文章编号:1002-6886(2005)06-0004-02

OPENGL 中工业机器人三维模型的实现

张永林,陈富林
(南京航空航天大学 机电学院,江苏　南京　210016)

摘要:本文阐述了在基于 VC++ 6.0 上的 OPENGL 中绘制工业机器人的一种方法。此方法利用 3D Studio MAX 给工业机器人建模并将其保存为 3DS 格式的文件,然后利用程序将 3DS 文件直接输入到 OPENGL 场景中。
关键词:3D Studio MAX　VC++　OPENGL　工业机器人

The Realization of Three-dimension of Industrial Robot In OPENGL

ZHANG Yong-lin, CHEN Fu-lin

Abstract: This paper introduces a method of drawing Industrial Robot in OPENGL based on VC++6.0. The method models Industrial Robot in 3D Studio MAX and saves the model as file of 3DS, then imports the 3DS file into OPENGL through program.

Key words: 3D Studio MAX; VC++; OPENGL; Industrial Robot

1　引言

OPENGL(即开放型性图形库 Open Graphics Library)是近几年发展起来的一个性能卓越的三维图形标准,它源于 SGI 公司为其图形工作站开发的 IRIS GL,在跨平台移植过程中发展成为 OPENGL,后成为工业标准[1]。通过 OPENGL 技术人员可以创建交互式应用程序,实现具有逼真效果的三维图形图像,从而在要求高度模拟真实世界的诸多领域中大显身手。但是 OPENGL 不提供绘制复杂三维物体模型的高级命令,使用 OPENGL 的程序员必须利用由一系列的点、直线、和多边形等几何图元的组合来建立期望的模型,即 OPENGL 并不是专长于建立模型。例如要在 OPENGL 中模拟出一个与现实世界中非常接近的工业机器人,只能用其提供的基本图元经过一系列的组合构造出来,而这一过程是相当复杂和漫长的,而且效果特别不好。

而本文阐述的方法是利用 3D Studio MAX 建模功能强大的特点,先将工业机器人的模型构造出来,然后存储为 3DS 格式文件,通过程序再将 3DS 格式文件输入到 OPENGL 场景中,流程如图 1。此方法充分利用 3D Studio MAX 的建模功能,从而避免在 OPENGL 中建模的问题,避开了 OPENGL 的弱点,同时又利用了 OPENGL 能实时交互的优点,避开 3DStudio Max 不能实时交互的弱点,为以后的工业机器人动态仿真提供更真实的模型和实时交互的基础。

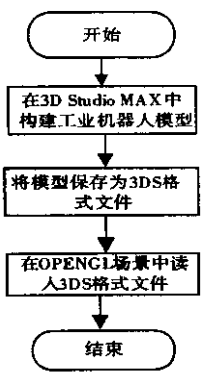

图1

2　利用 3D Studio MAX 建模

此工业机器人的模型是根据我们柔性制造实验室的工业机器人而建的,有五个自由度,在实际应用中,安装不同的末端执行器工业机器人可以实现不同的功能。由于第五个自由度是末端执行器的旋转,而仿真过程中不考虑其末端执行器,只考虑通用性,所以仅用一圆柱体表示。

首先在 3D Studio MAX 中画出底座,并将其坐标设置为(0,0),在此基础上依次画出各个关节。画各个关节时一定要注意各个关节坐标的设置,因为坐标的设置在动态仿真时极为重要,涉及到工业机器人的位姿矩阵。还要设置工业机器人的层级树,层级树在动画中作为联动的基础。设置层级树的方法就是选择连接命令,从底座开始依次连接,每一个关节作为下一个关节的父物体。这样当父物体运动时,子物体就跟着运动,而子物体运动时,父物体不会跟着运动。另外还要注意锁定坐标轴,因为大部分自由度都是只能围绕一个坐标轴旋转,这就需要锁定其坐标轴,使其不能围绕其它坐标轴旋转。关键技术见文献[2]。建立工业机器人的基本模型如图 2 所示。

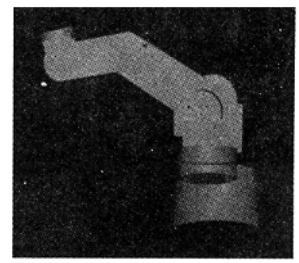
图2　3D Studio MAX 中工业机器人的模型

3　输出 3DS 格式文件

在 3D Studio MAX 中建好工业机器人的模型后,选择 File 菜单下的 Export 将物体模型保存成 3DS 格式的文件。

3DS 文件由许多块组成,每个块首先描述信息类别,即该块是如何组成的。块的信息类别用 ID 来标识,块还包含了下一块的相对位置信息。3DS 二进制文件中的数据是按低位在前,高位在后的方式组织的。块的前两项信息分别是:块的 ID 和块的长度(也即下一个块相对于该块的字节偏移量),块的 ID 是一个整型数,而块的长度是一个长整型数。每一个块实际上是一个层次结构,不同类型的块,其层次结构也不相同。3DS 文件中由一个基本块,其 ID 是 4D4D,每一个 3DS 文件的开头都是由这样一个块构成,它包含了整个文件。因此

作者简介:张永林(1979—),男,河南新乡人,在读硕士研究生。
收稿日期:2005-8-11

这个块的大小就是文件的大小减去主块的大小。还有两种主块 :3D 编辑程序块和关键帧块,前者的 ID 是 3D3D,后者的 ID 是 B000。3D 编辑程序块表明编辑程序数据开始,也就是物体的形体数据定义从此处开始。关键帧块表明开始定义关键帧信息。相对于物体的层次结构,场景中给予每个物体一个数字以标志其在场景树中的顺序。相应地,3DS 文件中也用相同地方法标志了物体在场景树中的位置。读取文件时,会得到一系列得物体数字标识。如果当前数字标识比前一个大,那么当前物体是前一个物体的子物体 ;反之,当前物体是前一个物体的父物体。此工业机器人的 3DS 文件部分内容如图 3 所示。

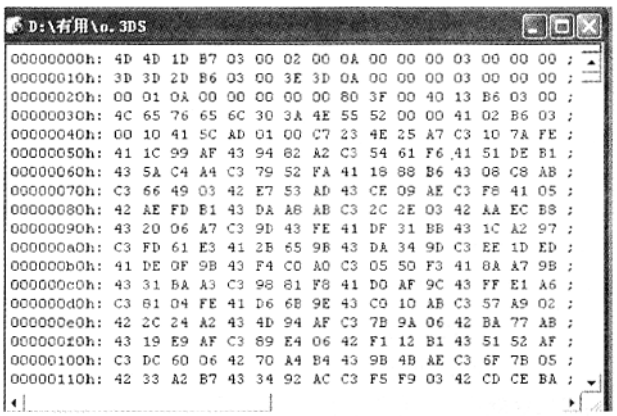

图3 工业机器人的3DS文件内容

4 将工业机器人的 3DS 文件直接输入到 OPENGL 场景中的实现

用 OPENGL 绘图之前,在 VC ++ 6.0 和 OPENGL 的基础上建立三维绘图场景。概括起来,在单文档中绘制 OPENGL 图形的主要步骤和关键技术是:

1)在单文档窗口的创建过程中,设置好显示的像素格式,并按 OPENGL 的要求设置好窗口的属性和风格;

2)首先获得 Windows 设备描述表,然后将其与事先设置好的 OPENGL 绘制描述表联系起来;

3)调用 OPENGL 命令进行图形绘制;

4)退出 OPENGL 图形窗口时,释放 OPENGL 绘制描述表 RC 和 Windows 设备描述表 DC[3]。

而将 3DS 直接输入到 OPENGL 场景中的关键技术就是 3DS 文件内容的读入和调用 OPENGL 命令进行 3D 图形的绘制。

此程序充分利用了 VC ++ 面向对象的编程思想 :编写一个通用的类 CTriObject,将 3D 对象的每一个构件看作这个通用类的对象;将 3DS 文件的读入过程也编写成一个类 C3dsReader。这样,此程序的所有的操作都对类的对象操作。在基于 VC ++ 的 OPENGL 中此程序主要通过三个类来完成,分别是 C3dsReader、CTriObject、CTriList,另外还有基本的数据结构程序等。先在 VC ++ 中单击 Project 菜单,选择子菜单 Add To Project 下的 new 插入基本的数据结构定义文件,取名位 glStructure.h,在其中定义一系列的结构 Struct,用于表示 3DS 文件块的结构以及三位对象的属性,如三维对象的材质、材质库、位置矢量、RGBA 颜色、四元数、关键帧等。然后在 VC ++ 中选择 Inset 菜单下的 New Class 子菜单,在工程中插入一个新类 C3dsReader,Class Type 选择 Generic Class,该类主要用于处理 3DS 文件中的各种块的读入。用同样的方法为工程分别再插入两个类 CTriObject(Class Type 为 Generic Class)和 CTriList(Class Type 为 Generic Class),前者主要用于处理 3DS 文件中的各种对象,后者主要用于处理 3DS 文件显示中的各种列表。文献[3]给出了具体的实现技术。其中 CTriObject 类对象的主要成员变量包括对象的几何位置的坐标、对象的法向量、面的数目、对象的材质、具有材质的面的数目以及名称;主要的成员函数包括一些基本操作和 drawGL()。CTriList 类对象的主要成员变量是 CTriObject 类对象;主要的成员函数是 drawGL()。而 C3dsReader 类主要是一系列成员函数,用于读取不同的内容,将 3DS 文件中的内容读入到 CTriObject 类中,包括将块的内容读入块结构中的函数、读入字符串的函数、读入子块的函数、读入颜色定义的函数、读入顶点的函数、读入多边形的函数、读入对象所用材质的函数、读入对象数据的函数、读入材质定义的函数、验证当前文件是否是 3DS 文件的函数等。

读入 3DS 文件时,通过调用 C3dsReader 类的一系列成员函数将 3D 对象的属性读入到 CTriObject 类对象的成员变量中,采用循环遍历 3DS 文件,这样 3D Studio MAX 模型中的每一个构件就转化成一个个 CTriObject 类对象,然后将每一个 CTriObject 类对象加入到 CTriList 类对象链表中。在 OPENGL 中进行 3D 对象的绘制时,只要调用 CTriList 类对象链表的成员函数 drawGL(),就可以将 3D 对象绘制出来。而链表的成员函数 drawGL()是调用链表中每一个 CTriObject 类对象的 drawGL()。最后,将 3D Studio MAX 构建的工业机器人模型读入 OPENGL 中,效果如图 4 所示。

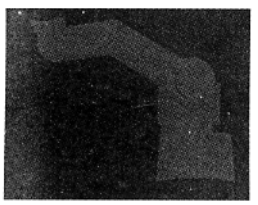

图4 将3DS文件读入OPENGL中的机器人模型

5 结束语

本文阐述了在基于 VC ++ 的 OPENGL 中绘制工业机器人的一种方法,充分利用 3D Studio MAX 的建模功能,绘制出与实际比较接近的工业机器人,输出成 3DS 格式文件。通过 VC + + 编程将 3DS 文件直接输入到 OPENGL 场景中,通过 OPENGL 的绘图命令将工业机器人绘制出来,为工业机器人的动态仿真提供了很真实的实验平台。其中将 3DS 文件读入 OPENGL 场景中的程序具有通用性。

参考文献

1 尚游,陈岩涛. OpenGL 图形程序设计指南. 中国水利水电出版社,2002.10
2 单福祺,陈润华. 如何使用 3D Studio MAX. 北京:机械工业出版社,1999.11
3 和平鸽工作室. OpenGL 高级编程与可视化系统开发. 中国水利水电出版社,2003.1