec{d}'$  给出点  $\vec{d}$  的坐标。通过注册函数, 我们获得点  $\vec{b} = g(\vec{d}) \in Q_n$  的位置。然后点  $\vec{b}'$  的颜色可以依据公式A-10 计算。

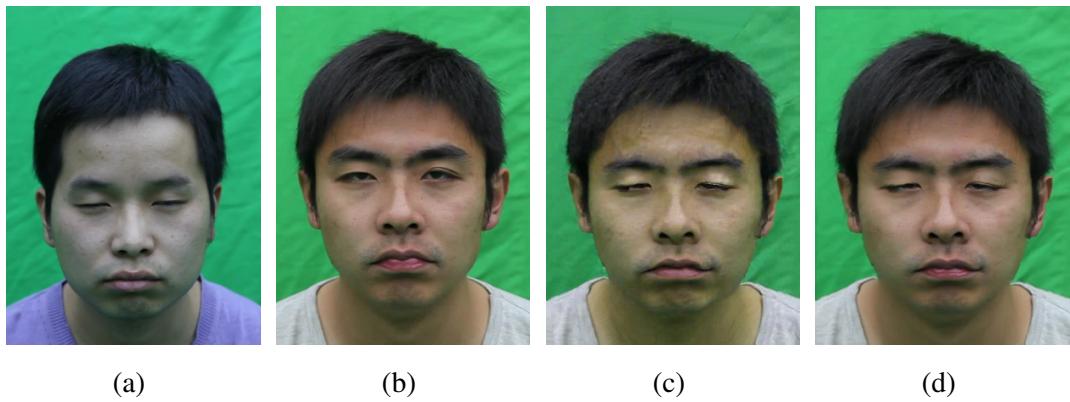


图 A.6 表情精炼。 (a) 查询帧。 (b) 检索帧。 (c) 表情映射图片。 (d) 最终结果。

最终, 我们计算每一帧时的EMI和检索结果的光流, 并利用光流扭曲检索图像至EMI。如图 A.6所示, 最终合成结果不只有从检索帧继承来的真实的外表, 还有由EMI图像继承来的和查询帧匹配的正确表情。

## A.5 结论和讨论

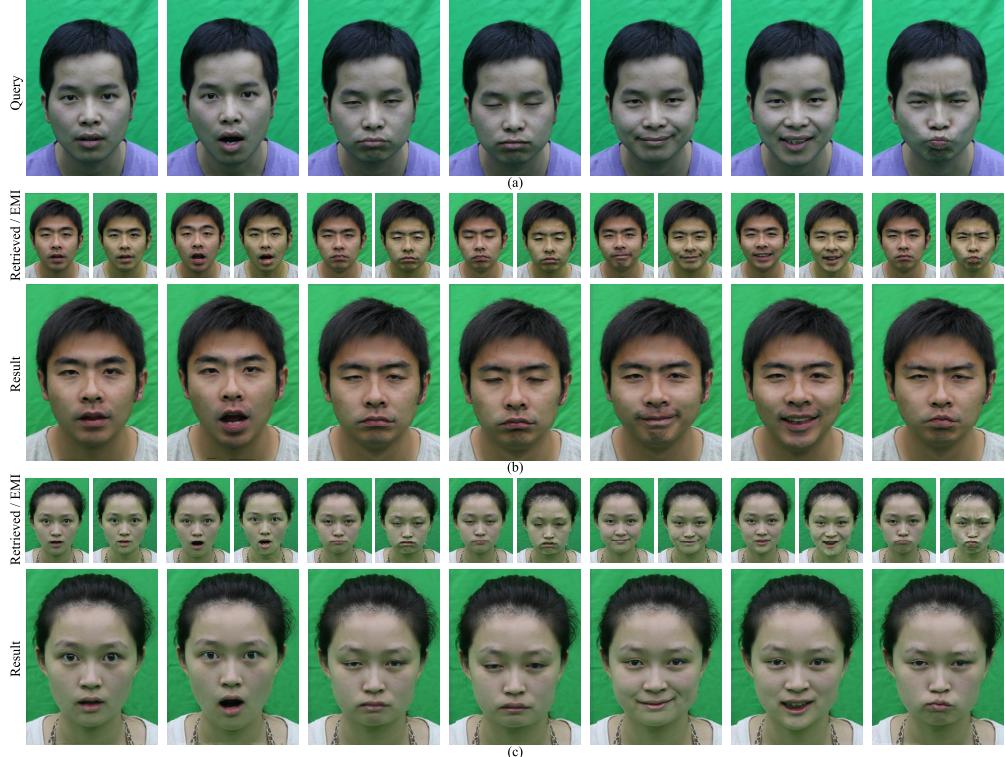


图 A.7 在目标  $T_1$  和  $T_2$  上运行的查询序列的结果。(a) 普通查询帧(b) (第一行) 检索和EMI帧 (分别是左和右); (第二行) 目标  $T_1$  的最终合成帧。 (c) 目标  $T_2$  的检索、EMI帧和Retrieved最终合成帧。

### A.5.1 实验

我们评估了三个数据库系统。其中两个是我们采集的，这两个主题一个是男性，一个女性，他们被要求表现6个基本表情：愤怒，厌恶，惊讶，恐惧，快乐和悲伤。每个数据库都是25fps拍摄，包含约1500帧。第三个数据库是从Extended Cohn-Kanade Dataset (CK+)<sup>[18]</sup>得到的S130。S130 中的11个短序列(共220帧)构成了该表情数据库。在所有试验中，我们算法的参数固定如下： $\alpha_e = \frac{0.6}{n_e}$ ,  $\alpha_m = \frac{0.4}{n_m}$ ,  $\beta_m = 0.9$ ,  $\beta_o = 0.1$ ,  $\gamma_e = 0.9$ ,  $\gamma_v = 0.1$ ,  $k = 20$ ,  $\lambda = 0.1$ ,  $\mu = 1$ ,  $\sigma = 2$ ，其中  $n_e$  和  $n_m$  分别是目标中性面眼镜和嘴巴区域的像素数量。

### A.5.2 结果和评估

图 A.7 显示了目标  $T_1$  (男性) 和  $T_2$  (女性) 由输入序列驱动的合成结果。注意我们的系统不仅在诸如笑脸和惊奇等表情在数据库中时可以合成逼真的表情，而且在诸如撅嘴且双眼紧闭的表情在数据库中没有的时候也能合成新表情。同时，最终合成的视频也是时间相干的。图 A.8 展示了由另一序列驱动，从来自 CK+ 数据库的 S130 中合成的结果。结果表明即使只是一个小数据库我们的系统仍然效果很好。可以在补充材料中找到包括额外的快速说话重定位结果的完整的视频序列。为了评价我们的合成结果，我们进行了包含 34 个参与者的用户研究。每位参与者都被展示了四个视频，分别由 LBP 特征查询方法<sup>[10]</sup>，在第 A.4.3 章介绍的 EMI 方法，我们的检索策略和我们的整套算法一帧一帧查询获得。每个视频并排展示了查询和结果。在实验中，参与者被要求根据表情的真实性和一致性评价在每个结果中表情的好坏，分数从 5 (非常好) 到 0 (一点也不好)。表 A.1 显示了 3 个目标的平均分。参与者发现我们的最终结果是最好的且我们的检索策略优于在<sup>[10]</sup>提出的方法。

|                            | $T_1$       | $T_2$       | S130        |
|----------------------------|-------------|-------------|-------------|
| 基于 LBP 的检索 <sup>[10]</sup> | 1.20        | 1.50        | 1.38        |
| 我们的检索                      | 2.49        | 3.00        | 2.56        |
| EMI                        | 2.89        | 1.91        | 3.35        |
| 我们的整套系统                    | <b>4.02</b> | <b>4.56</b> | <b>4.08</b> |
| $p$ 值                      | 0.002       | 0.005       | 0.0001      |

表 A.1 用户研究结果。该结果在统计上是有意义的，使用了单变量方差分析， $p$ -value < 0.01。

### A.5.3 局限性

我们目前的系统只针对正面脸部表情合成设计。可能会扩展系统在大旋转角下运行，通过稀疏相机阵列采集数据。如此我们需要估计表情和输入帧的 3D 脸部。此外，需要视图变形技术<sup>[24]</sup>来在不同视角查看脸部，以在所需姿态生成面部表情。

另一个限制是，当表情很极端，传统的光溜方法无法精确捕捉表情差异。除了调查更好的脸部光流技术，另一个解决方案是为每个特征使用多个预先对其的脸部图像而不只是在我们现有系统中使用单一中性面。



图 A.8 在来自CK+的S130中由查询序列2获得的结果。（顶部）查询帧。（中部）检索帧和EMI帧（分别是左和右）。（底部）最终合成帧。

## A.6 结论

我们提出了一种数据驱动的方法来用一个人的面部表情视频对目标人脸合成真实的面部动画。我们的系统采用了新颖的时空表情距离度量，可以准确地测量视频中不同的人相似的表情。我们也提出了最短路径优化的检索策略来平衡在最终视频的表情相似性和时间连续性。和那些直接表情映射相比，通过变换检索到的视频帧进一步改善了表情相似性。用户研究结果表明，我们的系统可以产生很高的保真度和时间上一致的面部动画。

## 致谢

作者要感谢和王瑞平，邓岳，索津莉的讨论，评审和领导建设性的意见。这项研究由国家自然科学基金项目支持（第61035002号，第61073072号，第60933006号）。

## 参考文献

- [1] T. Beier and S. Neely. Feature-based image metamorphosis. In *SIGGRAPH*, pages 35–42, 1992.
- [2] V. Blanz, C. Basso, T. Poggio, and T. Vetter. Reanimating faces in images and video. *Computer Graphics Forum*, 22(3):641–650, 2003.
- [3] V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. In *SIGGRAPH*, pages 187–194, 1999.
- [4] T. Cootes, G. Edwards, and C. Taylor. Active appearance models. *IEEE TPAMI*, 23(6):681–685, 2001.
- [5] T. Cootes, C. Taylor, D. Cooper, and J. Graham. Active shape models-their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, 1995.
- [6] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms*. MIT press, Cambridge, 2009.
- [7] K. Dale, K. Sunkavalli, M. K. Johnson, D. Vlasic, W. Matusik, and H. Pfister. Video face replacement. *ACM Trans. Graph.*, pages 130:1–130:10, 2011.
- [8] Y. Deng, Q. Dai, and Z. Zhang. Graph laplace for occluded face completion and recognition. *IEEE Trans. Image Process.*, 20(8):2329 –2338, 2011.
- [9] P. Ekman and W. V. Friesen. *Facial action coding system: a technique for the measurement of facial movement*. Consulting Psychologists Press, Palo Alto, 1978.
- [10] I. Kemelmacher-Shlizerman, A. Sankar, E. Shechtman, and S. M. Seitz. Being john malkovich. In *ECCV*, pages 341–353, 2010.
- [11] I. Kemelmacher-Shlizerman, E. Shechtman, R. Garg, and S. M. Seitz. Exploring photobios. *ACM Trans. Graphics*, pages 61:1–61:10, 2011.
- [12] L. Kovar and M. Gleicher. Flexible automatic motion blending with registration curves. In *SCA*, pages 214–224, 2003.
- [13] Z. Li, N. Snavely, B. Curless, and S. M. Seitz. Spacetime faces: high resolution capture for modeling and animation. *ACM Trans. Graph.*, 23(3):548–558, 2004.
- [14] G. Littlewort, J. Whitehill, T. Wu, I. Fasel, M. Frank, J. Movellan, and M. Bartlett. The computer expression recognition toolbox (cert). In *FG*, pages 298–305, 2011.
- [15] P. Litwinowicz and L. Williams. Animating images with drawings. In *SIGGRAPH*, pages 409–412, 1994.
- [16] C. Liu. *Beyond Pixels: Exploring New Representations and Applications for Motion Analysis*. PhD thesis, MIT, 2009.

- [17] Z. Liu, Y. Shan, and Z. Zhang. Expressive expression mapping with ratio images. In *SIGGRAPH*, pages 271–276, 2001.
- [18] P. Lucey, J. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews. The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In *CVPR Workshops*, pages 94–101, 2010.
- [19] M. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba. Coding facial expressions with gabor wavelets. In *FG*, pages 200–205, 1998.
- [20] J.-y. Noh and U. Neumann. Expression cloning. In *SIGGRAPH*, pages 277–288, 2001.
- [21] T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE TPAMI*, 24(7):971–987, 2002.
- [22] F. Pighin, J. Hecker, D. Lischinski, R. Szeliski, and D. H. Salesin. Synthesizing realistic facial expressions from photographs. In *SIGGRAPH*, pages 75–84, 1998.
- [23] J. M. Saragih, S. Lucey, and J. F. Cohn. Real-time avatar animation from a single image. In *FG*, pages 117–124, 2011.
- [24] S. M. Seitz and C. R. Dyer. View morphing. In *SIGGRAPH*, pages 21–30, 1996.
- [25] D. Vlasic, M. Brand, H. Pfister, and J. Popović. Face transfer with multilinear models. *ACM Trans. Graph.*, 24(3):426–433, 2005.
- [26] T. Weise, S. Bouaziz, H. Li, and M. Pauly. Realtime performance-based facial animation. *ACM Trans. Graph.*, pages 77:1–77:10, 2011.
- [27] T. Weise, H. Li, L. Van Gool, and M. Pauly. Face/off: live facial puppetry. In *SCA*, pages 7–16, 2009.
- [28] L. Williams. Performance-driven facial animation. In *SIGGRAPH*, pages 235–242, 1990.
- [29] F. Xu, Y. Liu, C. Stoll, J. Tompkin, G. Bharaj, Q. Dai, H.-P. Seidel, J. Kautz, and C. Theobalt. Video-based characters: creating new human performances from a multi-view video database. *ACM Trans. Graph.*, pages 32:1–32:10, 2011.
- [30] Q. Zhang, Z. Liu, B. Quo, D. Terzopoulos, and H.-Y. Shum. Geometry-driven photorealistic facial expression synthesis. *IEEE TVCG*, 12(1):48–60, 2006.