目 录

摘要

近年来，随着计算机等技术的飞速发展，运动捕获数据在动画界、电影界的广泛应用，而通常情况下，运动捕获操作复杂，成本较高，而且通过专业演员一次性捕获的运动序列较长，捕捉的动作数据需要大量的后期处理，运动数据的分割、检索、编辑与合成成为了近年来研究的一个热点。

本文研究人体运动数据分割算法，首先对现有分割算法主成分分析和高斯混合模型方法进行研究，实现了对动作数据的分割。主成分分析法依据简单的动作与复杂的动作具有不同的维度，及有不同的主成分，本文将高维动作数据投影到固定维度的空间，根据平均投影误差变化导数进行分割点判断；高斯混合模型依据不同的动作具有不同的聚类，对动作数据进行高斯聚类，在不属于同一类的两帧之间进行动作分割。

实际动作数据具有周期性和重复性，如在做动作时，演员会对某个动作重复做多遍，一个动作序列中可能包含多个相同的动作片段，这样传统的分割方法只是一味的把动作片段分割开来，不能有效的将动作进行分类分割，以便于后期的存储和应用。本文在主成分分析法的基础上，依据不同动作其内在信息的不同和同一种动作之间动作信息的相似性，提出了两种基本的分类分割方法，并对方法进行了验证。

首先使用基于投影误差变化的方法，在固定主成分个数的情况下，本文根据不同种类动作之间的转换误差是较大的，对当前动作数据计算投影误差，与前一分割帧的误差进行比较判断是否出现新动作。

实验中发现基于投影误差的分类分割方法能够有效地将连续的相同动作片段进行分类，而对于同一动作片段不连续出现的情况时，该方法不能得到较好的结果。

考虑到同一动作的主成分元素组成是相似的，反映到人体上是同一动作片段的各个骨骼的位置是相似的，本文提出了基于主成分元素组成的运动数据分类分割方法，对于某一分割片段的帧进行主成分分析，得到每个主成分的子成分组成，对于不同的动作片段，以各个动作片段主成分之间的距离误差进行分割，得到了较好的实验结果。

关键词： 人体运动数据 数据分类分割 主成分分析 GMM

Abstract

Motion Capture(Mocap) Data has been widely used in animation industry and filmland,but the Mocap data itself is very diffcult to capture and the capture operation costs a lot. the post-process of Mocap data is also very hard for human.The segmentation,retrieval,edit and synthesis of the Mocap data has become the focus of research these years.

This paper returns to the segmentation of Mocap data,the recent segment algorithm,for example,the PCA-SVD and GMM,will be discussed first.The Principal Component Analysis(PCA) approach is based on the assumption that simple motion exhibit lower dimensionality than more complex motions. In this paper,we project the complex Mocap data onto a fixd r-dimensional hyperplane, check for a possible cut by computing the derivative of the projection error;the underlying assumption of Gaussian Mixture Model(GMM) is that different simple motions form separate clusters,we estimate the GMM parameters firstly, a change of behavior is then detected where 2 frames belong to different clusters.

The real action is likely to be cyclical and repeated,for example, a long sequence of mocap data often contains several same behaviours,thus the current algorithm cannot effectively get a better classification segmentation.In this paper,we present two solutions for classification segmentation based on the diversity and similarity of internal information between different and same behaviors.

One approach is based on the projection error,the transform error will be larger.we calculate the transform error with fixed number of principal component,compare to the previous segmented frame to check if there is a new behavior.

The segment algorithm based on the projection error proved to be efficient for the consecutive behaviors,however,when comes to inconsecutive ones,this approach seems to cannot see great performance.

Considering that the principal components is likely to be similar between two same behaviors,that means the position of the skeletons is similar.In this paper,we put forward an segment algorithm on the basis of the component of behaviors,calculating the distance between two segmented behaviors,judging if they are same behavior or not,and we get the relatively reasonable result.

key words: Mocap data classification segmentation PCA GMM

第一章　绪论

§1.1　运动捕获分割技术

人体运动捕获数据是一个连续的长序列，其中包含不同的动作。运动数据的分割技术是将其中不同类型的动作分割成为独立的运动片段，以便于存储和复用[1]。图1表示一个动作序列以及其分割示意图1.1，其中可能包含几种不同性质的运动，左边部分为走路，中间部分为跳跃，右边部分为走路，虚线部分表示分割位置。

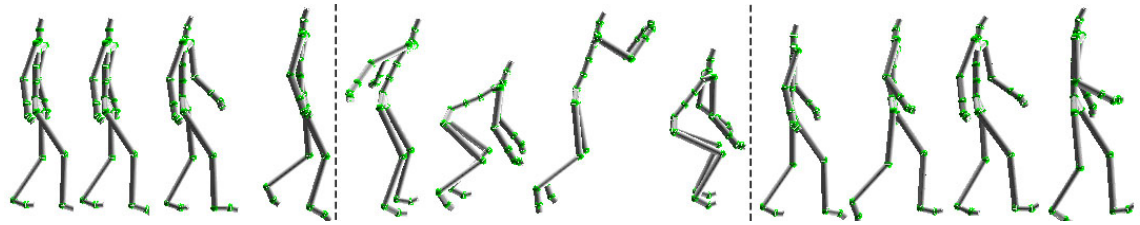

图 1.1: 运动捕获数据分割示意图

§1.2　运动捕获分割技术的发展

人体运动捕获数据的分割一直以来就是计算机动画学者所研究的一个热点，其中主要研究的内容便是对三维人体运动数据序列的进行分割和识别，本文在CMU动作数据库的基础上，重点研究对三维运动数据序列进行动作分割。

相对于传统多媒体数据，三维人体运动数据具有其自身良好的特点，（1）数据采样率高，最新的光学运动捕获设备最高可达2 000帧/s, 产生的数据量巨大;（2）三维人体运动数据流是结构化数据且具有一定物理意义(即表示了人体各主要关节点位置信息),这为进一步对其进行结构化分析与理解奠定了基础.本文的研究主要针对三维人体运动数据提出一种有效的几何特征表示以对其直观可视化并进行交互式分割,能够提高三维人体运动数据预处理与结构化分析的效率,并为下一步的运动数据检索提供帮助.

目前为止，国内外许多学者提出了一些对三维人体运动序列的分割技术。2000年Pomplun等人[2]提出一个基于关节空间(Joint-space)的运动分割与分解算法”同样的,Fod等人[3]通过检测人体关节运动角速度过零点的方法来分割人体运动数据”2001 年,wang等人[4]使用隐马尔可夫模型对人体运动序列的数据进行分割与聚类。2003年,美国亚利桑那州立大学的Kanav Kahol等人[5, 6] 在研究对复杂人体运动系列时提出了分级行为分割(Hierarchical Activitysegmentation)算法,该算法使用一个

动态层级数据结构来表示人体运动数据,在不同的层级上使用低水平运动参数来表示运动,最后用简单贝叶斯分类器对人体运动数据进行分割, 2004 年,Lu 等人[7]在对重复性运动进行分析的时候,提出了双闭值多维分割算法来将复杂的人体运动序列自动分割为一个简单动态线性模型序列，同样在2004 年,美国卡内基梅隆大学的Jemej等人[8]针对运动捕获数据, 利用PCA对人体动作进行降维从而提取出主要成分来；基于两个不同的行为会具有不同的主元成分，简单运动比复杂运动的维度低，就可以将行为区分开来。首先对一个区间段内的运动数据进行PCA提取，然后不断的加长此区间的长度，如果在加长到某个数据帧时发现提取出的PCA成分与前面区间段的PCA成分差异较大，那么就断定运动在此处发生了行为切换。进一步对PCA进行推广,提出了PPCA分割方法,该方法假设不同运动数据具有不同的高斯分布,并采用马氏距离进行距离度量,根据比较结果对运动数据进行分割;Jemej等人假设不同的运动会形成独立的聚类,并且每个聚类都可以用一个高斯分布来表示,并据此提出了基于混合高斯模型的分割方法,该方法使用混合高斯模型来对运动序列建模,根据所属高斯分布的不同对运动数据进行分割”2005年,美国圣路易斯华盛顿大学的Riehards souvenir:等人[9] 采用流形聚类(Manifoldexustering) 的方法对运动捕获数据进行分类，同年,德国波恩大学的Meinard Muller:等人[10]使用几何特征表示运动之间的逻辑相关性,利用人体骨架内部的空间位置关系来表示动作,并将该法应用于分割。

根据自动化程度的不同，人体运动捕获数据的分割技术主要可以分为以下三类：

1、手动分割：这是最为传统的一种分割方法，当我们得到捕获数据之后，通过人为的观察将数据进行分割，这种方法主观性较强，速度慢，而且开销巨大。

2、自动分割：这种方法通过导入动作数据，有计算机程序自动检测分割点，对动作数据进行分割，这种分割方法自动化程度高，速度快，但目前还没有公认的良好的方法。

3、交互式分割：基于传统的手动分割方法笨拙落后，而自动分割方法不能很好的解决问题的情况下，从而产生了交互式的分割方法[11]。2008 年，肖俊提出了一种三维数据可视化与交互式运动分割技术。

§1.3 研究背景和意义

计算机和互联网的普及和发展使得人们越来越多的用计算机来获取信息。基于运动捕获数据驱动的人体运动研究是在计算机处理的条件下进行的人体动作分析，它能够捕捉表演者的肢体动作在三维空间中的位置，并将其转化成数字化信息。

在运动捕获数据技术的发展过程中，我们已经有越来越多的动作数据库可以供研究者使用[12]，但人体运动是一个复杂的数据序列，动作数据需要进行有效的分割

和分类存储，以便于浏览、传输和后期人体动作的合成。

但利用运动捕获技术获取信息的过程往往耗费较大。近年来，随着各种运动捕获系统的普及，运动捕获技术发展迅速，各种大规模人体运动捕获数据库也已经出现，因此，许多动画等领域的研究人员都将重点转移到如何高效利用人体运动捕获数据,如何对这些数据进行分割、检索、合成等，现有方法从不同角度对人体运动的特征进行了定义和分析，由于人体运动的复杂性和多自由度特征，仍存在特征表示不统一，有些复杂运动难以使用简单特征进行定义、表示等局限性[23]。这些因素导致人体动作的分割仍然是一项具有挑战性的研究课题，在检索率、检索精度及检索效率等方面仍有进一步提高的空间，因此在进一步的研究中，实现运动数据的快速分割，对于提高运动捕捉数据重用效率进而降低制作成本具有一定的理论价值和现实意义。

提高运动捕获数据的利用率。表演者一般情况下习惯于长时间连续表演,这样表演得更加逼真,运动之间的过渡更加自然,而且一次性捕获长时间运动序列的效率比较高，所以,动画制作人员习惯于一次捕获包含多种运动的长序列运动数据,然后对其进行后处理。为了便于运动检索、编辑等数据重用技术的进行,运动捕获数据通常需要被保存成小的独立的运动片段，这就使得运动捕获数据分割成为一个无法回避的问题。如果对运动捕获数据进行手动分割,会耗费大量的人力物力,而且效率低下，这就要求人们可以对运动捕获数据进行正确、高效的自动分割，通过上述分析可以看出,本文中对运动捕获数据分割算法进行研究具有重要意义。

§1.4 研究内容与创新

本文研究的目的就是基于现有的运动数据，对高维人体运动数据进行自动分割，以提高现有运动数据的重用性，其中涉及的技术主要有运动数据降维、动作数据分类、动作转换识别等。从而为对这些运动数据的检索、编辑、合成等行为提供优质素材。

本文首先对现有的主成分分析、高斯混合模型算法进行实验，得到动作分割结果。通过对实验结果分析，我们发现：现有的动作分割算法对简单动作片段的分割效果较好，能够得到将长动作序列分割成为不同的动作片段。但现有的分割算法没有考虑动作的分类分割，即将动作序列中的动作片段进行分类，以便于后期的存储、查询和应用。

本文在主成分分析方法的基础上，通过对不同动作片段信息的不同，提出了以转换误差和动作主成分元素不同为依据的两种动作分类分割算法，对不同的动作数据进行实验，得到了较好的分类分割结果。

§1.5 本文组织结构

本文正文部分总共分为六章，主要组织结构安排如下：

第一章：介绍课题背景和研究的意义，以及研究现状和研究内容。

第二章：介绍运动数据的的相关背景知识，并对骨骼数据格式组织进行介绍。

第三章：对主成分分析方法进行介绍，并对基于主成分分析的人体动作数据分割算法进行阐述、实验和分析。

第四章：首先高斯混合模型进行介绍，并对基于高斯混合模型算法的动作数据聚类分割算法进行实验和讨论。

第五章：提出了基于主成分分析（PCA）以投影误差为和以主成分元素组成为分类依据的动作数据分类分割算法，并对算法进行介绍，对动作数据进行实验和分析。

第六章：对本文研究内容进行总结和对该研究领域的展望。

第二章　背景知识

§2.1　人体运动捕获数据

目前,运动捕获数据格式基本都是层次结构的，主要有BVH、ASF/AMC、FBK、C3D以及HTR [13]，其中Bvh 和ASF/AMC是常用的数据格式Acclaim Skeleton File/Acclaim Motion Data, 这种数据格式是由Acclaim 公司设计开发的,由两个文件组成:一个骨架文件和一个运动文件因为在大多数时间里，是由某个特定的骨架来完成各种运动的ASF文件是骨架数据文件，AMC 文件是运动数据文件由于ASF/AMC格式的文件比较容易获得,而且这种格式的数据结构清晰,易于分析、修改,所以论文采用的研究对象是这种格式的数据，从CMU提供的人体运动捕获数据库中获得这些人体运动数据是由Vicon光学运动捕获系统的12个MX-40相机在20Hz的频率下捕获到的捕获对象身着41个标记，运动数据中包含有绝对根节点的位置和方位，以及各个骨骼的相对旋转角度。

§2.2　人体骨骼数据

如图2.1是以ASF文件格式的人体骨架数据。 ASF文件 [14]定义了人体骨架的结构

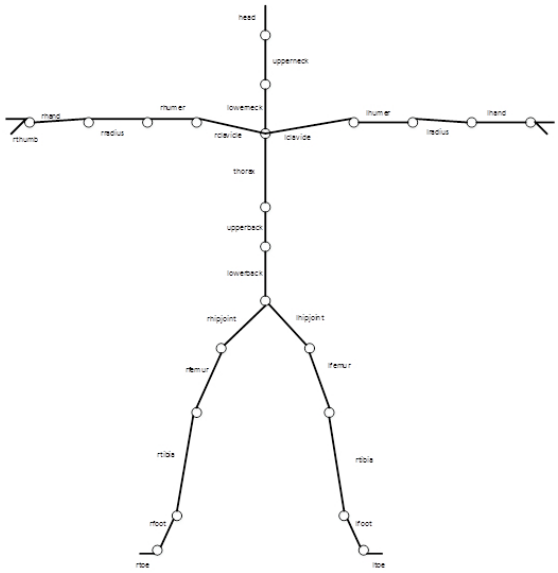

图 2.1: 人体骨架结构示意图

层次

“:version”指明骨架定义的版本

“:name”对骨架重新命名

“units”定义了不同数据类型的单位以及参数的默认值

“documentation”说明文档

“root”定义了场景中的一个特殊部分，它是骨架层次结构的根节点，除了它不包含方向和长度信息，和其他关节部位，“axis”定义了根节点对象的旋转顺序，“order”指明了运动信息，并且将以这样的顺序出现在amc文件中。“position”和“orientation”指明了根节点的起始信息和方向信息，初始都为0。

“：bonedata”对骨架结构中的每个关节以及骨架组织结构进行详细描述，每个骨架的定义以“begin”开始，以“end”结束。其中对每个骨架的描述信息如下：

“id”骨骼序号

“name”骨骼名称，每个骨骼都必须有唯一一个名称

“direction“骨骼的方向，定义了由父骨骼到子骨骼的方向，

“length“骨骼的长度

“axis“骨骼的旋转轴

“dof”自由度信息，指明了该骨骼运动通道的数量以及在amc文件中出现的顺序，其中“limit”部分给出了限制信息，对于每个通道都规定的该通道允许的最大值和最小值，它们以一对数字的形式给出。

“:hierarchy”部分描述了各部分骨骼之间的层次信息。

如果将捕获得到的人体运动数据视为由离散时间点采样得到的人体姿势序列，每个采样点为一帧，则每一帧的姿势是有各个关节点共同决定.在任意帧时刻i，人体姿势表示为

$$F_i = (p_i^1, r_i^1, r_i^2, \cdots)$$

其中p_i^1fi$r_{i\text{fi}}^1$分别表示Root关节点的位置和方向，即平移和旋转量，$r_i^j, j = 2, 3, \cdots$表示非root关节点的方向，即旋转向量，根据骨架中各个关节点之间的相互关系，在i时刻，人体骨架中任意非root 关节点N_j可以通过三维变换得到。

$$\vec{p_i^{(j)}} = T_i^{root} R_i^{root} \cdots T_0^{grandparent} R_0^{grandparent}(t) T_0^{parent} R_i^{parent} \vec{p_0^j}$$

其中，$\vec{p_i^{(j)}}$ 表示i时刻关节N_j 的世界坐标，T_i^{root}, R_i^{root} 分别表示i 时刻Root关节点的平移和旋转变换矩阵，分别由$p_i^{(1)}, r_i^{(1)}$ 生成; T_0^k 表示初始时关节N_k(N_k 为树形人体骨架中，从根结点到结点N_j 之间的任意结点)在其父关节点所在局部坐标系下偏移量生成的平移变换矩阵; R_i^k 表示i 时刻关节点N_k(N_k 取值同上)的旋转变换矩阵，由r_i^k生成，$\vec{p_0^j}$ 表示初始时，N_j。在其父结点所在局部坐标系下的偏移量。

第三章　基于主成分分析的人体运动数据分割

§3.1　算法理论

在多元统计分析中，主成分分析（Principal components analysis，PCA）是一种分析、简化数据集的技术。主成分分析经常用于减少数据集的维数，同时保持数据集中的对方差贡献最大的特征。这是通过保留主成分，做到的。这样主成分往往能够保留数据的最重要方面。

主成分分析（PCA，Principal Component Analysis）由卡尔·皮尔逊1901年提出[15]，用于分析数据及建立数理模型。PCA 提供了利用降低数据维度的有效办法，在尽可能保留原有数据信息量的前提下，将多个指标转化成少数的综合指标，通过这样得到的每个主成分之间没有相关性，是有原变量通过线性组合得到的。

PCA是最简单的以特征量分析多元统计分布的方法。通常情况下，这种运算可以被看作是揭露数据的内部结构，从而更好的解释数据的变量的方法。如果一个多元数据集能够在一个高维数据空间坐标系中被显现出来，那么PCA 就能够提供一幅比较低维度的图像，这幅图像即为在信息最多的点上原对象的一个‘投影’。这样就可以利用少量的主成分使得数据的维度降低了。

3.1.1　数学模型

设原数据有n个样本，每个样本有p个指标，则原数据的矩阵表示如下：

$$\mathbf{X}=\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix}=\begin{pmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \cdots & \mathbf{X}_p \end{pmatrix}$$

其中：

$$\mathbf{X}_i=\begin{bmatrix} x_{1i} \\ x_{2i} \\ \vdots \\ x_{ni} \end{bmatrix}, i=1,2,\cdots p$$

当p较大时，在p维空间解决问题就比较麻烦，为解决这一困难，我们需要进行降维处理，对$\mathbf{X}$的p个指标$\mathbf{X}_1,\mathbf{X}_2,\cdots,\mathbf{X}_p$线性组合，我们得到$m\ (m\leq p)$ 个新的综合指标$\mathbf{F}$，其中：

$$\alpha_i^T\alpha{=}\alpha_{i1}^2+\alpha_{i2}^2+\cdots+\alpha_{ip}^2=1$$

任意两个主成分的协方差$\mathrm{cov}(\mathbf{F}_i,\mathbf{F}_j)=0\ \ j<i,j=1,2,\cdots,m-1$, 则我们称$\mathbf{F}_i$为向量$\mathbf{X}$的第$i$个主成分。

3.1.2 算法步骤

$\mathbf{X}=\begin{pmatrix}\mathbf{X}_1 & \mathbf{X}_2 & \cdots & \mathbf{X}_p\end{pmatrix}^T$为样本矩阵，主成分分析的具体步骤如下：

（1）去均值，中心化处理。

（2）计算协方差矩阵$\mathbf{C}=(c_{ij})_{p\times p}$其中：

$$c_{ij}=\frac{1}{n-1}\sum_{k=1}^{n}(x_{ki}-\bar{x}_i)(x_{kj}-\bar{x}_j)\qquad i,j=1,2,\cdots,p$$

（3）计算协方差矩阵$\mathbf{C}$的特征值λ_i及特征向量$\mathbf{u}_i,\ i=1,2,\cdots,p$，并将$\lambda_i$从大到小进行排序。

（4）计算前m个主成分的累积贡献率：

$$\xi_m=\sum_{i=1}^{m}\lambda_i^2\Big/\sum_{i=1}^{p}\lambda_i^2$$

累积贡献率用来对主成分数据的保真程度进行度量，一般选择大于，在应用中需要根据实际情况具体确定该参数。

（5）计算前m个主成分：

$$\mathbf{F}=\mathbf{U}^T\mathbf{X},\mathbf{U}=(\mathbf{u}_1,\mathbf{u}_2,\cdots,\mathbf{u}_m)$$

3.1.3 本文应用背景

采用主成分分析法(PCA)基本出发点是简单的动作与复杂的动作用不同的维度，及有不同的主成分。动作数据是高度相关的，如我们走路时，当左髋带动左腿向前走时，右髋往往向后移动，对于其他一些简单的动作同样存在类似的相关性。这就暗示我们可以将主成分分析法用于人体运动捕获数据，去掉其内在的相关性来进行降维，重新组合成一组无关的数据，来进行进一步分析和处理，即将存在相关性的高维数据投影到低维空间[8]。

§3.2 算法描述

本算法的大致思想是简单运动与复杂运动相比，其内在维度要低，给定一个运动序列，我们首先对初始窗口帧进行PCA处理，得到其内在维度，然后逐渐以1帧的

距离向后追加，确定其内在维度，当运行到某一帧，该动作的内在维度发生变化时，进行分割，这是依据给定投影误差e，复杂动作比简单动作需要更高的维度；同样，固定投影的维度，我们继续逐渐向后追加动作帧，投影误差会在某一帧急剧增加，该帧就是分割点，即给定维度r，复杂动作的投影误差要比简单动作大。

3.2.1 SVD分解

算法最主要的部分就是进行主成分降维，本文使用了奇异值分解（SVD）的方法。

首先将运动序列数据的中心减去，进行中心化处理，得到一个$n \times q$ 的矩阵$\mathbf{D}$，其中n表示帧数，$n \gg q$，将$\mathbf{D}$进行SVD 分解，得到：

$$\mathbf{D} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$$

其中，$\mathbf{U}$是一个$n \times n$正交列矩阵，$\mathbf{V}^T$是一个$q \times q$的正交行向量，$\mathbf{\Sigma}$是一个$n \times n$的对角阵，即$\mathbf{\Sigma} = \begin{bmatrix} \mathbf{\Sigma}_1 & \mathrm{O} \\ \mathrm{O} & \mathrm{O} \end{bmatrix}$ $\mathbf{\Sigma}_1 = diag(\sigma_1, \sigma_2, \cdots \quad \sigma_R)$ ，其对角元素按顺序

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_R > 0 \quad R = rank(\mathbf{D})$$

$\mathbf{V}$的前r列给出了r维最优子空间的主要成分，即$v_1, v_2, \cdots, v_r$，此时相当于丢弃了r维以外的所有奇异值，此时投影误差为：

$$e = \sum_{i=1}^{n} \left\| (x_i - x'_i)^2 \right\| = \sum_{i=r+1}^{R} \sigma_i^2$$

累积贡献率为：

$$\xi_r = \frac{\Sigma_{j=1}^{r} \sigma_j^2}{\Sigma_{j=1}^{R} \sigma_j^2}$$

累计贡献率ξ_r表示了将数据投影到r 维子空间后还保留的原始信息量，通过设置阈值$\tau(\tau < 1)$，选取最小的维度r使得$\xi_r > \tau$，本文选取$\tau = 0.9$。

3.2.2 算法思想

首先，初始化一个$k(k = 240)$帧的数据窗口，对窗口中的数据进行降维处理，利用$\xi_r > \tau$，得到维度r。逐帧增加运动数据，利用SVD 分解前$i(k < i \leq n)$帧，使其投影到r维子空间，根据公式计算投影误差e_i，如图3.1

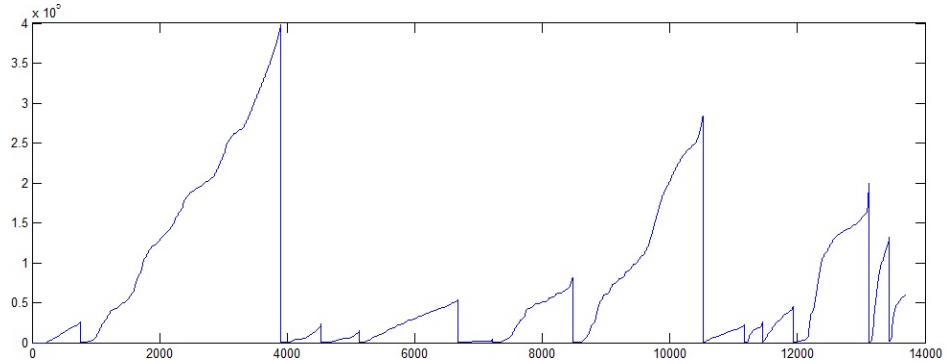

图 3.1: 动作片段投影误差e示意图

我们可以发现：投影误差e_i随着帧数的增加以相对平缓的速度增加，这主要是因为对于一个简单的运动，其内在维度是相对固定的，每一帧的动作数据对投影误差e_i的贡献是相近的。

当进入新的动作时，e_i会急剧增加，因此，动作的分割点实际上就是投影误差出现剧烈变化的帧。

为了检测动作的转换，本文利用误差e_i的导数$d_i = e_i - e_{i-l}(l = 60)$ 如图A.5所示。

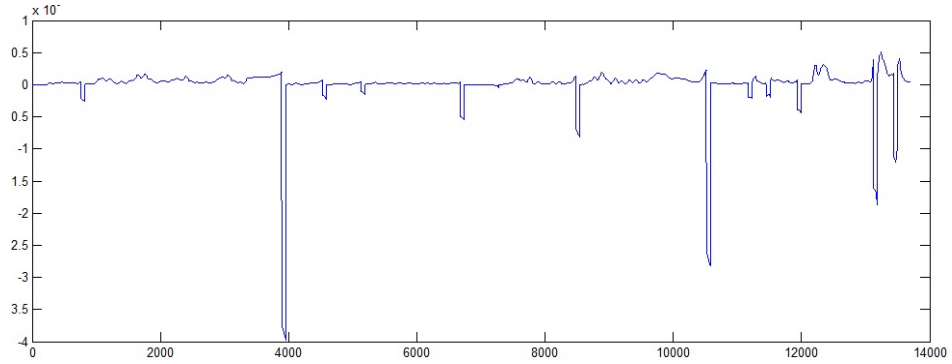

图 3.2: 动作片段示意图

对于简单的动作，d_i 会在某个均值范围内，有小幅度的抖动，基本保持不变；而当发生动作的变化时，d_i会发生急剧变化。

对动作数据进行分割，首先对于前i帧数据，我们计算其均值avgd和标准差sd，对于任一帧j，如果$\mathrm{d}_j > avgd + k_\sigma sd(k_\sigma = 3)$，则$j$为分割帧，逐帧增加$j$，计算得到全部分割点。

```
for every frame j
    e(K)=e(K)+SS(i);
    d(K)=e(K)-e(K-l);
```

```
    sd;
    if (d(j)-avgd)>3*sd
        add jth frame to cuts
    end
end
```

3.2.3 实验结果

对于测试动作M，有可以看到如图3.3，该动作有7个动作片段，分别是走路、跳跃、走路、击拳、走路、踢腿、击拳。分割结果

如图2.1是以ASF文件格式的人体骨架数据。 分割结果见表3.1

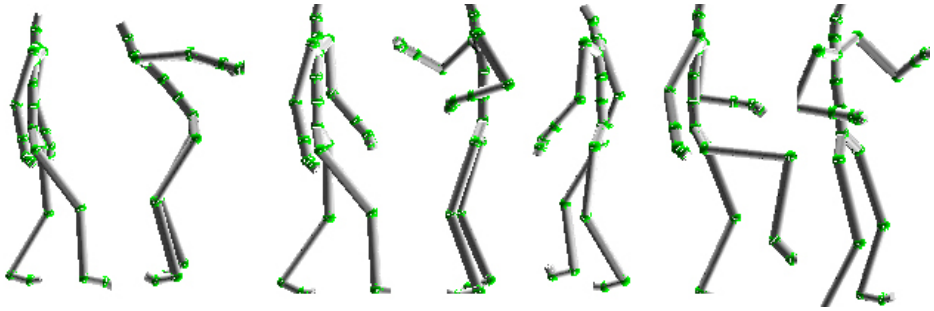

图 3.3: 动作片段示意图

表 3.1: 混合动作组成表

运动片段	起始帧	结束帧	PCA分割结果	动作描述
1	1	532	500	走路
2	533	1173	1140	跳跃
3	1174	1942	1959	走路
4	1943	2520	2509	击拳
5	2521	3177	3288	走路
6	3178	4080	4007	踢腿
7	4081	4579	4579	击拳

通过实验结果，我们可以看到，PCA算法对于又6个动作片段组成的长序列动作的分割能够得到比较正确的分割结果。

基于主成分分析的分割算法的分割结果的精确程度主要取决参数的正确选取，如本次实验中的k_σ、τ等。另外，如果对于某一个简单运动的分割出现了误差，那么

重启算法的位置将会是有问题的。根据该误差的大小，由本算法得到的下一个子空间会受到或大或小的影响，在分割的过程中不断延续和扩大，以致影响最终的分割效果，这也是本算法存在的一个局限性。

第四章　基于高斯混合模型的人体运动数据分割

运动数据分割，实际上就是对数据进行聚类，本章将首先对高斯混合模型GMM进行介绍，并将其用于人体运动数据进行聚类分割。

聚类分析是一种在缺少先验知识的条件下，将一个数据集划分成若干个组或簇，不同组之间中的对象相异，而同一组中的对象相似，从而达到分类的目的。聚类应用及其广泛，其研究历史也相当悠久。目前，聚类分析在海量数据出现后已成为数据挖掘领域中十分重要的研究课题之一。

§4.1　数学模型

高斯模型是有限混合模型的一种，并且已经被广泛应用于统计学习、模式识别等许多领域，在模型中，聚类数据被看成是多个来自多个正态分布的混合概率分布，其中每一个正态分布当做一个类。

高斯混合模型结构就是将一定数量的高斯函数（正态分布函数）线性组合，以此混合高斯函数来逼近某变量的概率分布的方法。从理论上说，我们可以用不同的分布函数来构造任意格式的混合模型，但是发展到现在，由于各个分布函数的特殊性，只有GMM 是最为流行的。通过增加高斯模型中的模型个数，该模型也可以变得非常复杂，能够任意地逼近任何连续的概率密度分布。

高斯混合模型是M个高斯密度函数的线性组合，定义形式如下：

$$p(\mathbf{x}|\lambda) = \sum_{i=1}^{M} \pi_i \mathrm{N}_i(\mathbf{x}|\lambda_i)$$

其中，$\mathbf{x}$是一个D维观测向量，$\mathrm{N}_i(\mathbf{x}|\lambda_i)$是高斯子分布，$\lambda$是GMM模型的参数集，$\pi_i$表示每个高斯子分布的权重，其中$0 \leq \pi_i \leq 1, \sum_{i=1}^{M} \pi_i = 1$，每个高斯分布都是一个$D$维联合概率分布，可以表示为：

$$\mathrm{N}_i(\mathbf{x}|\lambda_i) = |\Sigma_i| \ \exp\left\{ -\tfrac{1}{2} \left((\mathbf{x} - \mu_i)^T \Sigma_i^{-1} (\mathbf{x} - \mu_i) \right) \right\}$$

其中，μ_i表示第i个高斯子分布的$D \times 1$维均值向量，Σ_i表示第i个高斯子分布的$D \times D$维协方差矩阵。一个完整的GMM模型可由均值向量μ、协方差矩阵Σ 以及权重系数π决定。

$$\lambda = \{ \ \pi_i \ , \mu_i \, , \Sigma_i \ \} \ , \ i = 1, 2, \cdots, M$$

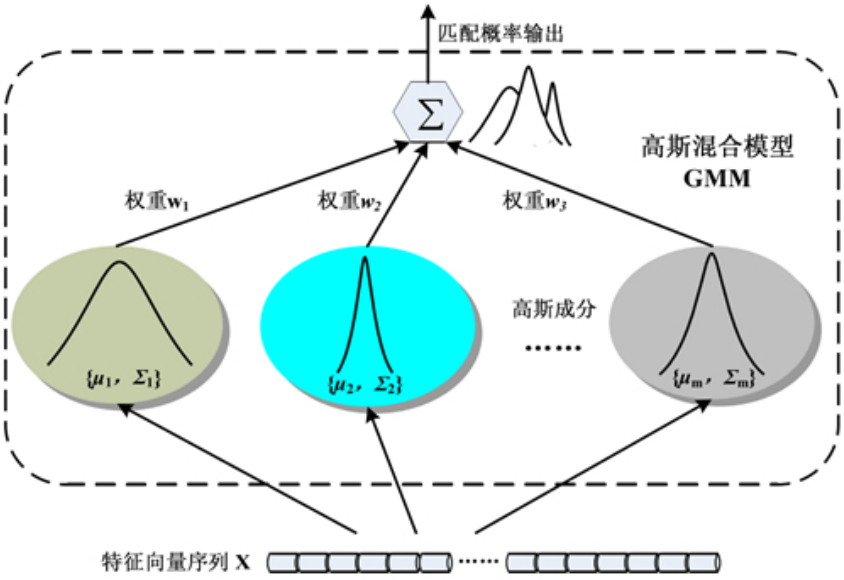

图 4.1: GMM模型结构示意图

为对GMM模型进行计算，协方差矩阵Σ_i一般采用对角阵：

$$\sum_i = diag(\delta_{i0}^2, \delta_{i1}^2, \cdots, \delta_{iD-1}^2)$$

其中，$\delta_{ij}^2\ j = 0, 1, \cdots, D-1$表示GMM模型中第$i$个高斯子分布所对应特征向量的第$j$维分量的方差，我们得到:

$$\begin{aligned} \mathrm{N}_i(\mathbf{x}|\lambda_i) &= |\ \sum_i|\ \exp\left\{-\tfrac{1}{2}\left((\mathbf{x}-\mu_i)^T \sum_i^{-1}(\mathbf{x}-\mu_i)\right)\right\} \\ &= \prod_{j=0}^{D-1} \tfrac{1}{\sqrt{2\pi}\delta_{ij}} \exp\left\{-\tfrac{(\mathbf{x}_{ij}-\mu_{ij})^2}{2\delta_{ij}^2}\right\} \end{aligned}$$

我们需要对进行参数估计，其中最常用的方法是最大似然估计（Maximun Likleyhood），本文使用EM算法进行极大似然估计，当观测数据为不完全数据时可以利用EM 算法求解残缺数据集的最大似然估计。EM 算法的性能与最大似然估计相近，但它可以大大降低最大似然估计的计算复杂度。EM 算法可以广泛的应用于成群数据、截尾数据、缺损数据等数据中，具有很好的实际应用价值。算法的每一次迭代都是由一个E 步(求期望)和一个M 步(将期望最大化)构成的。EM 算法通常用于解决残缺数据的情况下对未知的参数进行最大后验似然估计以及极大似然估计，算法的具体步骤见 [24]。

人体运动序列包含多个动作，而每个动作可以近似看成是一个高斯分布，这样，我们得到的运动数据就成为了一个GMM模型，为粗略验证这一猜想，本文首先绘制了简单的人体运动数据，将运动数据经PCA投影到2维空间，即只保留两个主成分，我们可以得到下图4.2：

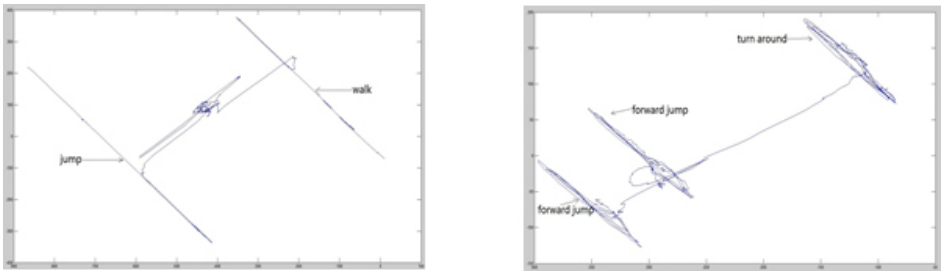

图 4.2: 投影在2维平面上的简单动作：（a）forward jump and turn around; (b) jump and walk.每一个动作都有自己的聚类。

我们可以看到，不同的动作有自己的聚类，而每一类动作可以用高斯分布进行建模，这就暗示我们可以用GMM 来进行动作分割。

§4.2 算法描述

捕捉一段动作数据$\mathbf{M}_{n\times D}$ $(n = frames, D = 56)$，假设每个单独的动作可以有单高斯分布表示，因此$\mathbf{M}$ 便是一个k阶混合高斯分布，首先我们用EM算法计算每一个高斯子分布的均值μ_i、协方差矩阵Σ_i以及权重系数π_i，每一个聚类具有不同的长度，长的动作片段有更多的点集，从而形成较大的类，其中π_i 表示某一点属于类别C_i的概率。

对动作序列的每一帧，我们计算该帧数据最有可能属于的类C_i，如果两帧数据属于不同的类，则人为在此处发生了动作变化，该帧是分割点。

```
function gmmsegmentation
for i=1:frames
    get the cluster k=cluster(i) of ith frame;
        if k!=cluster(i-1)
           Record i;
    end
end
```

§4.3 实验结果

对数据库中动作文件进行高斯聚类，我们得到图4.3

对同上一章中的同一动作进行高斯混合分类分割，我们得到结果见表4.1

通过上表我们可以看到，高斯模型的动作分割算法能够实现动作片段的分割。

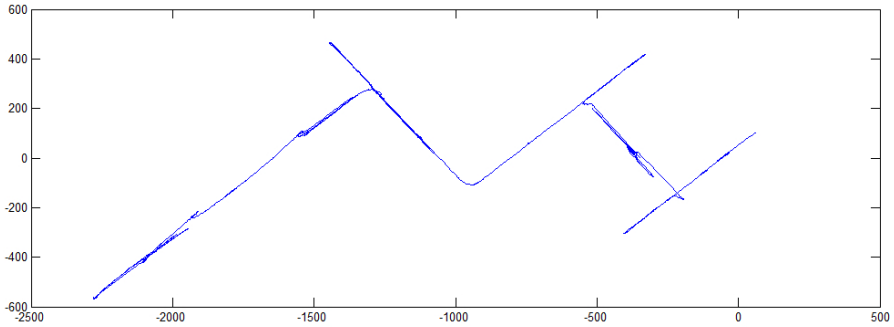

图 4.3: 动作的高斯聚类示意图

表 4.1: 混合动作组成表

运动片段	起始帧	结束帧	GMM分割结果	动作描述
1	1	532	523	走路
2	533	1173	1131	跳跃
3	1174	1942	1901	走路
4	1943	2520	2549	击拳
5	2521	3177	3221	走路
6	3178	4080	4042	踢腿
7	4081	4579	4579	击拳

在实际实验中，使用高斯混合模型之前，我们需要事先知道动作种类的个数，然后才能进行聚类分割；对于同一动作连续出现的情况，高斯混合模型处理效果也不理想，有一定的局限性。

第五章　基于主成分分析的运动数据分类分割

以上运动数据的分割，我们可以发现对于实际运动数据，常常会有某一动作连续演示多遍然后作为输入数据，如果按普通的PCA-SVD方法，我们只能将数据分割成多个动作，而实际的动作数据不能有效的分类。

针对以上问题，本文在原有的基础上提出了两种想法：

1、根据投影误差导数进行分类，一般而言，同一动作的转换误差是相似的，不同动作的转换误差差距是较大的

2、根据主成分的组成进行分类，同样，同一动作的主成分是相似的，不同动作的主成分是不同的。

§5.1　基于误差变化的运动数据分类分割

5.1.1　算法描述

通过观察某一连续动作的d_i分布，如图5.1：

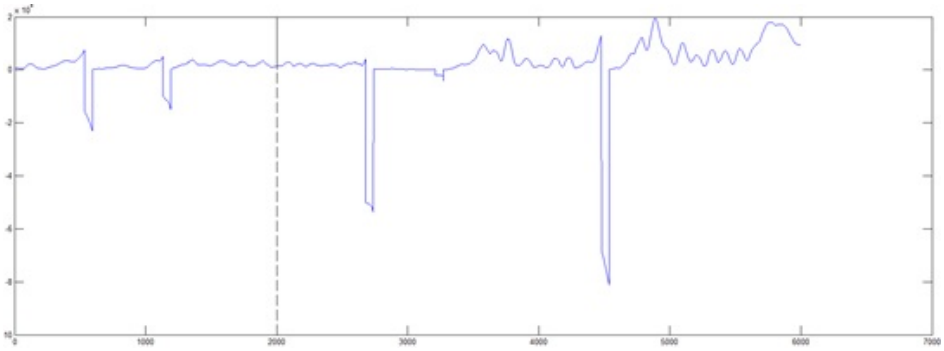

图 5.1: 动作序列d_i变化图(其中有两种不同的动作，有虚线分隔开)

我们可以发现：2000-4000帧两个的动作投影误差的导数明显比1-2000帧的大，这属于不同的动作，而且前两个动作与后两个动作的投影误差相近，是属于同一个动作。

对于某一连续动作的分类分割，我们首先按照PCA方法逐帧计算d_i，当计算得到分割帧i时，记录d_i，继续增加i，当再次出现第j帧为分割帧时，如果$d_j < 2d_i$则认为两个动作为同一动作，否则两个动作为不同的动作，对结果进行记录。

```
for every frame j
    e(K)=e(K)+SS(j);
    d(K)=e(K)-e(K-1);
    if find a cut
        if d(j)<2*previous d
                the same behaviour
        else
                a new behaviour
        end
    end
end
```

5.1.2 实验结果

本文对采集的动作进行分类分割，原始动作数据见表5.1

表 5.1: 基于投影误差的舞蹈动作分割结果

运动片段	起始帧	结束帧	动作描述
1	1	1030	向左走
2	1031	1717	向右走
3	1718	2414	向左跳
4	2415	3037	向右跳
5	3038	3464	向左跑
6	3465	5127	向右跑
7	5128	6812	向左走
8	6813	7217	向右走

基于误差变化的动作分类分割如下表5.2

通过分割结果可以发现，基于误差的分类分割方法能够在普通分割的基础之上把重复的动作片段归类。如原始动作帧到帧之间是重复的动作片段，而现在我们首先可以将它们归为一类，然后能够将每一类动作再进行划分。

本方法只是提出了动作需要逐步分类的思想，提出了基于误差大小的分割方法，能够简单实现动作片段投影误差较大的数据之间的分类。

表 5.2: 基于投影误差的舞蹈动作分割结果

运动片段	起始帧	结束帧	分类类别	动作描述
1	1	1030	1	向左走
2	1048	1717	1	向右走
3	1708	2414	2	向左跳
4	2455	3037	2	向右跳
5	3026	3464	3	向左跑
6	3477	5127	3	向右跑
7	5144	6812	4	向左走
8	6802	7217	4	向右走

方法还存在一些问题，例如，不同动作之间的判断依据只是动作发生变化时进行判断，不能有效的利用动作本身包含的信息作为动作分割的依据，无法处理多个相同的动作片段不连续的情况，因此我们需要对动作的自身信息进行利用，作为分割动作的依据。

§5.2 基于主成分元素组成的运动数据分类分割

5.2.1 算法描述

基于误差大小的动作数据分类分割方法，能够简单实现动作片段投影误差较大的数据之间的分类，但投影误差不能真实的反应不同动作之间的行为和肢体差异，无法处理不连续动作片段的分类。因此，本文又提出了一种基于动作数据主成分组成元素的分类分割算法。

通过第三章主成分分析法：

$$\mathbf{D} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$$

其中，$\alpha_i = \{\alpha_{i1}\ , \alpha_{i2} \cdots \alpha_{ip}\}$是动作$\mathbf{D}$第$i$个主成分中每个子成分所占的比例系数。

同一个简单动作的主成分是相似的，因此其主成分中各个子成分的比例系数是相似的，而不同的动作其主成分差异较大，反应到各子成分中便是α_i差异较大。

对于动作$\mathbf{D}$，我们按照先前的方法，首先固定主成分个数r，当逐帧增加到i，出现分割点时，记录前一动作片段到$i-1$帧的主成分信息α_{i-1}，当再次遇到动作分割点时，计算α_i，计算两者之间的绝对值距离

$$d(\alpha_{i-1}, \alpha_i) = \sum_{j=1}^{p} |\alpha_{i,j} - \alpha_{i-1,j}|$$

如果$d(\alpha_{i-1}, \alpha_i) > \theta$，则出现新的动作，否则第$i$帧属于重复的动作变化点，本文中，不同运动数据片段之间的距离d见图5.2

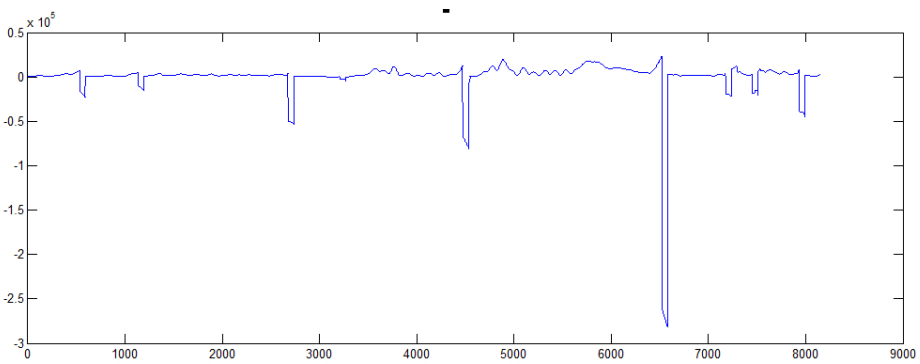

图 5.2: 动作序列片段之间距离d示意图)

```
for every frame
    if find a cut i
        for every previous cut j
            if d(i,j) > θ
                a new behaviour
            else
                the same behaviour j
            end
        end
    end
end
```

实验表明，当$\theta = 0.7$时，能够较好地区别不同的动作。

5.2.2 实验结果

对运动数据进行基于主成分组成元素的分类分割，输入动作为包含四类动作的文件，其中部分动作片段是重复而且不连续的，利用基于主成分元素组成的分类分割算法，我们得到结果见表5.3

我们可以看到，对于不同动作片段，该算法可以较好的实现分类。对于动作序列中位于不同位置的运动片段，也能较好的实现类别划分。片段1和片段9属于同一个动作，该方法可以根据其动作主成分的相似性划分为同一类动作，这是因为动作

表 5.3: 基于主成分元素组成的舞蹈动作分割结果

原始序列	原始运动片段类别	起始帧	结束帧	基于元素组成的分类分割结果	分类情况
1	1	1	1030	1012	1
2	1	1031	1717	1709	1
3	1	1718	2414	2410	1
4	2	1943	3037	3030	2
5	2	2521	3464	3490	2
6	2	3178	5127	5143	2
7	3	5128	6812	6820	3
8	3	6813	7217	7260	3
9	1	7218	8509	8540	1
10	1	8510	9780	9803	1
11	4	9871	10357	10397	4
12	4	10358	11516	11546	4
13	4	11517	12529	12762	4
14	2	12530	13482	13426	2
15	2	13483	13700	13700	2

片段1和9属于同一类动作，其主成分之间的距离小，因此尽管两个动作片段是不连续的，但是根据算法依然能够从动作序列判断其属于同一动作。

第六章　总结与展望

§6.1　总结

近年来，随着计算机技术的飞速发展，运动捕获技术应用范围越来越广。目前传统的运动捕获所需的费用较高，如何利用已有的运动捕获数据库，缩短数据获取时间，减少经费开销，已经成为了计算机动画等领域关注的热点。而出于保证运动的真实性，尽量减少时间和经费开销等方面的考虑，目前公开的运动捕获数据库中基本都是长序列的运动。手动对数据进行分割耗时耗力，这就使得研究人员不得不考虑动作数据的自动分割。

本文对运动捕获数据的分割算法进行了研究，首先对基于主成分分析和高斯混合模型的分割方法进行了实现，对于实际动作数据中存在重复动作的情况，本文提出了一种分类分割的方法，分别以投影误差和个动作主成分之间的距离作为分类依据进行动作的分类分割。

§6.2　展望

近几年来，国内外对人体运动捕获数据方面的研究逐渐升温，主要包括在运动捕获数据的分割、编辑、检索、风格化等方面。其中，分割方面目前还没有成熟有效地适用于各种运动捕获数据库，这对人体运动捕获数据分割方向未来的研究工作来说，既是挑战，也是机遇。

本文在人体运动捕获数据的分割方面做了一定的工作，但对这个方向来说，只是一个初步的研究，还有着很大的发展空间。对于未来的工作，认为主要可以从特征提取和分割方法两个方面去展开[22]：一：对于特征提取而言，原始信息应该更加考虑人体的运动信息，从中可以分析出很多的人体特征。在本文中，使用骨骼的主成分特征，而位置，速度等信息并没有使用。后期可以从骨架文件和运动文件中提取出人体骨架中root 节点和各个关节的位置、方位，以及速度等特征信息，将各种特征进行适当的融合，以期取得更好的分割效果；二：对于具体分割算法来说，人体运动捕获数据的处理涉及到图形学、模式识别、数据分析等多个领域，各个领域都有很多经典算法，应用到对运动捕获数据的分割上来，使得分割算法朝着普适性越来越强、自动化程度越来越高、分割效果越来越精确、耗时越来越短这几个方向发展。

参考文献

[1] 肖俊. 智能人体动画若干关键技术研究[D]. 杭州:浙江大学. 2007.6.

[2] M.PomPlun and M.J.Matarie. Evaluation metries and results of human arm movementimitation. In Proeeedings of the First IEEE-RAS intemational Conference on Humanoid Robots，Mrr，Cambridge，MA，2000.

[3] A.Fod，M.J.Matarie，and O.C.Jenkins. Automated derivation of Primitives for movement classification. Autonomous Robots，12(1): 39-54，2002

[4] wang.T.，Shum.H，Xu.Y，Zheng.N. Unsupervised Analysis of Human Gestures. IEEE Pacific Rim Conference on Multimedia: 174-181，2001

[5] Kanav Kahol，Priyamvada Tripathi，Sethuraman Panchanathan. Gesture Segmentation in Complex Motion Sequences. Proeeedings IEEE International Conference on Image Processing，2003:10508

[6] Kanav Kahol，Priyamvada Tripathi，Sethuraman Panehanathan. Automated Gesture Segmentation From Dance Sequenees. In Proeeedings of the IEEE International Conference on Automatic Face and Gesture Recognition，Seoul，Korea，2004:883-888 2001

[7] ChullMei Lu，Nicola J.Ferrier. Repetitive Motion Analysis: Segmentation andEvent Classifieation. IEEE Transaetions on Pattern Analysis and Machine Intelligence，2(26)，2004:258-263

[8] Jernej Barbic，Alla Safonova，Jia-Yu Pan，Christos Faloutsos，Jessica K. Hodgins. Nancy S.Pollard. Segmenting Motion Capture Data into Distinct Behaviors. Proceedings of Graphics Interface (GI 2004)，2004.

[9] Riehard Souvenir，Robert Pless. Manifold Clustering. IEEE International Conference on Computer Vision(ICCV05). 2005:648-653

[10] Meinard Muller，TidoR6der，Miehael Clausen. Efficient Content Based Retrieval of Motion Capture Data. ACM Trans. Graph，3(24)，2005:677- 685 2005:648-653

[11] 肖俊，庄越挺，吴飞. 三维人体运动特征可视化与交互式运动分割. 软件学报，2008，19(8): 1995-2003 9531

[12] http://mocap.cs.cmu.edu/subjects.php

[13] Brotman ，L.S.，NetravaliA.N. Motion interpolation by optimal control. Computer Graphics，Vol.22，No.4，1988:179-188

[14] http://research.cs.wisc.edu/graphics/Courses/cs-838-1999/Jeff/ASF-AMC.html

[15] Pearson，K. On Lines and Planes of Closest Fit to Systems of Points in Space (PDF). Philosophical Magazine. 1901，2 (6): 559–572.

[16] http://bubblexc.com/y2011/8

[17] 冯逢. 人体运动捕获数据的分割算法研究

[18] 黄锋. 一种改进的基于GMM模型的语音序列评分和分类方法

[19] 郭秀珍，陆建峰，汤九斌. 周期性时间序列数据聚类算法的改进研究. MICROELECTRONICS COMPUTER. 2012.

[20] 杨跃东，王莉莉，郝爱民，封春升. 基于几何特征的人体运动捕获数据分割方法. Journal of System Simulation. 2007

[21] I.T. Jolliffe. Principal Component Analysis. Springer Verlag. New York，1986.

[22] 冯锋. 人体运动捕获数据的分割算法研究. 2011.

[23] 肖伯祥，张强，魏小鹏. 人体运动捕捉数据特征提取与检索研究综述. 计算机应用研究. 2010.

[24] M.Jordan. Pattern Recognition and Machine Learning. 2006，430-439.

第七章　致谢

回顾四年大学的点点滴滴，我要衷心的感谢许多老师、同学和朋友，他们对我的学习、生活提供了数不清的帮助和鼓励，让我拥有了一个充实快乐的大学生活。

首先要感谢我的导师——陈宝权教授，陈老师严谨的治学态度、平易近人的个人魅力和卓越的学术眼光深深的影响力我，在与陈老师认识的这短短的几个月里，他教会了许多人生的道理，我非常庆幸和期待未来的研究生生涯能跟随陈老师。同时要感谢各位实验室的老师！感谢钟凡老师，钟老师技术大牛！每每都能让我感觉到自己是这么幼稚。未来我也要努力的提高自己!

感谢默默为我调bug的峰哥，绞尽脑汁为我想论文题目的新波学长，陪我打打闹闹的海森、庆楠、魏源，还有琼姐、斌哥、化永.....非常期待与你们共度我的研究生生涯。

感谢我一班的好同学、好兄弟，陪我一起度过四年的挺明、志鹏、大川、佳彬、小高、大高、郭哥、侯哥.....我们一起哭过、笑过、打闹过，经历有太多太多...我会把它当成人生中最美好的回忆。感谢一起陪我度过大学四年的各位老师同学!

最后感谢我的家人和帮助过我的人，谢谢你们一直对我默默的爱和支持，让我有了一直坚持下去的勇气，今后的我也一定会继续更加努力，闯出一片天地。

附录 A　原文

Motion segmentation and pose recognition with motion history gradients

Abstract

This paper presents a fast and simple method usinga timed motion history image (tMHI) for representingmotion from the gradients in successively layered silhouettes.This representation can be used to (a) determine the current pose of the object and (b) segment and measure the motions induced by the object in a video scene. These segmented regions are not "motion blobs", but instead are motion regions that are naturally connected to parts of the moving object. This method may be used as a very general gesture recognition "toolbox". We demonstrate the approach with recognition of waving and overhead clapping motions to control a music synthesis program.

key words: Mocap data　classification segmentation　PCA　GMM

§ A.1　Introduction and related work

Three years ago, a PC cost about U.S. $2500 and a lowend video camera and capture board cost about U.S. $300. Today, the computer could be had for under U.S. $700 and an adequate USB camera for under U.S. $701. It is not surprising then that there is an increasing interest in real-time vision on low-end computers. A heightened interest in understanding and recognizing human movements has appeared in tracking and surveillance systems, human–computer interfaces and entertainment domains. For example, monitoring applications may wish to signal only when a person is seen moving in a particular area (perhaps within a dangerous or secure area), interface systems may require the understanding of gesture as a means of input or control, and entertainment applications may want to analyze the actions of the person to better aid in the immersion or reactivity of the experience.

Recently, there have been several popular approaches to the recognition of human motion [8–11, 13–16, 20, 22, 33], with much emphasis on real-time computation. Several survey papers review vision-based motion recognition [36], human motion capture [31,

32] and human-motion analysis [1]. The most common frameworks to recognition of body motion and gesture include the analysis of temporal trajectories of the motion parameters [6, 29, 35, 40], hidden Markov models (HMMs) and state-space models [37, 39, 41], and static-activity templates [12, 17, 19, 34].

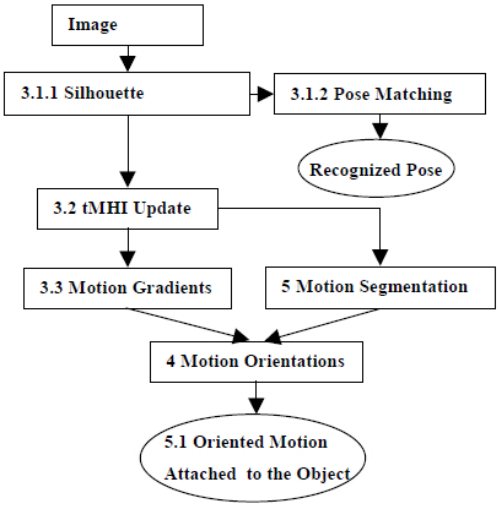

图 A.1: Process flow chart with section numbers

One other possible motion representation to describe an action sequence could be the collection of optical flow over the image or region of interest throughout the sequence, but this is computationally expensive and many times not robust. Hierarchical [2] and/or robust estimation [5] is often needed, and optical flow frequently signals unwanted motion in regions such as those containing loose and textured clothing. Moreover, in the absence of some type of grouping, optical flow happens frame to frame whereas human gestures may span several seconds. Despite these difficulties, optical flow signals have been grouped into regional blobs and used for gesture recognition [13]. An alternative approach was proposed in [17] where successive layering of image silhouettes of a person into a single template was used to represent and recognize patterns of human motion. Every time a new video frame arrives, the existing silhouettes are decreased in value subject to some threshold and the new silhouette (if any) is overlaid at maximal brightness. This layered motion image is termed a motion history image (MHI). MHI representations have the advantage that a range of times from frame to frame to several seconds may be encoded in a single image. In this way, MHIs span the time scales of human gestures. In [17], moment features of the entire MHI image were used to recognize particular activities.

The outline of this paper is as follows. Section 2 reviews previous MHI research.

Sections 3–5 are summarized in the processing flow chart in Fig. 1 where numbers indicate which section that processing step is described. In this paper, we factor pose from motion and segment the motion regions. We take Hu moment shape descriptors [25] of the current silhouette to recognize pose. We generalize the MHI to directly encode actual time in a floating-point format that we call the timed motion history image (tMHI). A gradient of the tMHI is used to determine normal optical flow (e.g. motion flow orthogonal to object boundaries). The motion is then segmented relative to object boundaries and the motion orientation of each region is obtained. The end result is recognized pose, and motion to that pose –a general "tool" for use in object motion analysis or gesture recognition. Section 6 compares the computational advantages of our approach with other optical flow approaches such as used in [13]. In Sect. 7 we use our approach to recognize walking, waving and clapping motions to control musical synthesis. Section 8 concludes the paper.

§ A.2 Pose and motion representation

The algorithm as shown in Fig. 1 depends on generating silhouettes of the object of interest. Almost any silhouette generation method can be used. Possible methods of silhouette generation include stereo disparity or stereo depth subtraction [3], infra-red back-lighting [16], frame differencing [17], color histogram back-projection [9], texture blob segmentation, range imagery foreground segmentation, etc. We chose a simple background subtraction method for the purposes of this paper.

Although there is recent work on more sophisticated methods of background subtraction [18, 24, 30], we use a fast,simplistic method here. We label as foreground those pixels that are a set number of standard deviations from the mean RGB background. Then a pixel dilation and region growingmethod is applied to remove noise and extract the silhouette. A limitation of using silhouettes is that no motion inside the body region can be seen. For example, a silhouette generated from a camera facing a person would not show the hands moving in front of the body. One possibility to help overcome this problem is to simultaneously use multiple camera views. Another approach would be to separately segment the flesh-colored regions and overlay them when they cross the foreground silhouette.

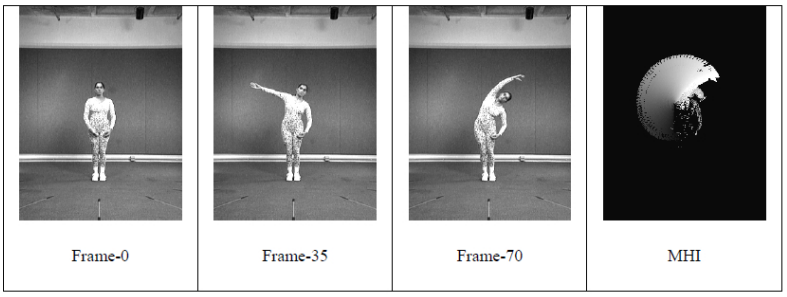

图 A.2: Motion history image for arm-stretching movement generated from layered image differences

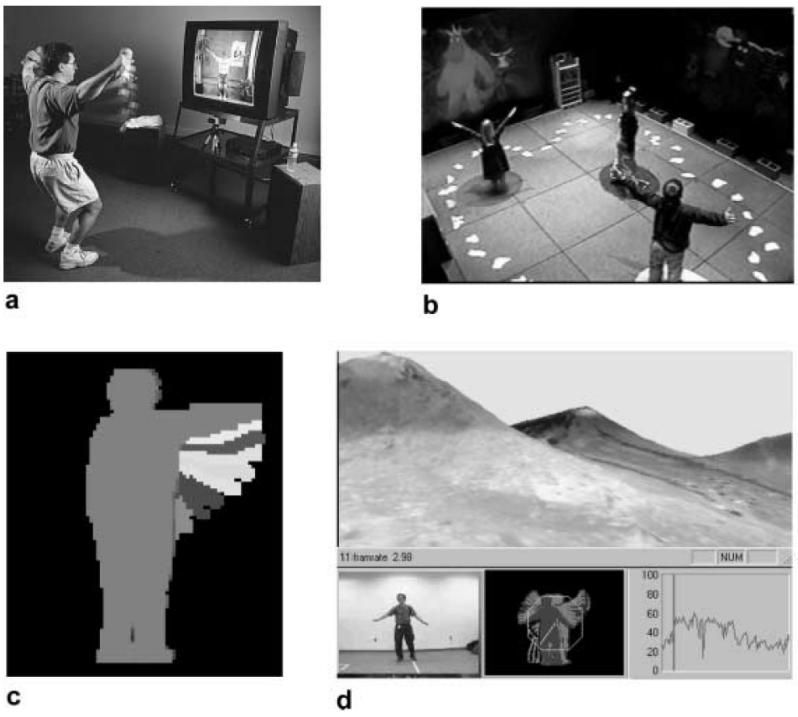

图 A.3: Test postures Y, T and ⊢

A.2.1 Mahalanobis match to Hu moments of silhouette pose

For recognition of silhouette pose, seven higher-order Hu moments [25] provide shape descriptors that are invariant to translation and scale. Since these moments are of different orders, we use the Mahalanobis distance metric [38] for matching based on a statistical measure of closeness to training examples:where x is the moment feature vector, m is the mean of the training moment vectors, and K1 is the inverse covariance matrix for the training vectors. The discriminatory power of these moment features for the silhouette poses is indicated by a short example. For this example, the training set consisted of five people performing five repetitions of three gesture poses (y,t,and b) shown in Fig. 4. A sixth person who had not practiced the gestures was brought in to perform the gestures for testing. Table 1 shows typical results for pose discrimination. We can see that even the confusable poses Y and T are separated by more than an order of magnitude making it easy to set thresholds to recognize test poses against trained model poses. An alternative approach to pose recognition uses gradient histograms of the segmented silhouette region [8]. In this paper, we use a floating-point MHI [14]

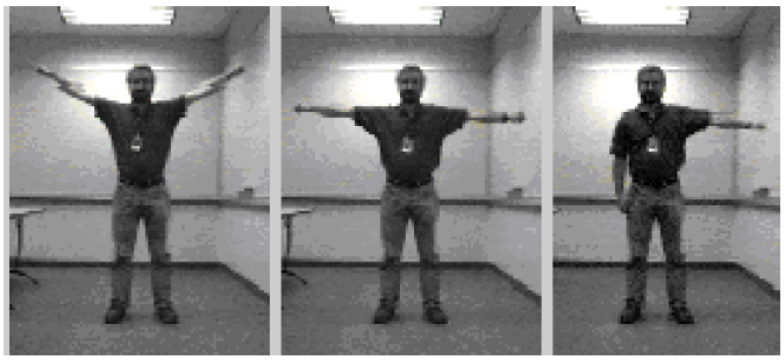

图 A.4: Successive silhouettes of an upward arm movement encoded in floating-point timestamps yields the tMHI. Brighter tMHI values depict more recent motion

where new silhouette values are copied in with a floating-point timestamp in the format seconds.milliseconds. This MHI representation is updated as follows:

where *tau* is the current timestamp, and δ is the maximum time duration constant (typically a few seconds) associated with the template. This method makes our representation independent of system speed or frame rate (within limits) so that a given gesture will cover the same MHI area at different capture rates. We call this representation the tMHI. Figure 5 shows a schematic representation of a tMHI for a person doing an upward arm movement. Notice in the right image in Fig. 5 that if we took

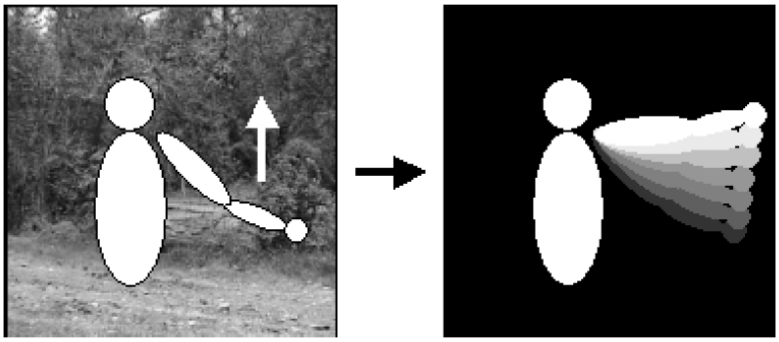

图 A.5: tMHI; gradients; mask; global orientation

the gradient of the tMHI, we would get direction vectors pointing in the direction of the movement of the arm. Note that these gradient vectors will point orthogonal to the moving object boundaries at each "step" in the tMHI giving us a normal optical flow representation (see center left image Fig. 6). Gradients of the tMHI can be calculated efficiently by convolution with separable Sobel filters in the x and y directions yielding the spatial derivatives Fx(x, y) and Fy(x, y). Gradient orientation at each pixel is then We must be careful, though, when calculating the gradient information because it is only valid at locations within the tMHI. The surrounding boundary of the tMHI should not be used because non-silhouette (zero value) pixels would be included in the gradient calculation, thus corrupting the result. Only tMHI interior silhouette pixels should be examined. Additionally, we must not use gradients of tMHI pixels that have a contrast which is too low (inside a silhouette) or too high (large temporal disparity) in their local neighborhood. Figure 6 center left shows raw tMHI gradients. Applying the above criteria to the raw gradients yields a masked region of valid gradients in Fig. 6 center right.

After calculating the motion gradients, we can then extract motion features to varying scales. For instance, we can generate a radial histogram of the motion orientations which then can be used directly for recognition as done in [14]. But, an even simpler measure is to find the global motion orientation.

§ A.3 Global gradient orientation

Calculation of the global orientation should be weighted by normalized tMHI values to give more influence to the most current motions within the template. A simple calculation for the global weighted orientation is as follows:

where Φ is the global motion orientation, Φ ref is the base reference angle (peaked value in the histogram of orientations), $\Phi(x, y)$ is the motion orientation map found from gradient convolutions, norm(τ , δ , MHI $\delta(x, y)$) is a normalized tMHI value (linearly normalizing the tMHI from 0–1 using the current timestamp τ and duration δ), and angDiff($\Phi(x, y)$, Φ ref) is the minimum, signed angular difference of an orientation from the reference angle. A histogram-based reference angle (Φ ref) is required due to problems associated with averaging circular distance measurements. Figure 6 shows from left to right a tMHI, the raw gradients, the masked region of valid gradients, and finally the orientation histogram with global direction vector calculated. Figure 7 shows global motion directions for the movements of kneeling, walking and lifting the arms.

§ A.4 Motion segmentation

Any segmentation scheme begs the question as to what is being segmented. Segmentation by collecting "blobs" of similar direction motion collected frame to frame from optical flow as done in [13] does not guarantee that the motion corresponds to the actual movement of objects in a scene. We want to group motion regions that are produced by the movement of parts or the whole of the object of interest. A novel modification to the tMHI gradient algorithm has an advantage in this regard: by labeling motion regions connected to the current silhouette using a downward stepping floodfill, we can identify areas of motion directly attached to parts of the object of interest.

A.4.1 Motion attached to object

By construction, the most recent silhouette has the maximal values (i.e. most recent timestamp) in the tMHI. We scan the image until we find this value, then "walk" along the most recent silhouette' s contour to find attached areas of motion. The algorithm for creating masks to segment motion regions is as follows (with reference to Fig. 8):

1. Scan the tMHI until we find a pixel with the current timestamp. This is a boundary pixel of the most recent silhouette (Fig. 8b).

2. "Walk" around the boundary of the current silhouette region looking outside for recent (within dT, e.g. the time difference between each video frame), unmarked motion history "steps". When a suitable step is found, mark it with a downward

floodfill (downfill) (Fig. 8b,c). If the size of the fill is not big enough, zero out the area.

3. Store the segmented motion mask that was found (Fig. 8c,d).

4. If the boundary "walk" has not yet circumnavigated the current silhouette, go to 2.

5. Calculate the motion orientation within each mask found in 3 above (Fig. 8e).

In the algorithm above, downfill refers to floodfills that will fill (replace with a labeled value) pixels with the same value or pixels of a value one step (within dT) lower than the current pixel being filled. The segmentation algorithm then relies on two parameters: (1) the maximum allowable downward step distance dT (e.g. how far back in time a past motion can be considered to be connected to the current silhouette); and (2) the minimum acceptable size of the downward floodfill (else zero it out because the region is too small –a motion "noise" region).

The algorithm above produces segmentation masks that are used to select portions of the valid motion history gradient described in Sect. 3.3. These segmented regions may then be labeled with their weighted regional orientation. Since these segmentation masks are derived directly from past motion that "spilled" from the current silhouette boundary of the object, the motion regions are directly connected to the object itself. We give segmentation examples in the section below.

A.4.2 Motion segmentation examples

Figure 9 shows a hand opening and closing in front of a camera. The small windows capture the two motion regions and the large window is the overall motion orientation. Note that the small arrows correctly catch the finger motion while the global motion is ambiguous. Figure 10 shows a kicking motion from left to right. In the leftmost image, the hands have just been brought down as indicated by the large global motion arrow. The small segmentation arrow is already catching the leftward lean of the body at right. In the center left image the left leg lean and right leg motion are detected. At center right, the left hand motion and right leg are indicated. At right, the downward leg motion and rightward lean of the body are found.

Figure 11 shows segmented motion and recognized pose for lifting the arms into a T position and then dropping the arms back down. The large arrow indicates global motion over a few seconds, the smaller arrows show segmented motion as long as the corresponding silhouette region moved less than 0.2 s ago.

参考文献

[1] Aggarwal J, Cai Q (1997) Human motion analysis: a review. In: IEEE Nonrigid and Articulated Motion Workshop, pp 90–102

[2] Bergen J, Anandan P, Hanna K, Hingorani R (1992) Hierarchical model-based motion estimation. In: Proceedings of the European Conference

[3] Beymer D, Konolige K (1999) Real-time tracking of multiple people using stereo. In: IEEE FRAME-RATE Workshop, http://www.eecs.lehigh.edu/FRAME/

[4] Bierling, M (1988) Displacement estimation by hierarchical blockmatching. Proceedings of SPIE Conference Visual Communications and Image Processing, vol 1001, pp 942–951

[5] Black M, Anandan P (1993) A framework for robust estimation of optical flow. In: Proceedings International Conference Computer Vision, pp 231–236

[6] Black M, Yacoob Y (1995) Tracking and recognizing rigid and nonrigid facial motions using local parametric model of image motion. In: Proceedings International Conference Computer Vision, pp 374–381

[7] Bobick A, Davis J, Intille S (1997) The KidsRoom: an example application using a deep perceptual interface. In: Proceedings Perceptual User Interfaces, pp 1–4, October

[8] Bradski G, Yeo B-L, Yeung M (1999) Gesture for video content navigation. In: SPIE’99, 3656-24 S6

[9] Bradski G, Computer vision face tracking for use in a perceptual user interface. Intel Technology Journal, http://developer.intel.com/technology/itj/q21998/articles/art2.htm, Q2

[10] Bregler C (1997) Learning and recognizing human dynamics in video sequences. In: Proceedings Computer Vision And Pattern Recognition, pp 568–574, June

[11] Cham T, Rehg J (1998) A multiple hypothesis approach to figure tracking. In: Proc. Perceptual User Interfaces, pp 19–24, November

附录 B　译文

基于运动历史梯度信息的运动分割与姿态识别

摘要

本文使用了一种简单快速的方法来表达运动，将运动理解为连续的与时间关联的轮廓层次，术语上称为带时间加权的运动历史图像(tMHI)。这种表达方法不仅可以用来决定物体的当前位置，并且可以利用物体在视频场景中的运动信息来分割和测量这些运动。这些被分割的区域不是“动作块”，而是自然的连接到物体的运动部分。这种方法可以广泛的应用于姿态识别。我们用它来识别上下挥舞手臂与举手拍掌的运动，这样就可以控制一个音乐程序了。

关键词： 动作分割 光流法

§B.1　相关工作

三年前，一台PC价格2500美元，USB摄像头300美元，现在PC价格700美元以下，USB摄像头50美元，所以用于实时的运动识别就热门了。特别说明的是，随着硬件设备的日益廉价与计算能力的增强，跟踪监视系统，人机交互，娱乐领域都加大了对人体运动的研究力度。例如，监视应用期望了解什么时候一个人移动到某个危险的位置区域，交互系统需要了解输入手势的意思，娱乐应用需要分析人的动作，来更好的协助沉浸体验感觉。

最近出现了很多对实时人体动作识别的方法[8–11, 13–16, 20, 22, 33]，一些文章对基于视觉的动作识别[36]、人体动作捕获[31 32]以及动作分析[1]进行了综述。对于人体动作和手势识别最常见的框架就是分析动作参数的轨迹[6 29 35 40]、隐马尔科夫模型和空间状态模型[37, 39, 41]。

一个可用的方法是使用光流法跟踪感兴趣的点，但是计算量大，而且鲁棒性差，经常需要分类[2]与鲁棒性估计[5]，而且光流法容易被宽大、纹理不一的衣服所干扰。并且，在缺少分类的信息情况下，光流使用在帧间人的手势有变化的情况。尽管有这些困难，光流法还是成功用到了姿态识别的领域。一个可替代的方法在[13]提到了，是利用连续的图像轮廓来表达运动的模式，在每个时间点上，得到一个新的帧，这已经存在的轮廓减去一些阈值，新的轮廓被赋予最大的亮度。这种分层的轮廓叫

做运动历史图像(MHI)，MHI表达了帧与帧之间的时间特性，而将其集中在一个图像内，因此，MHI扩展了人体姿态的时间范围。

本文的组织结构如下，第2节对之前的MHI研究综述。3-5节总结了本文处理的流程。本文中我们直接在运动历史图像中加入了浮点格式的时间点，称作tMHI,我们采用Hu 运动[19]形状描述来表示当前的轮廓与位置。tMHI的梯度信息用来决定光流的法向，那么运动会根据物体的边界进行分割，并且可以得到每个区域的运动方向与速度大小。整个处理过程在例图1给出了，其中文献的各个章节部分会按照图中每一个过程的标记数字来解释，最终的结果是识别的姿态，和对应姿态的动作。图1的过程是一个在运动分析与姿态识别领域的通用方法。在第5节中对比了光流方法[9]与这种方法的的优劣。在第6节中我们用它来识别上下挥舞的手臂与举手拍掌的运动，控制一个音乐程序。第8节对本文进行了总结。

§B.2 姿态与动作的表示

算法如图1所示，依赖于对产生的感兴趣的对象的轮廓，几乎所有的轮廓算法都可以使用，例如可用的方法有立体视差或者立体深度提取[3]、红外线提取[12]、帧间差分[13]、颜色背景[6]、纹理块分割、前景分割等。本文中我们选用了简单的背景提取的方法。

尽管现在优许多复杂的背景提取的方法，我们使用了一个最简单的方法，我们从均值RGB中分离一系列的象素来作为前景，然后使用膨胀和区域增长的方法来消除噪音，得到轮廓。这种方法的限制是，在身体内部区域的运动会被忽略，例如一个面对摄像头的人在身体前方挥舞手臂，那么手臂的运动信息就会被身体遮挡。

一个可用的方法是使用多摄像头。另外一个方法是使用颜色分割区域，然后在背景轮廓上覆盖它们。

B.2.1 Mahalanobis 匹配HU运动轮廓姿态

为了识别轮廓姿态，7个更高HU运动提供了平移和放大不变的形状描述工具，因为这些运动有不同的顺序，我们必须使用Mahalanobis 距离方法用来匹配，匹配是基于静态的测量训练样本。

公式中x是运动特征向量，m是训练运动向量的均值，K是训练向量集的协方差的逆矩阵，那么如何识别这些轮廓的运动特征由下面一个例子给出，例如，训练集由5个人做五套同样的三个姿势，X，T，1-,那么第六个人没有练习过这些姿势，而进来试着做这些姿势。

列表1显示了一些姿态识别的结果，我们可以看到就算是姿态"Y" 与"T",这两个

容易混淆的姿态，识别结果相差不止一个数量级，这样可以容易的设置一个阈值就可以把它们从训练样本中识别出来，Table1 姿态识别的结果，可以看到正确模型间的距离远远小于不正确模型匹配的距离

另一个替代的方法是使用梯度柱图来分割轮廓区域[5]。

在本篇中，我们使用了一个浮点性的运动历史图像，那么新的轮廓值会被加上一个浮点的时间戳。

其中r(译注，字符无法表达，实际见图上)是当前的时间标记，6 是最大的过期常量(一般是几秒),这种方法可以摆脱系统的速度或受限制的帧的速度，就算一个特定的手势在MHI记录时速度不一样，效果也相同的。我们称作为tMHI.图中显示了一个手臂上举动作。

注意在图中留意tMHI的梯度，我们会得到手臂运动的方向，注意这些梯度矢量在梅一个边缘是与边界成直角的，在MHI中，假定我们得到一个法向量。tMHI的梯度可以使用分离的sobel算子在x,y方向空间上有效的计算出来，每个象素的梯度方向由如下公式：

我们必须小心，因为当计算梯度信息时候只有位置信息是有效的，那些非轮廓的边界在计算时不应该被包括进来，而且它们影响了结果。只有外部边界的的轮廓才应该被计算外，我们在背景反差太低（轮廓内部），太高（大的瞬间差）的情况下也不能使用梯度信息。应用以上的规则，一个带有原始梯度信息标注区域就产生了。在计算完运动的梯度后，我们就可以区分不同的运动特征了。例如，我们用一个雷达图来表示运动方向，它可以用来识别过什么。另外一个更简单的方法用来找全局的方向，下面我们会继续讨论这个。

§B.3 全局梯度方向

全局梯度方向的计算时，为了突出当下这个时刻的运动，应该考虑到tMHI 的法向值的加权，一个简单的计算公式如下：

其中0 (译注，字符无法表达，实际见图上) 是全局方向，0ref 是基查考角度（在方向柱图的最高值），0(x,y)是从梯度卷积中运动方向映射，norm(r,0,tMHI(x,y))是MHI的法向值，angDIFF是一个减去参照角度最小的带正负的角度差，因为要得到平均园距离方法，所以也需要得到基于柱图的参照角度。图4显示了原始梯度信息，标记合法后的梯度信息，最终的全局计算的柱图方向。

§B.4 运动分割

任何运动分割都涉及到要分割什么的问题，[9]提供了通过从帧间运动收集相同

运动方向块的光流的方法，但是它不能保证动作本身是否一致。我们想按照运动的部分，或者感兴趣的整个物体，来分组运动区域。tMHI 的优点就是使用轮廓逐渐填充的方法，这样可以利用这个特点来标记运动区域。我们可以由物体感兴趣的各个部分直接分辨出运动区域。

B.4.1 给物体找到运动

在构造中，时间上最近的轮廓由最大的值，我们扫描图像直到找到这个值，然后呢，沿着轮廓的边缘找到运动的区域。下面假设dT是时间差阈值，例如，帧间的时间差。这个算法创建掩码来分割运动的，见图6。

1. 扫描tMHI直到找到当前的时间戳。图6b显示了最近的轮廓象素。

2. 沿着当前轮廓向外找没有标记的运动图像，当轮廓被找到后，用前下填充标记它，如果填充区域不够大，扩充。

3. 存储找到的分割掩码

4. 继续直到循环完所有的轮廓

在上述的算法中，“downfill”是指用同样的值填充，或者每一步的值低于当前值，分割算法依赖与两个参数，

1. 最大的可允许下降的距离（当前的动作与过去的动作之间的时间有多远？）

2. 最小可接受填充面积，（因为太小会被误认为噪音，需要放大）

以上描述的算法用来选择2。3章节合法运动梯度的各个部分，这些分割的区域会被加上方向权值（见章节3）

既然运动掩码可以直接从过去的轮廓到现在轮廓”溢出”来，那么我们就可以从对象本身得到运动区域。下面给出例子。

B.4.2 动作分割例子

图8显示手张开与关闭，注意小箭头为手指的方向，这时全局方向并不确定。图9显示了一个踢腿的动作，左边手部位置的降低指出全局的方向，小箭头捕捉到身体右倾，中间的图左右腿的动作也捕获到了，在右图腿部下降与身体右倾也检测到了。图10显示了分割运动与姿态，抬起手臂到T位置，然后放下，大箭头会在几秒后指出全局的方向，小箭头会在轮廓移动的0.2秒内显示。

附录 C　源程序

```
%%pcasegmentation.m
clear all;
%dbstop if warning;
warning off all;
addpath(genpath('./helper'));
addpath(genpath('./ExperimentData'));
list = dir('./ExperimentData/');
filenum = length(list);
for i=1:filenum
    file{i} = list(i).name;
end
%initialization
K=240;
step=K-1;
l=60; %
startcut=1;
lastcut=startcut;
cuts=startcut;
direction=1;
% load the amc data
[origlen, lenaftersample, rawdata, amcdata] =
   motionloader(file, 1);
totalframes = size(amcdata,1);

e=zeros(1,totalframes);
d=zeros(1,totalframes);

while lastcut<totalframes
    flag=0;
    [signals,V, S, SS] = pcasvd(amcdata(startcut:
       direction:startcut+step,7:62),0,0.95);
    r=size(signals,1); % number of principal component
    n=size(SS);
    for i=r+1:1:n
```

```
33              e(K)=e(K)+SS(i);
34      end
35      K=K+1;
36      while flag==0 && lastcut<totalframes
37          [signals,V, S, SS] = pcasvd(amcdata(startcut:
               direction:K,7:62),0,0.95);
38          n=size(SS,1);
39          for i=r+1:1:n
40              e(K)=e(K)+SS(i);
41          end
42              d(K)=e(K)-e(K-1);
43              avg_deviation=mean(d(startcut+step+1:K));
44              standard_deviation=std(d(startcut+step+1:K));
45
46              lastcut=K;
47              if (d(K)-avg_deviation)>3*standard_deviation
48                  if lastcut>totalframes-step
49                      lastcut=totalframes;
50                  end
51                  cuts=[cuts lastcut]
52                  flag=1;
53                  startcut=lastcut;
54              end
55              if lastcut==totalframes
56                  cuts=[cuts lastcut]
57              end
58              K=K+1;
59      end
60  end
61  disp('results:');
62  allcuts = sort(cuts)
63
64  %% output the segments
65  lenallcuts = length(cuts);
66  for i=1:lenallcuts-1
67      amcfilename=sprintf('./result/seg_%d.amc', i);
68      matrix_to_amc(amcfilename, amcdata(allcuts(i):allcuts
           (i+1)-1,:) );
```

```
end
   % save ./result/seg_pos.txt allcuts -int
    fid=fopen('./result/seg_pos.txt','a+');
    fprintf(fid,'%d\t',allcuts);
    fclose(fid);
```

```
%%GMMSegmentation_mine.m
function  [INDEX,Mu, Variances cuts] =
   GMMSegmentation_mine(No_of_Clusters)
    warning off all;
    addpath(genpath('./helper'));
    addpath(genpath('./GMM'));
    addpath(genpath('./ExperimentData'));
    list = dir('./ExperimentData/');
    filenum = length(list);
    for i=1:filenum
        file{i} = list(i).name;
    end
    % load the amc data
    [origlen, lenaftersample, rawdata, amcdata] =
       motionloader(file, 1);
    No_of_Iterations=5;
    %No_of_Clusters=2;
    amcdata=amcdata';
    cuts=[];
    for no=2:No_of_Clusters
        [INDEX,Mu, Variances] = GMM(amcdata, no,
           No_of_Iterations);
        [m,n]=size(INDEX);
        for i=1:n-1
            if INDEX(i)~=INDEX(i+1)
                cuts=[cuts i+1];
            end
        end
        if size(cuts,2)==1
            continue;
        else
             cuts=[cuts n];
```

```
                fprintf('No_of_behaviors:%d\n',size(cuts,2)
                    -1);
                return;
            end
        end

disp('results:');
allcuts = sort(cuts)

%% output the segments
lenallcuts = length(cuts);
for i=1:lenallcuts-1
    amcfilename=sprintf('./result/seg_%d.amc', i);
    matrix_to_amc(amcfilename, amcdata(allcuts(i):allcuts
        (i+1)-1,:) );
end
   % save ./result/seg_pos.txt allcuts -int
    fid=fopen('./result/seg_pos.txt','a+');
    fprintf(fid,'%d\t',allcuts);
    fclose(fid);
end
```

```
%%pcasegmentation_error.m
clear all;
%dbstop if warning;
warning off all;
addpath(genpath('./helper'));
addpath(genpath('./ExperimentData'));
list = dir('./ExperimentData/');
filenum = length(list);
for i=1:filenum
    file{i} = list(i).name;
end
%initialization
K=240;
step=K-1;
l=60; %
startcut=1;
```

```
lastcut=startcut;
cuts=startcut;
direction=1;
behavecuts=[];
% load the amc data
[origlen, lenaftersample, rawdata, amcdata] =
   motionloader(file, 1);
totalframes = size(amcdata,1);

e=zeros(1,totalframes);
d=zeros(1,totalframes);

while lastcut<totalframes
    flag=0; % cut or not
    [signals,V, S, SS] = pcasvd(amcdata(startcut:
       direction:startcut+step,7:72),0,0.90);
    r=size(signals,1); % number of principal component
    n=size(SS);
    for i=r+1:1:n
            e(K)=e(K)+SS(i); % discarded value
    end
    K=K+1;
    while flag==0 && lastcut<totalframes
        [signals,V, S, SS] = pcasvd(amcdata(startcut:
           direction:K,7:72),0,0.90);
        n=size(SS,1);
        for i=r+1:1:n
            e(K)=e(K)+SS(i);
        end
            d(K)=e(K)-e(K-1);

            avg_deviation=mean(d(startcut+step+1:K));
            standard_deviation=std(d(startcut+step+1:K));
            lastcut=K;
            if (d(K)-avg_deviation)>3*standard_deviation
                if lastcut>totalframes-step
                    lastcut=totalframes;
                end
```

```
                cuts=[cuts lastcut]
                flag=1;
                startcut=lastcut;
            end
            if lastcut==totalframes
                cuts=[cuts lastcut]
            end
            K=K+1;
    end
end
[mm,nn]=size(cuts);
for i=2:nn-1
    if d(cuts(i))>1.5*d(cuts(i-1))
        behavecuts=[behavecuts cuts(i)];
    end
end
%% output the segments
lenallcuts = length(cuts);
for i=1:lenallcuts-1
    amcfilename=sprintf('./result/seg_%d.amc', i);
    matrix_to_amc(amcfilename, amcdata(cuts(i):cuts(i+1)
       -1,:) );
end
   % save ./result/seg_pos.txt allcuts -int
    fid=fopen('./result/seg_pos.txt','a+');
    fprintf(fid,'%d\t',cuts);
    fclose(fid);
```

```
%%pcasegmentation_ratio.m
clear all;
%dbstop if warning;
warning off all;
addpath(genpath('./helper'));
addpath(genpath('./ExperimentData'));
list = dir('./ExperimentData/');
filenum = length(list);

for i=1:filenum
```

```
    file{i} = list(i).name;
end

%initialization
K=240;
step=K-1;
l=60; %
startcut=1;
lastcut=startcut;
prestartcut=startcut;
prelastcut=lastcut;
cuts=startcut;
direction=1;
behavecuts=[];
% load the amc data
[origlen, lenaftersample, rawdata, amcdata] =
   motionloader(file, 1);
totalframes = size(amcdata,1);

e=zeros(1,totalframes);
d=zeros(1,totalframes);

while lastcut<totalframes
    flag=0; % cut or not
    [signals, U, V, S, SS] = pcasvd2(amcdata(startcut:
       direction:startcut+step,7:62),0,0.95);
    r=size(signals,1); % number of principal component
    n=size(SS);
    for i=r+1:1:n
            e(K)=e(K)+SS(i); % discarded value
    end
    K=K+1;
    while flag==0 && lastcut<totalframes
        [signals,U, V, S, SS] = pcasvd2(amcdata(startcut:
           direction:K,7:72),0,0.90);
        n=size(SS,1);
        for i=r+1:1:n
            e(K)=e(K)+SS(i);
```

```
46          end
47              d(K)=e(K)-e(K-1);
48              avg_deviation=mean(d(startcut+step+1:K));
49              standard_deviation=std(d(startcut+step+1:K));
50              lastcut=K;
51              if (d(K)-avg_deviation)>3*standard_deviation
52                  if lastcut>totalframes-step
53                      lastcut=totalframes;
54                  end
55                  cuts=[cuts lastcut]
56                  flag=1;
57                  if prelastcut > 1
58                       [signals1, U1, V1, S1, SS1] =
                            pcasvd2(amcdata(prestartcut:
                            direction:prelastcut,7:72)
                            ,0,0.90);
59                      D=pdist2(U1(:,1:r)',U(:,1:r)','
                            euclidean');
60                      Dd=diag(D);% distance between
                            observer i and i
61                      distance=sum(Dd)/r;
62                     if distance > 0.5
63                          behavecuts=[behavecuts lastcut];
64                     end
65                  end
66                  prestartcut=startcut;
67                  prelastcut=lastcut;
68                  startcut=lastcut;
69              end
70
71              if lastcut==totalframes
72                  cuts=[cuts lastcut]
73              end
74              K=K+1;
75      end
76  end
77
78  %% output the segments
```

```
79 lenallcuts = length(cuts);
80 for i=1:lenallcuts-1
81     amcfilename=sprintf('./result/seg_%d.amc', i);
82     matrix_to_amc(amcfilename, amcdata(cuts(i):cuts(i+1)
       -1,:) );
83 end
84    % save ./result/seg_pos.txt allcuts -int
85     fid=fopen('./result/seg_pos.txt','a+');
86     fprintf(fid,'%d\t',cuts);
87     fclose(fid);
```