XYZ系统在动画设计中的应用*

唐小平[1,3] 唐稚松[1] 马华东[2,3] 赵 琛[1]

[1](中国科学院软件研究所 北京 100080) [2](北京邮电大学计算机工程系 北京 100088)
[3](中国科学院计算技术研究所CAD开放实验室 北京 100080)

摘要 XYZ系统是一套以线形时序逻辑为基础的CASE环境系统,其核心是时序逻辑语言XYZ/E. XYZ/ADL(animation description language)是XYZ系统在动画领域的应用.马华东博士采用基于时序逻辑的动画描述模型TLAD(temporal-logic-based animation description model),提出了动画剧本描述语言(SDL/A).本文介绍的XYZ/ADL语言是以XYZ/E的框架为基础扩充SDL/A而成.它使用XYZ/E中面向对象程序设计的基本结构——代理机构(Agent)作为基本描述单元.代理机构由通讯进程和相应的包块构成,包块处理对象封装,进程处理通讯.从而能更方便地刻画动画过程中的复杂、多层次的角色(包括父子角色)的运动特征及多角色之间的同步协调关系,并且在统一的逻辑框架下实现了计算机动画中运动对象的行为抽象和运动的抽象描述,更具一般性.

关键词 XYZ/E,ADL,代理机构,角色,条件元,动画描述模型.

中图法分类号 TP311, TP391.

XYZ系统是以时序逻辑为基础的适合多种程序设计和不同层次的抽象描述的CASE环境,由时序逻辑语言XYZ/E及一组基于该语言的CASE工具集组成,其目标是提高软件可靠性和软件生产率.[1-3]时序逻辑语言XYZ/E作为XYZ系统的基础,既是一个逻辑系统又是一门程序设计语言.它是面向软件工程设计的,以适应软件工程如下领域:逐步求精的设计方法;速成原型方法;抽象描述和验证;图形程序设计;包括分布式和基于共享存储器的并发程序设计;并发进程与数据模块相结合意义下的面向对象程序设计;满足各种工程要求,从非形式化到形式化描述的平滑过渡;在统一的形式框架下表示不同层次的抽象描述和可高效执行的算法过程;在编译的编译、源代码到源代码的转换中起中间语言的作用.在XYZ系统中,以XYZ/E作为各个CASE工具之间接口的公共语义基础.因而,这些工具可以可靠地结合在一起,在许多领域得到广泛的应用.本文以动画设计作为一个具体领域说明XYZ系统如何应用于多媒体程序设计.

对计算机的动画设计的研究大致可分为3个部分,即造型、运动描述与合成绘制技术.其中动画描述和运动的控制是计算机动画研究的主要问题.一个复杂的动画场景包含背景和多个运动对象(角色),每个运动对象具有静态属性(物理属性、几何属性以及运动特征)与动态属性,动态属性涉及到与其它运动对象之间的动作同步协调关系和整个动画场景的动态规划,这种运动对象间的行为并发描述是系统应考虑的重要因素.

由此,参考面向对象的方法,考虑到动画设计的逐步求精过程和动画中各个抽象对象以及运动对象间的并发描述,我们运用XYZ/E中面向对象程序设计的基本结构——代理机构(Agent)来描述动画过程.XYZ系统中面向动画方面应用的子语言称为XYZ/ADL(animation description language).这充分考虑到XYZ/E的优点,它具有很强的表达能力,不仅能描述不同层次上的逻辑结构和抽象描述,而且还能表示一般语言的各个方面.此外,XYZ系统作为CASE工具集[4],包括如下5组工具:①验证和一致性检测工具;②形式描述逐步求精和快速原型工具;③用于结构化程序设计的图形工具;④语言转换工具;⑤模块管理和系统集成工具.我们将ADL嵌入XYZ系统中,这样可以利用其中的相应工具,极大地促进动画研制过程.

---

* 本文研究得到国家863高科技项目基金、国家自然科学青年基金和国家"九五"攻关项目基金资助.作者唐小平,1972年生,硕士生,主要研究领域为软件环境,多媒体技术.唐稚松,1925年生,研究员,博士导师,中国科学院院士,主要研究领域为计算机科学,软件工程.马华东,1964年生,博士,讲师,主要研究领域为图形学,计算机动画,多媒体技术.赵琛,1967年生,助理研究员,主要研究领域为软件环境和工具.

本文通讯联系人:唐小平,北京 100080,中国科学院软件研究所

本文1996-11-19收到原稿,1997-02-24收到修改稿

1 XYZ/E及代理机构(Agent)简介

时序逻辑语言XYZ/E是基于Manna和Pnueli(1996年图灵奖获得者)提出的线形时序逻辑系统[5]设计而成的.它既是声明式逻辑系统,又是命令式程序设计语言.该语言的显著特性就是它能在统一的逻辑框架下以其适当的形式表示普通高级语言的基本性质、不同层次的抽象描述以及其它一些程序设计风范.为了满足不同的需要,XYZ/E被分成不同的子语言,例如XYZ/BE,XYZ/SE,XYZ/PE等,分别以符号%ALG,%STM,%RLE开头,其后接相应语言的控制结构序列,其基本构成单位是条件元(Conditional Element).[3]

代理机构(Agent)是XYZ/E中面向对象程序设计的基本结构,它是由一个通讯进程(Process)和相应的包块(Package)构成.包块内定义变量和操作,进程扮演包块的管理者,包块内的变量和操作自动输出到其伙伴进程中.包块处理对象封装,进程处理通讯.代理机构吸收了传统面向对象设计的精华,将其模糊不清的静态语义和动态语义分开,由包块和进程来分别描述,其中包块管理传统的对象封装和继承,伙伴进程负责与外界的代理机构或者进程进行通讯.于是,在代理机构中,对象操作的静态语义和进程的动态语义和谐统一.代理机构的形式如下:

```
AgentDeclPart ::= %AGT [ Agent; ... ; Agent ]
Agent ::= [ Process, Package ]
Process ::= %PROS[   ProcessName(ParameterDeclPart) ==
                     [
                     ProcessBody
                     ]
                     [WherePart]

Package ::= %PACK[   PackageName;
                     [TypeDeclPart;]
                     [FatherNamePart;]
                     SharedVarDeclPart;
                     OperationDeclPart
                     [WherePart (constraint; rela. seman.)
```

这里,[...]表示[X]::=X|空,其中,Process中的ProcessName和Package中的PackageName必须相同,以便成对构成一个Agent,它们之间的联系通过自动的输入输出关系在编译的时候建立.

FatherNamePart声明此代理机构的父亲的名字,用以实现继承关系.

2 动画描述语言ADL的设计思想

XYZ系统在动画方面的应用是由马华东博士首先提出的.为此基于时序逻辑建立了动画描述模型TLAD[6],研制实现了SDL/A语言.[7]本文扩充文献[7]所介绍的SDL/A的结构,运用代理机构来作为基本描述单元,使之更符合时序逻辑语言XYZ/E的框架,也更符合面向对象语言结构的要求,从而更加具有一般性.

2.1 描述单元

动画场景包含背景、环境光和多个运动对象(角色),每个运动对象具有静态属性与动态属性.静态属性包括物理属性(质地、颜色、表面光洁度等)、几何属性(形状、大小、位置等)以及运动特征(动画行为以及涉及到的坐标系、坐标变换、背景、时间段等);动态属性涉及到与其它运动对象之间的动作同步协调关系和整个动画场景的动态规划.XYZ/E中的代理机构实现了静态语义与动态语义的和谐统一,我们运用它来描述动画中的各个角色(包括光源和摄影机),用代理机构的包块来描述角色的静态属性,而用其伙伴进程来描述角色的动态属性.

于是动画描述语言ADL将动画中的行为客体,诸如角色(Actor)、光源(Light)、摄影机(Camera),抽象为一个个代理机构(Agent),其相应的静态属性封装在代理机构的包块中.由于代理机构具有适于抽象描述和逐步求精的优点,提供了面向各领域应用所需知识的一种自然表示,所以用户用以设计动画(写剧本)时,只需自然地从一个个代理机构考虑起,这样将一个复杂的动画场面逐步分解求精,并且在要修改原剧本时,只需修改相应的代理机构,不增加修改的复杂性,便于维护.

动画角色(也包含光源和摄影机)的描述单元的形式为:

```
%AGT [
ACTOR ActorName
[
  %PROS[
       ActorName()==[
```

```
            %CHN ChannelDeclaration
            %LOC [VarDeclaration]
            %RLE[
              ce1|ce2|...|cen
            ]
        ]
    ],
    %PACK[
        ActorName:
            FATHERNAME: parent
            %VAR [AttributeDeclaration]
            %PROC [ActorActionDeclaration]
    ]
  ]
];
```

其中,(1) ce_i 即为条件元(Conditional Element),其形式为:$LB=S_i \wedge P_i \Rightarrow @_i(Q_i \wedge act_i \wedge BG=b_i \wedge TR=t_i \wedge LB=SE_i)$. 其中,$@_i$ 是一时序逻辑算子;P_i,Q_i 都是合式的时序逻辑公式;Q_i 、act, 为当条件 P_i 满足时角色的行为;BG 是场景的背景描述;TR 给出行为的执行时间;LB 是一个指明当前状态值下一状态的控制元.

(2) parent 是子角色的父角色的名字,以此实现父子角色.

(3) AttributeDeclaration 用以定义角色的属性数据.

(4) ActorActionDeclaration 用以定义角色行为.它的描述形式为:

```
%PROC [
  actoraction1;
  ......
  actoractionn
]
```

它给出动画中角色动作的详细描述,由若干动作函数(Actoraction)构成.一个动作函数具体刻画角色的一个运动特征,它的描述项有:平移参数、旋转参数、变比参数、参考坐标系等.每个动作函数具有若干虚参变元,由变换来刻画.变换包括平移变换、旋转变换、伸缩变换等.变换定义了4种运动类型:F,L,S,D,分别表示不变、线性变化、样条曲线变化和以数据文件给定变化过程.一个动作函数描述实例如下:

```
walk(%INP x1,y1,z1:FLOAT; %INP x2,y2,z2:FLOAT)==[
%ALG [LB=START⇒ $OT (x1,y1,z1→x2,y2,z2) ∧ $OL ∧ $OLB=L1;
      LB=L1⇒ $OReference (refman) ∧ $OF ∧ $OLB=STOP
     ]]
```

(5) 这里 ACT,BG,TR 为我们引入的动作原语的描述,它们合在一起构成了行为客体的动作原语描述.除了行为客体的第1次动作原语描述和背景发生变化外,背景描述(BG)部分可以省略,省略的缺省值为前一时间段的背景;执行时间描述(TR)也可省略,缺省值为1.一般来说,TR 给出行为动作的执行时间,但有例外.譬如一人在等待一辆车的到来,这人"等待"这一动作的执行时间省略或者给出了一执行时间.但实际上,"等待"这一动作的执行时间取决于车到来所用的时间,这就涉及到两个行为客体之间的动作同步协调问题.因此"等待"这一动作的执行时间,将持续到车到来为止.其中的协调关系,通过消息传递来实现.何时结束,由车向人发出信号而定.

2.2 同步协调

ADL 中,对每一个角色(包含光源和摄影机)的运动行为的描述为一个代理机构,角色之间行为的同步协调关系是通过进程之间通道的消息传递来实现的(上面已提到).通道是可以动态决定的,XYZ/E 中,有两条与通道操作相关的通信命令,即输出命令(写通道)ch!y 和输入命令(读通道)ch?x,这里 ch 是从输出进程到输入进程之间的通道的名字,y 是输出进程的输出表达式,x 是输入进程的输入信号变量.传递的信号既可以是不同角色之间的同步协调信号,也可以是角色之间、父子角色之间的命令应答信号.

譬如,猫追老鼠,老鼠逃进洞或者逃跑过程中被猫逮住吃掉的动画过程:猫出现,老鼠开始逃跑;老鼠逃跑,猫开始追赶.老鼠何时开始逃,取决于猫的出现.猫一出现,老鼠警觉,于是开始逃.其间的同步,由猫发出信号,老鼠接收;而猫何时开始追赶老鼠,取决于它何时发现老鼠.老鼠开始逃跑,发出在逃信号,猫发觉(即接收到这一信号),于是开始追赶.而老鼠逃得及时,逃进洞,猫停止追赶,或者不幸被猫逮着,猫也因而停止追赶.这一过程由老鼠发信号,猫接收.至于是老鼠逃进洞信号,还是被逮住信号,可视场面情况动态决定.此外,猫追的动作和老鼠逃的动作,其本身也需要同步协调.比如猫追的动作,怎样跨前腿,动后腿,迈左腿,抬右腿,这需要协调一致,有板有眼.实现它,是通过猫向腿

(包括前左腿、前右腿、后左腿、后右腿)发出命令信号,再由腿向猫发出应答信号的方式.

2.3 子角色

在描述复杂的动画角色时,单层次的描述单元难以胜任,由此,我们引入了多层次描述角色的方法,即以父子角色技术来描述.父子角色通过代理机构的继承机制来实现,每个子角色拥有自己的代理机构描述单元,同时继承其父角色的运动特性.子角色和父角色之间也可建立通道,动作协调关系也通过消息传递来实现.

譬如对人的描述:人具有两只上肢,上肢又有手,每只手又具有5个手指,这就构成了多层次的角色关系.这种父子角色关系的描述可发展到任意层次,以描述复杂角色的运动特性.

3 XYZ/ADL系统实现

XYZ/ADL作为XYZ系统的子系统已经在SUN及SGI系列工作站上实现,包括两大部分:剧本解释器(Script Interpreter)和动画生成器(Animation Generator).

3.1 剧本解释器

剧本解释器将以如前所述的描述方法写的动画描述体(剧本)分析处理,并对角色的动作规划和角色之间的动作协调关系进行动态模拟执行,得到动作描述的具体含义和各个角色的并发行为的动作原语序列,生成中间代码.基本处理流程见图1.

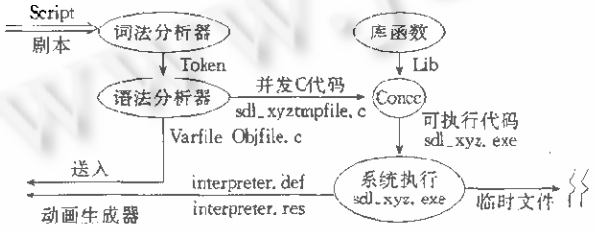

图1　剧本解释器流程图

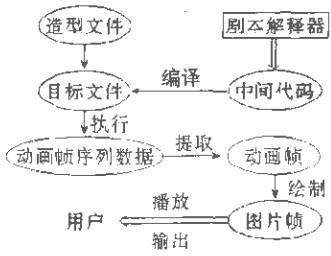

图2　动画生成器处理过程图

3.2 动画生成器

动画生成器将剧本解释器生成的结果文件经编译处理,合成所有角色的运动,得到动画的关键帧,然后插值生成所有关键帧间的其余中间帧,从而产生动画帧数据序列,经过绘制,最终生成动画画面.处理过程见图2.

3.3 实现环境

系统实现的硬件平台是SUN及SGI系列图形工作站.系统实现的软件环境包括:UNIX操作系统;面向对象的程序设计语言C++和并发程序设计语言并发C;GL图形库;SGI提供的图形图象处理工具.

4 例　子

猫和老鼠问题:如前所述,猫追老鼠,老鼠逃进洞的动画过程.

为节省篇幅,这里只列出猫和老鼠的行为体的同步描述部分.

```
%AGT [
ACTOR cat
    [
    %PROS[
    cat()==[
      %CHN ch1(*,mouse):INT,ch2(*,mouse):INT;
      %TYPE SIGNAL=(APPEARSIGNAL,ESCAPE,ESC_SUCCESS,ESC_FAILURE);
      %LOC i,j:SIGNAL]
      %RLE[
        LB=START⇒$OLB=scat1;
        LB=scat1⇒appear()∧$OBG=B0∧$OTR=1∧$OLB=scat2;
        LB=scat2⇒$Och1! APPEARSIGNAL∧$OLB=scat3;
        LB=scat3⇒$Och2? i∧$OLB=scat4;
        LB=scat4∧(i=ESCAPE)⇒runafter()∧$OBG=Bc∧$OTR=4∧$OLB=scat5;
        LB=scat4∧(i/=ESCAPE)⇒$OLB=scat3;
        LB=scat5⇒$Och2? j∧$OLB=scat6;
        LB=scat6∧(j=ESC_SUCCESS)⇒disappointed()∧$OLB=scat7;
```

```
        LB=scat6∧(j=ESC_FAILURE)⇒snatched()∧$OLB=scat7;
        LB=scat7⇒stoprunning()∧$OLB=scat8;
        LB=scat8⇒$OLB=STOP
      ]
    ]
  ],
  %PACK[
    cat:
      %VAR[
        cat.dat:SHAPE;
        refo:REFERENCE
      ]
      %PROC[appear()==...]
  ]
  ];

ACTOR mouse
  [
  %PROS[
    mouse()==[
      %CHN ch1(cat,*):INT,ch2(*,cat):INT;
      %TYPE SIGNAL={APPEARSIGNAL,ESCAPE,ESC_SUCCESS,ESC_FAILURE};
      %LOC[i,j:SIGNAL]
      %RLE[
        LB=START⇒$OLB=smouse1;
        LB=smouse1⇒befree()∧$OBG=B0∧$OLB=smouse2;
        LB=smouse2⇒$Och1?i∧$OLB=smouse3;
        LB=smouse3∧(i=APPEARSIGNAL)⇒escape()∧$Och2!ESCAPE∧$OBG=B0∧$OTR=3∧$OLB=
          smouse4;
        LB=smouse3∧(i!=APPEARSIGNAL)⇒$OLB=smouse1;
        LB=smouse4⇒<>analysiscondition∧$OLB=smouse5;
        LB=smouse5∧(escaped=$TRUE)⇒enterhole()∧$Och2!ESC_SUCCESS∧$OLB=smouse6;
        LB=smouse5∧(escaped=$FALSE)⇒$Och2!ESC_FAILURE∧besnatched()∧$OLB=smouse6;
        LB=smouse6⇒$OLB=STOP
      ]
    ]
  ],
  %PACK[
    mouse:
      %VAR[
        mouse.dat:SHAPE;
        refo:REFERENCE
      ]
      %PROC[befree()==...]
    ]
  ];
```

5　相关工作的比较

基于角色理论而开发的动画语言系统如 DIRECTOR,特点是将角色定义为一个可发送和接收消息的对象,对消息的反应采用命令式语言.其它的命令式动画制作系统如 Macromedia Animation Studio[6],对动画过程的描述由一系列的指令构成,例如:

```
on enterFrame
    puppetSprite 2, TRUE
    setstartCast = the number of Cast "mm01i"
    repeat with n=0 to 16
        set the CastNum of Sprite 2 = startCast + n
        updateStage
        wait 2
    end repeat
    puppetSprite 2, FALSE
end
```

与之相比较，我们的 XYZ/ADL 采用时序逻辑语言，便于逐步求精和抽象描述，且集成于一个 CASE 环境中，提供了许多辅助设计工具. 其它的一些商业动画软件如 3D Studio，Renderman 等，这类软件采用交互式的图形环境，交互功能强大，功能较侧重于动画造型；而我们的 XYZ/ADL 以代理机构为基本描述单元，对动画过程的动态规划及并发描述提供了较为理想的支持.

6 结 论

本文着重阐述了以代理机构(Agent)为基本描述单元的动画描述语言 ADL 的设计思想，还论述了 XYZ/ADL 的系统设计和实现策略. 由此可以看出代理机构为动画描述语言提供了一种理想的方法. 这样可以将一个复杂的动画场景逐步分解求精. 代理机构具有对象封装性、继承性以及动态规划性，可以在许多领域得到更广泛的应用.

制作动画是一个复杂、繁琐的过程，深入的工作要将 ADL 及其制作环境进一步完善及实用化，考虑描述过程的交互性、实时性，并和 XYZ 系统的其它工具(如验证工具等)更有机地结合，推向更广阔的应用.

参考文献

1 Tang Zhi-song. Towards a unified logic basis for programming language. In: Proceedings of the IFIP Congress, 1983

2 Tang Zhi-song. The design philosophy of XYZ system. Journal of Software, 1990,1(1):47～55

3 Zhao Chen, Tang Zhi-song. A temporal logic language oriented toward softwate engineering. Journal of Software, 1994,5(12): 1～15

4 Tang Zhi-song, XYZ Group. An outline of the XYZ system. In: Pnueli A, Lin H eds. Logic and Software Engineering, International Workshop in Honor of Chih-Sung Tang, Beijing, Singapore: World Scientific, August 1995. 299～311

5 Manna Z, Pnueli A. Temporal logic for reactive system. 1991

6 Ma Hua-dong, Liu Shen-quan. A temporal logic based animation description model TLAD. Chinese Journal of Computers, 1995,18(11):814～821

7 Ma Hua-dong, Liu Shen-quan, Tang Xiao-ping *et al*. The design and implementation of animation script description language SDL/A. Journal of Software, 1996,7(7):385～393

8 Gary Chapman. Macromedia Animaiotn Studio. New York: Random House, Inc., 1995

The Application of XYZ System in the Animation Field

TANG Xiao-ping[1,3] TANG Zhi-song[1] MA Hua-dong[2,3] ZHAO Chen[1]

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

[2](*Department of Computer Engineering Beijing University of Posts and Telecommunications Beijing* 100088)

[3](*CAD Laboratory Institute of Computing Technology The Chinese Academy of Sciences Beijing* 100080)

Abstract XYZ system is a CASE Environment based on the linear time temporal logic proposed by Manna and Pnueli, whose kernel is the temporal logic language XYZ/E. XYZ/ADL (animation description language) is its application in the animation field. Dr. Ma put forward the SDL/A (animation script description language) oriented towards the TLAD (temporal-logic-based animation description model). The XYZ/ADL language introduced in this paper is expanded from SDL/A on the base of the XYZ/E framework. It takes the object-oriented programming basic structure of XYZ/E——Agent as the main description unit. The agent is constituted of a process and a corresponding package. The package part serves for encapsulation and the process part for communication. Therefore it is more convenient to describe the action feature of the complicated hierarchical actors (including the parent-child actors) and the concurrency among the multi-actors, and in a unified logic system it implements the actor's action abstraction and the movement's abstract specification, which is adaptive more generally.

Key words XYZ/E, animation description language, agent, actor, conditional element, animation description model.

Class numbers TP311, TP391