基于隐曲线的速度插值算法研究与实现*)

李 丹 卢正鼎 徐海银 胡利安
(华中科技大学计算机科学与技术学院 武汉 430074)

摘 要 本文主要研究了隐曲线在行为动画中的应用,提出并进一步研究了隐曲线在行为动画中的任务指定作用。提出了基于隐曲线的速度插值算法。隐曲线用于指定行为动画的运动路径,速度曲线用于描述运动的时间分布。基于角色的运动路径和运动速度,速度插值技术生成与速度映射的插值位置点,进而通过逆向运动学技术,可以求解得到动画角色的骨架序列。同时对本文提出的隐曲线速度插值算法进行了实现和分析。通过设置不同的运动路径和速度曲线,速度插值技术可直观、快速地满足用户的具体要求创建不同的运动序列,且运动复用性强。

关键词 隐曲线,速度插值,行为动画

Research and Realization of Velocity Interpolation Algorithm along Implicit Curves

LI Dan LU Zheng-Ding XU Hai-Yin HU Li-An
(College of Computer Science & Technology, Huazhong University of Science & Technology, Wuhan 430074)

Abstract The use of implicit curves in motion animation is investigated in this paper. Implicit curves for task specification in motion animation is proposed and further explored. An approach and algorithm to velocity interpolation along implicit curves is proposed. An implicit curve is used to specify the motion path while a speed profile curve is adopted to describe the motion timing. Based on the motion path and motion speed, velocity interpolation technique is utilized to generate interpolation points from which skeleton sequences for character animation are obtained by inverse kinematical technique. This algorithm has been implemented and analyzed. Given the different motion path and motion speed, velocity interpolation technique allows easy creation of a wide variety of motion sequences that specifically meet the needs of an animator and good reusability.

Keywords Implicit curve, Velocity interpolation, Motion animation

1 引言

在许多实时应用如电脑游戏以及虚拟环境中,生成具有真实感的角色运动是一个很大的挑战。当前,关键帧技术依然被广泛使用于角色行为动画的设计与制作中,并出现了很多相关的角色行为动画的运动控制方法,如双插值技术、运动弯曲、运动混合以及运动映射[1~4]。这些方法在参数空间已经较为成熟。但随着隐曲面在计算机图形动画的建模、变形和渲染等运用中获得的巨大成功,隐曲线在行为动画中的应用也越来越受到重视。

角色行为动画既可以在参数空间也可以在事件空间生成动画帧序列。参数空间中的行为动画就是指在参数空间中对角色进行参数化表示,然后直接控制参数可以获得动画序列[5,6]。参数空间中最主要的行为动画方法就是关键帧技术,即每一个角色的自由度函数都是独立地进行插值生成插值运动帧。传统的关键帧插值方法如线性插值,包括 Lagrange 插值、Newton 插值和样条插值等作为最简单的插值形式,只计算角色位置插值点,而不考虑角色的不同运动速度,使得运动形式单一且复用性较差。为了解决这些问题,更好的插值技术被用来生成与角色运动速度相关的插值运动。在文[7]中,Steketee 提出的双插值技术引入了运动参数、时间、帧编号三个变量,构造了两个样条曲线,运动样条表示帧编号是时间的函数,而位置样条表示运动参数是帧编号的函数。保持位置样条不变,改变运动样条,即可生成不一样的帧序列。在文[8]中,Witkin 提出了运动弯曲的方法,其中引入了时间弯曲,同样构造了两个函数即帧姿态曲线和时间弯曲曲线,用于帧序列的再生和重用。构造时间弯曲函数和新的运动函数,把初始的运动函数和时间弯曲函数代入求解运动函数公式中即得新的运动函数。在文[9]中,Terra 提出了运动映射的方法,给定帧的主运动曲线和一辅助运动曲线。通过给定辅助运动曲线弧长是时间的函数,建立了帧的主运动与辅助运动之间的映射关系,就能得到帧的主运动与时间的关系。这样通过设定不同的辅助运动,就可以生成不同的帧运动的轨迹。这类参数空间的行为动画方法局限性主要在于,这些方法在参数空间中较为有效,但是无法应用到隐式表达的各类运动形式中。

事件空间的行为动画通过隐式地指定角色的动作或者目标,在笛卡尔坐标系中生成动画,然后再结合逆向运动学生成直观的、高逼真度的运动序列。在事件空间中,低维的运动形态可以在高维空间表示,以提高对角色参数的控制灵活度。高维空间的行为能力意味着可以在满足用户具体要求的情况下快速地创建各种不同的运动序列。因而,相比现有的在参数空间进行的行为动画技术,事件空间的行为动画具有更多的优点。

*)国家自然科学基金项目(项目编号 50305007)资助;湖北省自然科学基金项目(项目编号 2006ABA092)资助。**李 丹** 博士生,主要研究领域为计算机图形学、计算机动画;**卢正鼎** 硕士,教授,主要研究领域为信息安全、多媒体技术;**徐海银** 博士,副教授,主要研究领域为计算机图形学、多媒体技术;**胡利安** 硕士生,主要研究领域为计算机图形学。

本文研究了隐曲线在角色行为动画中的应用，组织如下：第 2 节以理论推导的方式提出基于隐曲线的速度插值算法，第 3 节给出了实验仿真演示以及相关的分析，最后给出结论。

2 基于隐曲线的速度插值技术

角色行为动画的速度插值的基本流程如图 1 所示，分为三个阶段：输入阶段、计算阶段和输出阶段。(1)输入阶段：首先对角色进行建模；同时对角色的运动路径进行规划，在本文中角色的运动路径由隐曲线指定；速度曲线在该阶段引入，用来描述运动的时间分布。(2)计算阶段：根据用户输入的运动路径和速度函数，计算机能够采用速度插值技术自动生成角色模型所在帧的目标位置和时间，并结合逆向运动学，求解得到每一帧各关节参数的自由度。(3)输出阶段：将求解得到的参数序列最终转化为角色模型在运动过程中的动作姿态序列输出。至此，在给定运动路径和速度曲线的情况下，速度插值技术可以用来生成动画中角色逼真的不同运动效果。

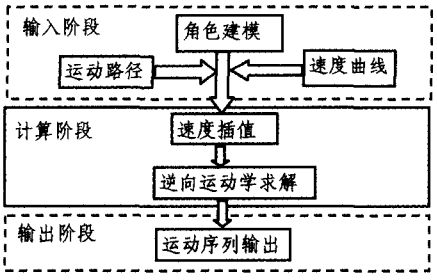


图 1 速度插值流水线

2.1 运动路径

由于隐曲面的两大优点：(1)光滑性高；(2)空间点与隐曲面的相对位置明确、易判断，使其在计算机动画中的建模、变形以及渲染等应用中都获得了广泛应用和成功。本文将隐曲面的交线所形成的隐曲线引入到计算机动画运动控制的应用中，用来描述动画角色的运动路径。如图 2 所示，两个隐曲面 $g(x,y,z)=0$ 和 $q(x,y,z)=0$ 相交形成一条如公式(1)所示的隐曲线：

$$\begin{cases} g(x,y,z)=0 \\ q(x,y,z)=0 \end{cases} \tag{1}$$

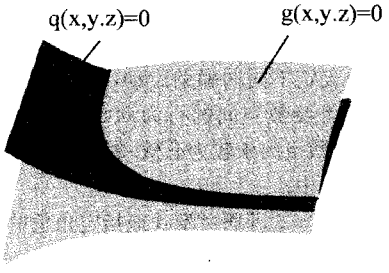


图 2 隐曲线的表示

2.2 速度曲线

在传统的关键帧插值技术中，插值位置沿着运动路径分布，没有包含任何关于关键帧的时间分布的信息。在本文的研究中，引入速度曲线来描述运动的时间分布。速度曲线体现了运动的性质和特点，既直观又便于生成不同的具有真实感的行为动画。如图 3 所示，设运动速度为 $v=v(t)$，v 是关于时间的函数。速度曲线描述了运动速度与时间的关系，可以用插补和混合得到的样条曲线来表示。

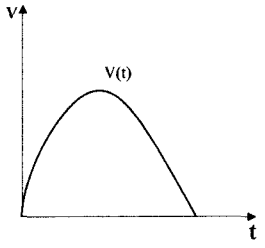


图 3 速度曲线

2.3 隐曲线速度插值算法

速度插值的目标是获得沿运动路径的一系列与速度有关的插值点位置序列。相对于传统的插值算法，速度插值的插值点位置序列可以由给定的速度曲线控制。随着速度曲线的变化，插值点位置序列也会在运动路径上发生变化。

假定在曲线(1)表示的运动路径上，k 和 s 分别表示单位矢量和隐曲线的弧长，$k=\left[\frac{dx}{ds}\quad\frac{dy}{ds}\quad\frac{dz}{ds}\right]$。公式(1)关于 s 求导得：

$$\begin{cases} kG_1=0 \\ kQ_1=0 \end{cases} \tag{2}$$

其中，$G_1=[g_x\quad g_y\quad g_z]$。$Q_1=[q_x\quad q_y\quad q_z]$。$k$ 为单位矢量，由公式(2)可以求解得到：

$$k=\frac{G_1\times Q_1}{|G_1\times Q_1|} \tag{3}$$

设 c 表示隐曲线的曲率矢量，则 $c=\left[\frac{d^2x}{ds^2}\quad\frac{d^2y}{ds^2}\quad\frac{d^2z}{ds^2}\right]$。公式(2)关于 s 求导，得：

$$\begin{cases} kG_2k^T+G_1c^T=0 \\ kQ_2k^T+Q_1c^T=0 \\ kc^T=0 \end{cases} \tag{4}$$

其中，$G_2=\begin{bmatrix} g_{xx} & g_{xy} & g_{xz} \\ g_{xy} & g_{yy} & g_{yz} \\ g_{xz} & g_{yz} & g_{zz} \end{bmatrix}$

$$Q_2=\begin{bmatrix} q_{xx} & q_{xy} & q_{xz} \\ q_{xy} & q_{yy} & q_{yz} \\ q_{xz} & q_{yz} & q_{zz} \end{bmatrix}$$

解公式(4)，得

$$c=[-kG_2k^T\quad -kQ_2k^T\quad 0]W^{-T} \tag{5}$$

其中，$W=\begin{bmatrix} g_x & g_y & g_z \\ q_x & q_y & q_z \\ k_x & k_y & k_z \end{bmatrix}$

点 p 沿曲线(1)运动的速度和加速度分别表示为

$$v=\frac{ds}{dt},\dot{v}=\frac{d^2s}{dt^2} \tag{6}$$

平移速度表示为

$$\dot{p}=vk \tag{7}$$

平移加速度表示为

$$\ddot{p}=v^2c+\dot{v}k \tag{8}$$

假定$\{p_x\}$表示速度插值位置的序列，当前的速度插值位置是 $p_n(x_n,y_n,z_n)$，下一个位置是 $p_{n+1}(x_{n+1},y_{n+1},z_{n+1})$，速度插值的任务就是根据当前位置计算下一个速度插值点位置。使用二阶泰勒级数展开，有

$$p_{n+1}=p_n+\dot{p}_nT+\frac{1}{2}\ddot{p}_nT^2 \tag{9}$$

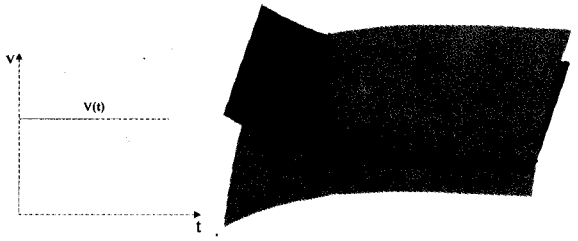
其中 T 是帧周期。

将公式(7)和(8)代入公式(9),可以得到

$$p_{n+1}=p_n+vTk_n+\frac{1}{2}(vT)^2c_n+\frac{1}{2}\dot{v}T^2k_n \qquad (10)$$

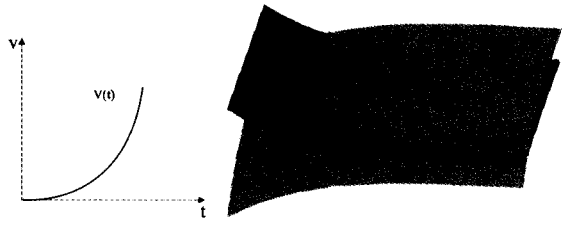
其中,k_n、c_n 分别由公式(3)和(5)计算得到,公式(6)计算出 v 和 $\dot{v}$,则公式(10)计算得到的 p_{n+1} 为时间的多项式。该多项式的系数由不同时刻的速度和加速度决定,即插值位置点求解取决于该角色运动的速度曲线。

在速度插值算法中,给定一条隐曲线作为运动路径,通过设置不同的速度曲线,可以得到曲线上不同的速度插值点序列。图 4 和图 5 分别表示线性插值和速度插值的速度曲线图和插值点序列图。



(a) 速度曲线图　　(b) 插值点序列图

图 4　线性插值



(a) 速度曲线图　　(b) 插值点序列图

图 5　速度插值

将线性插值技术作为传统插值技术的典型代表来分析其特性。图 4(a)是线性插值的速度曲线图,该运动为匀速运动。在传统线性插值中,由于插值计算不考虑速度,所以运动是否为匀速或变速均不影响插值点序列,运动形式均简化为匀速运动。因此在图 4(b)线性插值的插值点序列图中,线性插值计算所得相邻插值点之间的弧长是恒定的,不能反映其他变速运动形式。传统插值技术中的其他方法由于同样不考虑速度变化带来的插值影响,因此插值效果类似。

图 5(a)对应速度插值的速度曲线图,该运动为一变速运动。因此在图 5(b)速度插值算法的插值点序列图中,相邻插值点之间的弧长是变化的。在角色速度变化时,如该例中角色速度逐渐变大,在相同周期下,角色每个周期的位移逐渐增大,所以反映到运动曲线上为相邻速度插值点间的弧长逐渐增大。反之,若角色速度逐渐变小,运动曲线上为相邻插值点间的弧长逐渐减小。

2.4　逆向运动学求解

对于常见的具有关节链的角色动画,由速度插值算法的插值位置序列可以得到角色每一帧时末端效应器的位置集合 $\{p_n\}$,即关节链的末端效应器位置后,可以采用已有的逆向运动学算法进行求解,得到末端效应器在指定位置时各关节需要旋转的角度。

逆向运动学解决的问题是:指定末端效应器的位置 $p_n(x_n,y_n,z_n)$,反求出所有关节的自由度 θ,即 $\theta=f^{-1}(P_n)$。

2.5　运动序列输出

给定角色目标末端效应器的位置,并通过逆向运动学求解得到目角色目标各关节参数的自由度后,即可得到角色运动的参数序列,将这一组参数序列转化为角色在运动过程中的动作姿态,并最终以姿态形式输出。

若考虑角色对象的运动速度,对于同样的运动路径,在给定不同的速度曲线情况下,同样的角色可以产生不同的动作帧序列。而在传统的关键帧插值动画中,一旦确定运动路径,角色对象的动作帧序列便固定了。

3　算法实现与分析

为了验证基于隐曲线的角色行为动画的速度插值算法,对人的一只手臂沿指定路径运动进行了仿真。人的上肢骨架由大臂、小臂、手组成,可以定义为如下形式:

```
class MArm
{ private:
float position_x,position_y,position_z;//手臂根关节在世界坐标系中的位置
float angle_x[3],angle_y[3],angle_z[3];//存储 3 个关节的关节角
float length[3],radius[3];//关节的长度和半径
MJoint m1,m2,m3;//肩关节,肘关节,腕关节
public :
MArm();
void draw_arm(double s[]);//绘制手臂,实现手臂中各关节的运动层次关系
}
```

实现人体关节运动链的运动层次关系的时候,每个关节都要建立一个局部坐标系,如图 6 所示为上肢骨架关节链示意图。

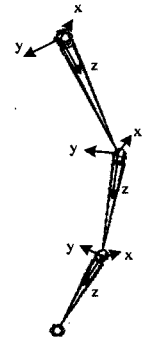


图 6　上肢关节链图

给定一条隐曲线作为上肢末端效应器的运动路径,假定频率为 24 帧/s,分别采用传统线性插值和速度插值两种方法来进行运动生成。

由速度插值算法计算出末端效应器的位置,然后使用关节链的逆向运动学,可以得到肩、肘、以及腕关节的变化,其算法实现的伪码如下:

```
初始化 P 点位置与帧周期 T;
position (n)//计算下一个插值点位置函数定义
{
根据运动轨迹计算 p 点处 k 值和 c 值;
计算速度曲线上 nT 时刻的运动速度 v 和运动加速度 a;
Δp=v*T*k+0.5*v*v*T*T*c+0.5*a*T*T*k;
p=p+Δp;//下一插值点位置由当前插值点位置与插值增量计算得到
return p;//返回下一插值点位置
}
for(n=1;n<=24;n++)
{
position(n);//计算下一插值点位置
IK(p);//利用已有逆向运动学算法求解当前位置各关节自由度
}
```

如图 7 和图 8 所示,分别显示了传统线性插值和速度插

(下转第 254 页)

经网络算法中应当采用交互对比技术，结合最优停止准则，找出“最好”的神经网络结构，这里“最好”指的是网络具有较强的泛化能力，换言之，让神经网络在达到较好的泛化能力时即刻结束训练。交互对比技术将可用数据分为两个集合：训练集和测试集。训练集用来训练网络参数，测试集用来计算预测误差。然而，这种划分有可能引入偏差，因为只能用部分数据进行训练，另一部分用于测试。一个改进的措施是按几种不同方式、标准划分原始数据，进行多次训练和测试，这样就可以通过平均所有预测误差得出无偏的预测误差估计。

考虑一个隐层节点数为 N_h，一个输出单元，有 $N>N_h$ 个训练样本的神经网络。对于每一个隐层神经元，都可以找到一个 N 维向量，这个向量的第 i 个分量是这个隐单元对第 i 个训练样本的响应。既然网络输出是隐层输出的一个线性组合，那么可能的输出向量存在于一个由 N_h 维基函数向量张成的 N 维向量空间当中。在 L^2 范数意义下，最符合输出向量 t 的隐层输出向量无非是一个 t 到 N_h 维子空间的投影。这种情况下的误差平方和为：$E=t^TP^2t$。此处，$P=I_N-\Phi A^{-1}\Phi^T$。其中，I_N 是一个 N 维单位矩阵，Φ 为基函数矩阵（$(\Phi)_{pj}=\phi_j(x^p)$），$A=\Phi^T\Phi$。比如，有 N 个数据点，$N-i$ 个用于训练，i 个用于测试。使用这种方法，预测误差的方差估计是：

$$\delta^2=\frac{t^TPi(diag(p))^{-2}Pt}{Pi}$$

其中，$P=I_N-\Phi A^{-1}\Phi^T$。

训练一个神经网络的目的之一是在过度学习和泛化能力之间寻找一个平衡点。上述交互对比的核心思想就是在训练网络的过程中，以一个独立的数据集为基准来检查算法的进展变化过程。网络在训练的样本数据上和选择集的样本数据上最初性能总是相同或者近似的。对于比较简单的神经网络系统，如前馈神经网络不但对训练样本的模拟程度较好，而且对未训练样本的拟合效果也比较理想，即网络泛化能力较强。而对于比较复杂的系统，要想使神经网络对系统的模拟程度提高，在增加训练样本的同时也要增加网络的隐节点个数，这样就使网络变得复杂。因此，在训练网络时，不必使训练误差达到尽可能的最小，而给出一个适当的允许误差，这样虽然网络对训练样本的拟合程度不是很好，但网络的泛化能力增强。即先从一个神经元开始学习，每次循环采用使网络产生最大泛化误差的输入作为新的隐层神经元。在具体实现中，可以采用通过计算矩阵和学习输出向量之间的相对误差的办法来选择新加入的神经元，然后用剩余的样本检查新网络的误差，直到达到目标误差或最大神经元数为止。

参 考 文 献

1 李杰，韩正之. 神经网络的学习误差函数及泛化能力. 控制与决策，2000，15(1)：95～97
2 鲁子奕，杨绿溪，吴球. 提高前馈神经网络泛化能力的新方法. 电路与系统学报，1997，2(4)：69～72
3 张胜，刘红星，高敦堂. 神经网络泛化特性改善方法. 计算机应用与软件，2005(12)：12～14
4 张乃尧，阎平凡，李衍达. 神经网络与模糊控制. 北京：清华大学出版社，1998
5 武妍. 神经网络学习算法中的初始参数对泛化性能和效率的影响研究. 计算机工程与应用，2002(23)：25～27，31
6 An G. The effect of adding noise during back propagation training on a generalization performance. Neural Computation，1996(8)：643～671

（上接第 227 页）

值求解的关节链序列。传统线性插值时，关节链末端效应器沿运动路径相邻点的弧长是相等的；而速度插值时，末端效应器沿运动路径的各插值点之间的弧长随速度的变化而发生变化。由此可知，确定的运动路径下，角色对象的运动速度变化不同，相同的对象模型可以产生与真实速度相映射的不同动作帧序列。

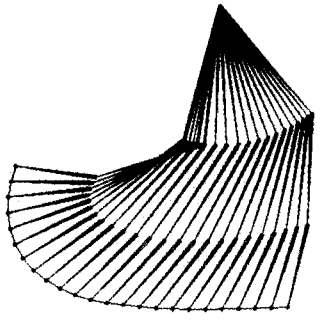


图 7 传统线性插值求解

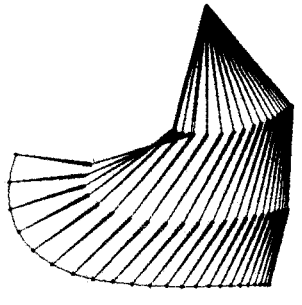


图 8 速度插值求解

结论 本文研究了隐曲线在角色行为动画中的应用。通过设置不同的运动路径和速度曲线，速度插值技术可直观、快速地满足用户的具体要求创建不同的运动序列，且运动复用性强。在未来的研究中，可以通过将角色的风格控制并入逆向运动学技术，可以得到更多真实感动画。

参 考 文 献

1 Pullen K, Bregler C. Motion capture assisted animation: Texturing and synthesis. ACM Transactions on Graphics 2002, 21(3), 501～508
2 Gleicher M. Retargeting motion to new characters. In: ACM SIGGRAPH, Orlando, Florida, 1998. 33～42
3 Witkin A, Kass M. Spacetime constraints. Computer Graphics, 1988, 22(4): 159～168
4 Kovar L, Schreiner J, Gleicher M. Footskate cleanup for motion capture editing. In: ACM SIGGRAPH Symposium on Computer Animation, 2002. 97～104
5 Rose C F, Bodenheimer B, Cohen M F. Verbs and adverbs: multidimensional motion interpolation. IEEE Computer Graphics and Application, 1998, 18(5): 32～40
6 Kochanek D H U, Bartels R. Interpolating splines with local tension, continuity and bias control. Computer Graphics, 1984, 18(3): 245～254
7 Steketee S N, Badler N I. Parametric keyframe interpolation incorporating kinetic adjustment and phrasing control. In: ACM SIGGRAPH, San Francisco, 1985. 255～262
8 Witkin A, Popovic Z, Motion warping. In: SIGGRAPH'95, 105～108
9 Terra S C L, Metoyer R A. Performance timing for keyframe animation. In: Eurographic/ACM Symposium on Computer Animation, 2004. 253～258