多种动态模拟和复杂三维动画控制的框架结构*

王裕国 潘 峰 胡静倩

(中国科学院软件研究所 北京 100080)

摘要 本文提出了一种新的适用于多种动态模拟和复杂三维动画控制的面向对象的框架结构,它采用混合层次模型,其角色按部件层次组织其静态结构,角色中包含有决定其形状、样式和行为变化的属性,它按类的层次结构定义,属性的动态赋值通过引进的动态约束机制实现.该框架结构已用于SIMUKIT——用于多种动态模拟视觉效果自动生成的集成工具环境中,并在SGI工作站上初步实现,实验表明该框架结构对解决多种动态模拟生成是可行的.

关键词 面向对象,框架结构,动态约束,动态模拟,动画.

中图法分类号 TP391

随着三维图形及动画的广泛应用,人们对复杂的基于物理的三维动画、动态模拟及各种现象的真实模拟的需求越来越强烈,这给研究开发满足这些需求的应用软件提出了新的课题.它既要解决动态模拟中对象的形体、样式及行为随时间变化的控制方法,又需处理动态模拟的真实视觉效果的生成.前者完全依赖于具体应用,后者涉及景物及动画实体的造型和组织、实体动态变化的控制与表达、各帧画面的真实感绘制及人的交互干预处理等.

显然,采用传统的关键帧动画技术和造型(Modelling)—动画(Animation)—绘制(Rendering)相分离的管道式框架结构已难以实现这些需求.近些年来,国际上针对上述某些需求已开展了一些研究工作,取得一些很好的实验性成果.例如,Inventor[1~3],Fifth Dimention[4]和GROOP[5]是几种用于三维图形编程的面向对象的工具箱,它们都给出各自通用的可扩充的面向对象的框架结构,通过定义一系列的类来表述三维场景,以使应用程序能将它们的数据集成到图形对象中.

上述框架结构均属基于类—实例—继承模式,与此不同,BROWN大学的UGA[6],采用原型—委派(Prototype—Delegation)机制,使其对象的所有属性和行为都可动态地改变,这种改变依赖消息传送协议,这是第1个支持直接表达其值随时间和输入信息而变的三维编程框架结构.

* 本文研究得到国家自然科学基金资助.作者王裕国,1941年生,研究员,主要研究领域为计算机图形学,多媒体技术.潘峰,1969年生,实习研究员,主要研究领域为计算机图形学.胡静倩,女,1970年生,实习研究员,主要研究领域为计算机图形学.

本文通讯联系人:王裕国,北京100080,中国科学院软件研究所

本文1996-09-19收到修改稿

本文提出了另一种面向对象的框架结构，一种新的适用于多种动态模拟和复杂三维动画控制的面向对象的框架结构，并可以此为核心，创建一个便于多种动态模拟视觉效果自动生成的集成工具环境，从而实现将依赖于具体应用的动态模拟控制算法的研究与模拟视觉效果的实施相分离，使动态模拟领域中的研究人员专心致力于控制算法的研究，然后，使用该集成环境将这些算法集成于控制对象中，并方便地产生所需要的动态模拟视觉效果.

该框架结构具有以下特点：

(1)基于参量化角色模型结构和动态约束机制(表达角色动态变化控制规则)的动态模拟描述. 所有影响动画角色的形状、样式和行为的参量均作为属性对象按类的层次结构结合于角色对象中，其值决定了角色的形状、样式和行为. 通过引进的动态约束机制，以一种统一的形式给出角色的属性值的约束和控制关系，使对象间的接口是通过可直接获取和更改的数值而不是通过方法实施.

(2)采用的面向对象的机制是一种混合层次模型. 它是在基于类的模型上借鉴原型的特色来构造框架结构，场景及角色按部件层次结构组织，角色中的属性按类的层次结构组织，它们联合构成角色的静态结构描述，采用动态约束机制，将角色的动态变化的控制方法封装在按类层次结构表示的控制对象中，通过动态约束机制，形成统一的反映角色属性之间、角色与控制器之间的约束控制关系的有向无环图的语义结构，并据此实施角色属性的动态赋值.

(3)具有良好的可扩充性.

(4)易于在其上建造一个可视化界面环境，使设计者方便、直观、声明式地描述其动态模拟活动.

目前，该框架结构已用于 Simukit(用于多种动态模拟视觉效果自动生成的集成工具环境)中，并在 SGI 工作站上，在 Inventor 的支持下初步实现，实验表明该框架结构对解决多种动态模拟生成是可行的.

1 框架结构的设计

1.1 为什么提出混合层次模型

类—实例—继承(Class-Instance-Inheritance)与原型—扩展—委派(Prototype-Extension-Delegation)是两种不同的面向对象的机制. 目前广泛应用的是类—继承机制，但这种机制也存在一些缺陷，如对象和方法的封装是固定的，运行中不能动态地增删和改变；对象的类型和继承关系是预先设定的，运行中不能改变；继承法则固定不变. 因此，在动态模拟环境中(在这里，活动对象的行为甚至形状都会随时间而变化)很难单纯用这种经典的类来描述这些活动对象. 基于原型—委派的机制则便于动画实体动态的构造和对其行为变化的描述，但至今这种机制尚未成为主流. Stein[7]在 1987 年对委派和继承在功能上进行了比较，认为两者在功能上是同等的，委派可以实现继承的功能，反之，也存在一个继承的"自然"模型可获得委派的全部特性. 基于此，我们提出在基于类的模型上借鉴原型的特色来构造适用于多种动态模拟的混合层次模型的框架结构.

1.2 参量化的角色模型结构

角色是动态模拟设计者所建立的活动实体，是该框架结构的基本单元. 角色的外形、样

式和行为的动态变化在该框架结构中被归纳为角色的属性及其动态求值. 这些属性包括反映角色的形状和定位状态的几何形体属性、几何变换属性，反映角色外观及绘制方式的表面特征属性，用于具体动态模拟中角色动态变化控制的应用属性——如质量、惯量、速度、加速度、力、力矩、碰撞点等等.

角色的模型结构中包括以下几个组成部分：

①参量表示的角色属性，它们作为对象，以类定义，按类的层次来组织，在该类中包含可动态修改这些属性值的成员函数. 应用属性有一个基类，设计者根据其具体应用需求，由此派生出所需的应用属性类，然后添加到角色中.

②角色可以是一个复合对象，可由其它角色组成，故在角色模型结构中可包含子角色.

③动态约束控制关系函数用于表达动态模拟过程中该角色属性值的动态变化规则.

④添加属性成员的函数，角色模型按部件层次结构组织，它可看成是一个原型. 角色初始为一个空对象，通过动态添加属性产生所需求的实体.

此外，角色还可以作为模板(Template)，通过拷贝继承生成其它角色. 图 1 是角色的类描述的一个示例.

```
class Actor
{
SoSeparator * root;
SoTransform * transform;
SoMaterial * material;
SoShape * shape;
SoNode   * ApplicationProperty;
Actor * child;
public:
//constructor
  Actor();
//destructor
  ~Actor();
void addtransform(SoTransform   *t);
void addmaterial(SoMaterial * m);
void addshape(SoShape * s);
void addapp(SoNode * a);
void addchild(Actor * c);
SoSeparator * getroot();
SoTransform * gettransform();
SoMaterial * getmaterial();
SoShape * getshape();
SoNode * getapp();
Actor * getchild();
void constraint(SoNode * dest,controller * executecontroller);
void conditionalconstraint(controller * conditioner,SoNode * dest,controller * executecontroller);
void conditionalconstraint(Bool C, SoNode * dest,controller * executecontroller);
};
```

图 1 角色类描述示例

1.3 动态约束机制

在框架结构中解决角色属性的动态赋值问题，关键是给出一种通用的统一的表达方式，描述在任一时刻角色属性求值规则的问题. 基于时态逻辑的语言表达方式是一种通用、有效的方法，但软件开销很大，基于我们的目标是建立适用于多种而不是各种动态模拟和复杂三

维动画控制的框架结构.为处理简单起见,我们采用的方法是通过对多种动态模拟的分析、归纳,提出了一种动态约束机制,即将模拟中用到的各种控制方法封装于各种控制器对象中,将角色的属性值赋值规则用一种统一的格式,经由约束控制关系函数来表达,通过这种声明式描述,建立动态约束控制关系的有向无环结构图,并据此实现角色属性的动态赋值.

1.3.1 控制器对象

该框架结构将所有与时间相关的行为描述函数或算法封装在控制器对象中,用抽象类Controller建立起各类控制器在框架结构中的统一接口,这些控制器包含从简单的低级控制,如交互设备控制(Valuator)、轨迹控制(Track),通常的运动学控制到复杂的模拟算法,如逆运动学、动力学、逆动力学、有限元方法、基于约束的方法、目标驱动控制及随机过程控制、柔性体受力后形变控制等等.图2给出控制器对象的类层次说明.

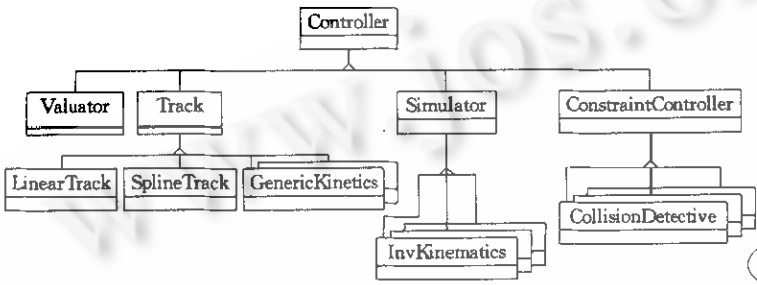

图2 控制对象的类层次说明

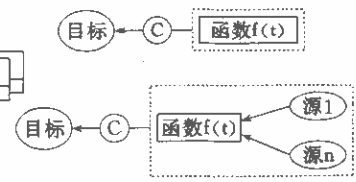

图3 带0个或n个源的单向约束(若含C为条件约束)

在控制器对象类中包含如下一些信息:控制源(一组影响受控目标的角色或角色属性,它们以链表形式出现在类的成员中)、结果域(约束控制函数执行的结果)、约束控制函数、结果获取函数以及构造函数、析构函数等.

1.3.2 动态约束控制关系

目前,框架结构仅提供单向约束控制表达(如图3所示),目标是指受控的角色或角色属性,$f()$为约束控制函数,源$i(i=1,\ldots,n)$为约束或控制该目标的角色或角色属性,条件C(如果存在)是约束控制函数可有效执行的条件表达,一般地,它由约束条件控制器给出.在我们提出的框架中,约束控制函数$f()$和源被封装在控制器对象中(图3中虚框所示),以控制器对象名标识.动态约束控制关系由约束控制函数表示.

对无条件约束控制,表示为:

Constraint(SoNode * dest, Controller * executeController)

对条件约束控制,表示为:

ConditionalConstraint(Controller * conditioner, SoNode * dest, Controller * executeController)或

ConditionalConstraint(Bool C, SoNode * dest, Controller * executeController)

这里dest为目标对象或其属性指针.

1.3.3 光源、摄像机、场景对象及时钟

光源、摄像机也被视为角色对象,有它们自己的属性,其属性也可以受其它角色或控制器的约束和控制.场景(Scene)是由角色组成,通过*add*(),*delete*()分别插入和删除角色.

场景包含一时钟对象(Timer),全局控制其时间的演变(包含起始、终止和时间间隔等属性).间隔属性值可以有缺省值,由系统根据每秒需显示的帧数设置,也可以受到其他对象的约束控制.这在时间连续的动态模拟进行离散化处理时是需要的.例如,碰撞模拟中时间间隔要受到碰撞检测控制器的约束.

2 框架结构及其实现

2.1 框架结构的构成

图4是以该框架结构为核心所构造的一个用于多种动态模拟视觉效果自动生成的集成工具环境——SIMUKIT的系统结构.图的上半部分是动态模拟描述工具(详见文献[8]),下面部分是该框架结构的构成.其中系统内部数据结构主要包括两部分:动态约束关系图和场景结构图.动态模拟设计者使用SIMUKIT系统提供的动态模拟描述工具给出场景、角色、光源、摄像机及角色动态变化规则性描述,经由代码生成器产生该框架结构中的角色对象模型、光源对象模型、摄像机对象模型、场景对象模型、控制器对象模型,并自动生成和维护动态约束控制关系图.在时间控制器的控制下,通过对动态约束控制关系图的扫描处理实现对角色属性的动态赋值,并利用IRIS INVENTOR的绘制机制,通过对场景结构图的扫描完成场景的绘制,进而实现动态模拟过程.

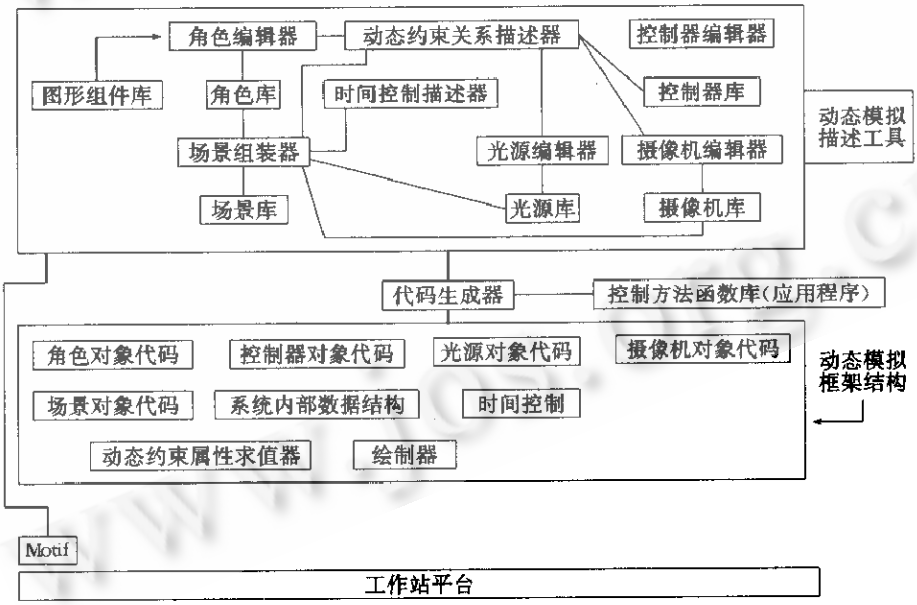

图4 SIMUKIT系统结构

2.2 框架结构实现中的关键技术

该框架结构在SGI工作站上,在IRIS INVENTOR支持下进行了初步实现.该实现只是为验证该框架结构的可行性而进行的,因此,造型、光源、摄像机、绘制是在IRIS INVENTOR所提供的相应功能基础上实现的,该框架结构的核心是在IRIS INVENTOR提供的类库基础上建造的各种类库.以下仅就该框架结构实现中的几个关键技术做一些简单介绍.

2.2.1 动态约束控制关系图的构成和作用

动态约束控制关系图是一种有向无环图结构,用来描述角色属性之间、角色与控制器之间的动态约束控制关系.它由无条件约束控制关系结构单元和条件约束控制关系结构单元组成.前者由受控对象、执行控制器及其源(可空)组成.后者由受控对象、条件控制器及其源、执行控制器及其源组成.只有当条件控制器所产生的条件满足时,执行控制器才起作用.一个受控对象可以受多个源的控制,我们用链表的形式来描述这些源,同时,增加入度信息来指明受控对象的受控状态.

2.2.2 动态约束控制关系图的建立

该框架结构初始化时,自动生成并初始化动态约束控制关系图.当用户使用Constraint()函数或ConditionalConstraint()函数来描述角色内在的或外在的或与其它控制器之间的约束控制关系时,该框架结构自动向动态约束控制关系图中填加相应的结构单元.

2.2.3 角色属性的动态赋值及场景的绘制

该框架结构在IRIS INVENTOR的Render类基础上开发了自己的绘制类Display.在Display的处理中,通过对动态约束控制关系图的扫描,修改场景中受到约束控制的角色属性值,从而实现角色属性的动态赋值.我们在常规的有向无环图拓扑排序算法基础上稍加改进,实现了对该关系图的扫描算法.由于条件控制器和执行控制器的源可以有重复的,因而在扫描时同等看待,均作为图中一个结点处理,通过对入度信息的检查可以免去大量重复计算.

框架结构根据时钟对象的时间间隔值控制场景的动态显示,每次绘制之前,要扫描一次动态约束控制关系图.在扫描该图的同时,修改相应的角色属性值,从而实现角色属性的动态赋值,然后利用IRIS INVENTOR的绘制机制完成场景的绘制.

2.3 实 例

我们利用该框架结构作了2个简单实例.

例1:台面(带有4个壁)上有3个不同颜色、不同大小的球,实现给定初速度后的球在台面上运动(不考虑摩擦力)和与台壁及其它球可能的碰撞效果的动态模拟.

在该动态模拟中,用到5种控制器,即球的运动位置控制器PCT(根据球的速度属性,计算当前时刻球的位置)、碰撞发生条件控制器CDC(检测此时刻是否发生碰撞,若发生,则置该控制器的RESULT域的值为TRUE,并置源中各发生碰撞的角色的碰撞发生应用属性值为TRUE;否则置为FALSE)、无碰撞发生条件控制器!CDC、碰撞响应控制器CRC(计算当前时刻该球的位置并修改球的速度属性值,恢复时钟对象TIMER的时间间隔属性值为缺省值)、时间间隔控制器T(根据当前场景中各球的位置,确定时钟对象TIMER的时间间隔属性值),用到2个有条件约束:

ConditionalConstraint(! CDC,ball→transform,PCT);

ConditionalConstraint(CDC,ball-.transform,CRC).

例2:给定初速度的球在方盒中运动,一束光(呈锥形)跟踪该运动的球.

在该动态模拟中,球在方盒中运动模拟与例1类似,而光束跟踪运动的球的动态模拟涉及到一个约束控制器CCT(根据球的位置计算光束的方向)和一个无条件约束:

Constraint(Light→transform,CCT)

3 结 论

我们提出的混合模型机制统一地刻画了动态模拟中静态和动态行为特征,在此基础上设计和初步实现了可作为适用于多种动态模拟和复杂三维动画自动生成的集成化开发环境的核心的面向对象的框架结构.实验表明,该框架结构对多种动态模拟的自动生成是可行的,虽然目前实现的还只是一个雏型,但对用于多种动态模拟自动生成的集成开发环境的研究开发而言,是一种新的、有意义的探索.

参考文献

1 Iris Inventor Programming Guide, 1992.

2 Strauss, P S, Carey R. An object-oriented 3D graphics toolkit. Computer Graphics (SIGGRAPH'92), 1992, **26** (2): 341～349.

3 Wernecke J. The inventor mentor programming object-oriented 3D graphics with open inventor. Release 2, 1994.

4 Turner R *et al*. An object-oriented methodology using dynamic variables for animation and scientific visualization. In: Proc. Computer Graphics International'90, Springer Verlag, 1990. 317～328.

5 Koved L, Wooten W L. GROOP: an object-oriented toolkit for animated 3D graphics. In: Proc. OOPSLA'93, 1993. 309～325.

6 Zeleznik R C *et al*. An object-oriented framework for the integration of interactive animation techniques. In: Proc. SIGGRAPH'91, Computer Graphics, 1991, **25**(4): 105～111.

7 Stein L. Delegation is inheritance. In: OOPSLA'87, 1987.

8 王裕国,胡静倩,潘峰.用于多种动态模拟视觉效果自动生成的集成工具环境 SimuKit.计算机学报(图形学专刊), 1996.

A FRAMEWORK FOR MULTIPLE DYNAMIC SIMULATION AND COMPLEX 3D ANIMATION CONTROLS

WANG Yuguo PAN Feng HU Jingqian

(*Institute of Software The Chinese Academy of Sciences Beijing* 100080)

Abstract In this paper, the authors present an object-oriented framework which adapts to a variety of dynamic simulation and complex 3D animation controls. The framework is based on a hybrid model. The static structures of actors are combined as part hierarchies. The attributes of actors determine their shape, appearance, time-varying behavior and are defined as class hierarchies. The attributes's dynamic assignment is implemented by dynamic constraint mechanism. This framework has already been used in SimuKit-an integrated environment for automatic generation of visual effect for multiple dynamic simulation. It is basically implemented on SGI workstation. It is feasible to implement multiple dynamic simulation by this framework.

Key words Object-oriented, framework, dynamic constraint, dynamic simulation, animation.

Class number TP391