A robot simulation, monitoring and control system based on network and Java3D

Chen Yimin Zhang Tao Wang Di He Yongyi
School of Computer Engineering and Science, Shanghai University, Shanghai 200072
E-mail: ymchen@yc.shu.edu.cn

Abstract This paper introduces a robot simulation, monitoring and control system based on network and Java3D. When this system is connected with off-line programming module, users can program a robot by the legible robot language off-line, and finally download the robot instructions to the controller. When the system is connected with the robot controller via network, the status of the robot can be displayed in 3D-simulation mode, and the behavior of the robot can be monitored dynamically, and the authorized user can also control the action of the robot through the network as well. This paper describes the technique and method implementing executable interactive robot 3D simulation based on the Java platform, using Java3D for 3D visualization in combination with VRML for external robot representation in detail. This system can run on any operating system that supports JVM, and realizes the concept of "Write Once, Run Anywhere". This paper also discusses robot instruction, instruction interpreter and off-line programming, and introduces the functional actualization of the 3D graphic simulation, monitoring and control system. Fully implemented cases include using this system under linux and windows operation system, and applying to PT500 and PT600 robot.
Keywords robot controller, JAVA3D, JAVA, simulation, monitoring and control, off-line programming

基于网络和 Java3D 的机器人仿真与监控系统

陈一民 张 涛 汪 地 何永义
（上海大学计算机工程与科学学院 上海 200072）

摘 要 本文介绍了一种基于网络和 Java3D 的机器人仿真与监控系统。该系统通过网络和离线编程与仿真数据发生器相连，用户能够使用易于理解的机器人语言对机器人进行离线编程，将机器人指令下装到机器人控制器中执行；当通过网络与机器人控制器相连时，不但能够接收机器人控制器发来的当前机器人状态数据，动态地以三维图形方式进行显示，监视机器人的运动状态，而且有权限的用户还能够通过网络对机器人的动作进行远程的控制。本文论述了使用 JAVA 语言作为开发平台、以 JAVA3D 三维显示技术完成三维可视化、以 VRML 作为外部机器人模型表示、实现运行时可交互的机器人三维实时图形仿真系统的开发技术与方法。本系统能够在任何支持 JAVA 虚拟机的操作系统平台上运行，实现了“只写一次，到处运行”的思想。同时，在对机器人语言、指令解释和离线编程进行分析探讨的基础上，介绍了图形仿真和监控系统的功能实现。该系统分别在 LINUX 和 WINDOWS 操作系统下进行了实现，已成功地应用于 PT500 和 PT600 不同机器人本体上。
关键词 机器人控制器、JAVA3D、JAVA、仿真、监控、离线编程

一、引言

不论机器人在生产线中起怎样作用，从系统的观点出发，首先要考虑机器人如何与其他设备实现通信，这样才能保证机器人具有远距离控制功能,同时保证机器人在应用中发挥更大的作用。清华大学计算机系智能技术与系统国家重点实验室所承担的 863 项目“利用远程网络技术的机器人遥操作研究”正是在这样的背景下展开的。华南理工大学也正在研究基于 INTERNET 的机器人实时跟踪系统。1999 年底我们向国家教委申请并确立了“基于网络的远程机器人监控技术” 科研项目。基于网络的远程机器人在海洋探测、军事及靠人类自身不能到达的各种危险和特殊场所具有独到的作用。远程机器人监控系统能满足未来的机器人对控制器的要求。

二、机器人控制系统的原理、结构

机器人控制系统的功能模块原理如图 1 所示。整个系统分为三大部分：机器人控制器、图形仿真与监控系统、离线编程与仿真数据发生器。

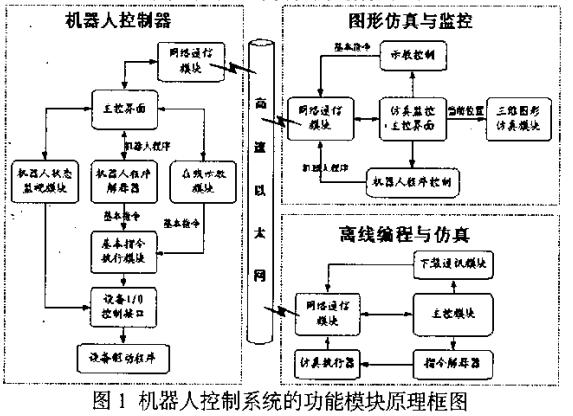

图 1 机器人控制系统的功能模块原理框图

2.1 机器人控制器 机器人控制器使用一台 Pentium III 工业控制微机（IPC）。在该 IPC 上安装了

国家教委《高等学校骨干教师资助计划》项目（705809）
上海市科学技术委员会基金支持项目（015115042）

位置控制卡，其输出直接与机器人伺服电机驱动器相连，驱动机器人本体上的伺服电机。机器人控制器提供了人机交换界面，能方便地对机器人进行示教编程、操作，建立和编辑机器人指令程序，设置机器人运行参数，动态反馈机器人的状态等。机器人指令由解释器解释执行，可将指令序列送到位置卡，驱动模块控制机器人本体完成相应动作。同时，可送至仿真执行器产生相应的实时机器人关节状态信息，这些关节状态信息可通过网络发送到图形仿真系统，完成相应的仿真动作。

2.2 离线编程与仿真数据发生器　　离线编程与仿真数据发生器使用另一台微机。其功能与机器人控制器有许多相似之处，主要差别是离线编程与仿真数据发生器不需要与示教器进行通信，也不需要去驱动位置控制卡。另外，对机器人指令的编辑功能要求比机器人控制器略高一些。经仿真模拟反复修改调试好的机器人程序，可通过网络下载到机器人控制器中执行。

2.2 图形仿真与监控系统　　由于要实时显示三维机器人动画，需要有很强的图形处理能力和较快的运算速度，我们采用了带有图形加速功能的显示卡的奔腾 III 微机作为硬件平台。它能够接收来自机器人控制器或者离线编程与仿真数据发生器的机器人状态数据，通过三维图形仿真的方式实时显示机器人运行状态，用户可随时监视机器人的运动状态。另外，具有权限的用户能够对机器人进行控制。控制方式也有两种：示教控制和机器人程序控制。

为实现通用化与低成本，课题组选择 Linux 作为操作系统平台。对于仿真系统，需实现：1）所仿真的物体的三维图形表示；2）用于场景和物体渲染的渲染器；3）程序定制的交互代码。

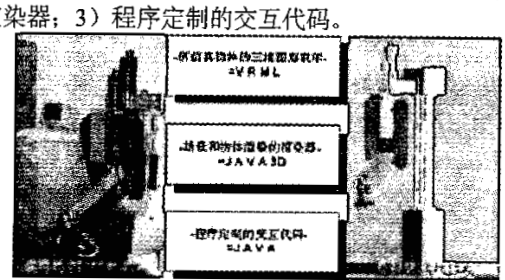

图 2 三维图形仿真实现途径

我们分别用 VRML 实现机器人的三维图形表示，用 JAVA3D API 完成场景和物体渲染，用 JAVA 完成定制的交互。如图 2 所示。

三、图形仿真与监控系统

图形仿真与监控系统的主要功能是接收来自机器人控制器或离线编程与仿真数据发生器的机器人状态数据，并将状态数据换算成机器人几何模型的关节状态数据，据此来控制机器人几何模型的显示姿态。还要实时计算光线跟踪，完成机器人运动过程的实时三维模拟。

为了使编写的应用程序能够在任何支持 JAVA 虚拟机的操作系统平台上运行，我们选用了 JAVA3D。

3.1 JAVA3D 对几何实体的表示方法　　JAVA3D 没有基本形体类，因而在程序中无法直接生成大量应用的基本形体，如 BOX、CONE、SPHERE 等。我们可以通过复杂的编程生成这些基本形体，也可以直接调用 JAVA3D 提供的辅助库 Utility 中的 geometry classes，利用它生成程序所需要的 BOX、COLORCUBE、CONE、SPHERE、CYLINDER。

同样，也可以利用 JAVA3D 提供的用于编写点、线、面的对象来创建所需的三维形体。JAVA3D 可通过编程显示出面来，面有两种：三角形和四边形，相应的对象为 Triangle 和 Quad。实际上是将任意复杂的三维实体用一个个小的多边形来近似表示，实质就是将复杂的三维实体分解为小多边形。

3.2 机器人运动姿态显示的实现　　机器人机构可以视为一种杆件机构。一般机器人都是由多个连杆组成，由关节将各个连杆连接起来，大部分机器人都是串联杆件式机器人。

如图 3 所示的机器人由机座、立柱、手臂和手部组成，通过五个转动关节连接起来。其中机座（BODY1）由关节 JOINT1 与立柱（BODY2）相连；立柱（BODY2）通过关节 JOINT2 与手臂 1（BODY3）相连；手臂 1（BODY3）通过关节 JOINT3 与手臂 2 相连 ……

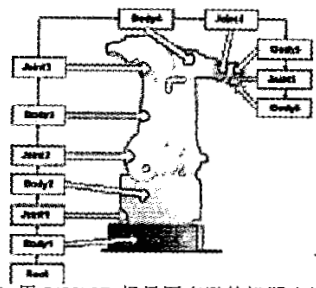

图 3 用 JAVA3D 场景图表示的机器人模型

建立机器人的场景图的过程为：

```
root = new BranchGroup();
root. addChild( body1);
body1.addChild( joint1);
joint1. addChild( body2);
body2.addChild( joint2);
joint2. addChild( body3);
body3.addChild( joint3);
joint3. addChild( body4);
body4.addChild( joint4);
joint4. addChild( body5);
body5.addChild( joint5);
joint5. addChild( body6);
```

其中各个 BODY 部分是通过 VRML97 LOADER 转换得到的机器人各个部件模型的三维场景图信息，是 BranchGroup 类型的对象，这些部分在程序运行过程中并不发生变化；各个 JOINT 部分包含机器

人关节的信息，是 TransformGroup 类型的对象，这些部分在程序运行过程中会发生变化，从而实现机器人各个关节的运动。

3.3 用定制的交互代码实现机器人三维动态仿真

用户的交互界面使用 JFC（Java Foundation Classes，Java 基础类）开发，主交互框架为标签视图，用户可以很方便地进行操作。

在没有进行任何几何变换前，缺省的观察点位于（0，0，2.41）。为实现观察机器人的三视图的效果，需要将观察点变为可以方便改变的点，需要进行几何变换。为简便起见，我们定义了 Viewpoint 类，对 Transform3D 类型的变换矩阵的设置提供了便利函数，以方便进行初始点的平移、旋转、比例变换。如正视图、顶视图、左视图的变换矩阵所对应的 Viewpoint 对象分别为：

```
viewpoint[0]=new Viewpoint(-Math.PI/2,0.0,-Math.PI/2,
                0.0,-1.0,0.0,1);   // 正视图
viewpoint[1]=new Viewpoint(0.0,0.0,0.0,
                -0.2,0.0,0.0,1);   // 顶视图
viewpoint[2]=new Viewpoint(-Math.PI/2,0.0,0.0,
                -0.4,-1.0,0.0,1);  // 左视图
```

其中，前三个参数分别对应绕 X、Y、Z 轴旋转的角度，接着的三个参数分别表示沿 X、Y、Z 轴平移的长度，最后一个参数是比例变换的大小。变换的结果可通过 getTransform 得到 Transform3D 类型的变换矩阵。

在生成 ViewingPlatform 对象后，将建立的变换矩阵作为其视图变换的矩阵即可。

```
ViewingPlatform v = new ViewingPlatform( );
v.getViewPlatformTransform();
setTransform(viewpoint[0]. getTransform());
```

3.4 图形仿真与监控子系统 Socket 通讯程序 网络通信子系统为图形仿真和监控系统构筑了信息交换平台，使图形仿真、监控子系统能够通过高速以太网发送机器人指令，或将控制信息传给机器人控制器及离线编程子系统，分别进行在线、离线方式的控制；机器人指令由机器人控制器或离线编程子系统解释执行，并将执行情况（机器人运行状态）返回给用户。图形仿真与监控子系统是作为客户机与作为服务器的机器人主控计算机进行通讯的，因此在图形仿真与监控子系统中只需要实现客户端的 Socket 程序。

在 JAVA 中有多个类允许用户创建基于套接字的网络应用程序。由于只需要使用创建客户端的程序，因此只使用了 Java.net.Socket 类，它提供了方法使得程序的设计更为容易。可以使用 getInputStream() 和 getOutputStream()分别对套接字进行读写操作，但更方便的方法是采用 Java.io 对象 BufferedReader 和 PrintStream 的实例对套接字进行读和写。具体为：首先建立一个与服务器连接的套接字，然后创建这个套接字的输入、输出流，用于与服务器进行通信。如果连接成功，接着发送使用者的用户和密码进行身份验证，通过验证后则可进行进一步的操作，如果身份验证失败则服务器端会断开套接字的连接。

客户端的通讯过程在 RobotClient 类中实现，它由 Thread 类扩展，因此通讯过程在线程中完成，这是出于对系统的性能方面的考虑，因为通讯的过程不应该影响三维图形仿真的效果。

四、离线编程与仿真数据发生器

离线编程与仿真数据发生器完成以下功能：

1. 输入、修改机器人的指令。
2. 机器人指令文件的调入、修改、保存。
3. 实时解释机器人指令，生成机器人当前的状态数据，并通过网络输送到图形仿真与监控系统。
4. 将机器人指令下载到控制器中执行。
5. 接收由机器人控制器和离线编程与仿真数据发生子系统来的反馈数据和操作命令。

4.1 机器人语言 我们构造了一种包括 31 条指令的机器人语言：

程序控制类指令：IF，THEN，FOR，NEXT，GOTO，GOSUB，RET，CASE；运动类指令：关节运动指令，MOVE，PTP；直线运动指令，LMOV，IMOV，LPTP，IPTP；圆弧运动指令，CMOV，CPTP，CIRC1，CIRC2；用于机器人焊接和搬运的指令：OUT，SETT，HANDLE，APAWD，STAWD 等。

4.2 解释器的基本原理 指令解释程序的主要功能是对机器人语言进行解释，也就是对上述 31 条指令关键字进行处理。这些关键字在解释程序内部都不是以字符串出现，而是被指定为一个数字代码。除了这些关键字外，还有另外两个特殊符号也需要指定内部数字代码，一个是行结束符 EOL（End of line），另一个是程序结束符 FINISHED。它们的内部数字代码如下所示：

```
#define MOVE            1
#define LMOV            2
.............
#define SETC            31
#define EOL             32
#define FINISHED        33
```

为了便于把一个 token 的外部表示（字符串）转换成内部表示（数），把这两种表示法对照地放在一起，存在一张名为 table 的结构数组中，如下所示：

```
struct commands{                /*keyword lookup table */
        char command[6];
        int tok;
}table[]={      /*commands must be entered uppercase */
        { "MOVE",   CM_MOVE },      /*in this table */
        { "LMOV",   CM_LMOV },
        ...........
        { "SETC",   CM_SETC   },
        { "END",    CM_END    },
        { "",       CM_END    }  /*mark end of table*/
};
```

所有的解释程序都是通过主函数内最高一级的循环实现的。每一次循环都从程序中取下一个 token，

如果该指令是单条指令则直接调用相应的函数去进行处理；如果某些功能指令两条或多条相关指令一起解释执行才能完成，需在主循环中进行判断。

机器人指令有多条，由于它们之间的互通性，现仅通过一个实例来予以说明。对于 LPTP 指令，它要用到运动学中使机器人末端沿直线移动的算法。沿空间两点之间直线移动算法的思想如下：

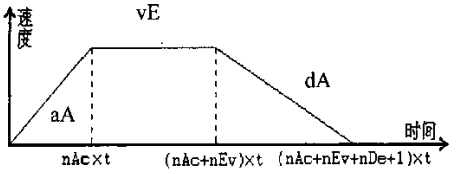

图 4 机器人末端直线运动的速度变化

已知空间两点 pt1 和 pt2，从起点 pt1 到终点 pt2 之间的直线运动是一个加速——匀速——减速运动的过程。如图 4 所示。加速度 aA、减速度 dA 和匀速运动的速度 vE 已事先给出。从起点 pt1 到终点 pt2 的运动是由许多步小的移动完成的，每移动一步的起点和终点都在 pt1--pt2 直线上。由于每一步很小，从整体上看，机器人末端由 pt1 到 pt2 是按照直线运动的。每移动一步所用的时间即插补时间是固定的，事先给出为 t。这样，根据机器人运动学理论和已知条件，可以求出加速阶段需要移动的步数 nAc，匀速阶段需要移动的步数 nEv 和减速阶段需要移动的步数 nDe。一般在走完这些步数后，机器人末端离终点 pt2 还有一小段距离 d（误差），解决办法是把误差距离作为单独的一步放到减速阶段来走。这样从 pt1 到 pt2 要走 nSum 步（nAc+nEv+nDe+1）。每一步耗时为插补时间 t。

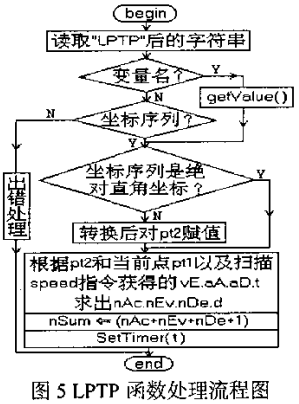

图 5 LPTP 函数处理流程图

下面我们来分析 LPTP 的实现过程。LPTP 的处理流程如图 5 所示。当命令处理程序 execLine 读到的命令为 LPTP 时转入该过程。该过程主要是对一些参数赋值，而命令的具体执行（仿真数据的产生）则由定时处理流程来完成。getValue 函数根据输入的变量名查询变量链表 m_varList，返回变量种类和坐标值。SetTimer 函数用于开启一个时间间隔为 t 的定时器，该定时器每过 t 毫秒发出一个定时器消息 WM_TIMER，一般对该消息的响应处理工作由 OnTimer 完成（也可在开启定时器时指定其它函数响应）。KillTimer 用于关闭定时器以中止其再发出定时器消息。在机器人程序解释执行的任一时刻，至多只开启一个定时器，因而 SetTimer 和 KillTimer 函数均针对同一定时器。插补次数计数器 cnt 用于记录运动过程中已经走完的步数，在响应每组定时器消息前，cnt 被置为 0，以后每响应一次定时器消息便被加 1。

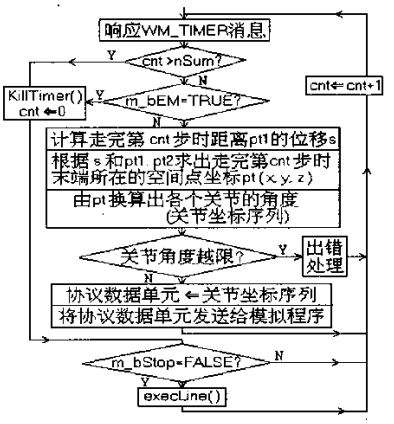

图 6 定时消息处理流程

图 6 示出了定时处理流程的流程框图，它是 WM_TIMER 消息的响应函数。在图中可看出，若 m_bEM 为真，则无论当前步数 cnt 是否大于总步数 nSum，都会关闭定时器，从而立即中止当前动作。若 m_bStop 为真，则要等到 cnt 超过 nSum 后，机器人语言程序的自动解释执行才被暂时中断（即不再调用 execLine 函数）。例如，机器人末端起初处于点 pt1，在接着执行“LPTP pt2”这一基本动作时，若 m_bEM 为真，则无论机器人末端运动到直线 pt1-pt2 上的哪一点均会立即停下来；若 m_bStop 为真，则要等到机器人末端运动到终点 pt2 后才会停下来。

上面选择了具有代表性的 LPTP 来说明机器人基本运动步骤算法的实现，其它指令运动算法的实现亦采取类似的处理过程，在此不再赘述。

五、 不同机器人本体图形仿真实现

由于我们的仿真与监控系统采用了面向对象的方法和模块化的开发，因此可以很方便地应用于对不同机器人本体进行三维的图形仿真。

在对不同的机器人本体进行仿真时，首先需要建立相应的机器人模型，这由三维建模软件完成，在我们的仿真系统中需要改动的是建立与该机器人本体对应的类。这个类也是从 Robot 基类中继承的，但它需要实际定义代表其模型的 VRML 文件和各个

关节的信息。另外场景的视点也需要改变。

在远程监控方面，因为我们的仿真与监控系统与机器人控制器之间的通讯主要是通过发送机器人指令来实现的，所以不需要进行改动。

图 7 所示是我们的仿真与监控系统应用于 PT600 机器人本体的实例。

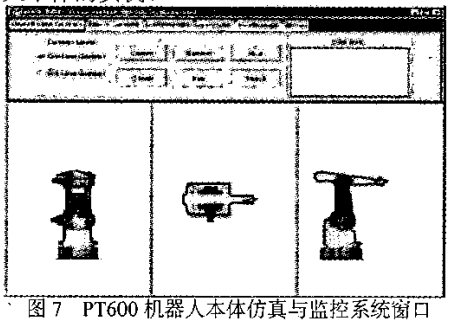

图 7 PT600 机器人本体仿真与监控系统窗口

六、不同操作系统下的仿真与监控系统

由于使用 JAVA 和 JAVA3D 作为编程的语言，JAVA 语言的平台无关性特点使我们的仿真与监控系统能够应用于各种操作系统平台，从而实现了“Write Once, Run Anywhere”，不需要作任何改动就能在 LINUX、WINDOWS 和 UNIX 等操作系统中运行。

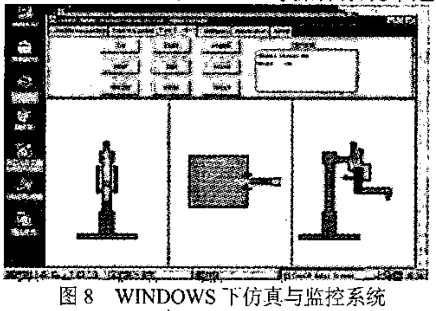

图 8 WINDOWS 下仿真与监控系统

通常在 WINDOWS 操作系统中需要安装 JDK1.2 及 JAVA3D1.2 基于 WINDOWS 的 OPENGL 或 DIRECT3D，以及用于加载 VRML 文件的加载器 vrml97.jar。图 8 所示为运行在 WINDOWS 操作系统下的仿真与监控系统。

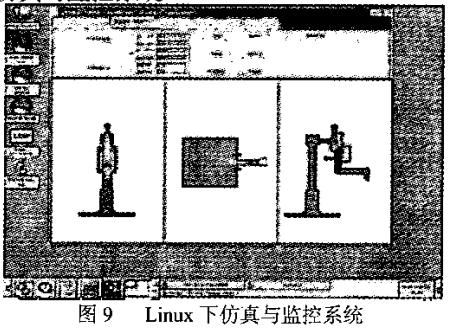

图 9 Linux 下仿真与监控系统

在 LINUX 操作系统，需要安装 Blackdown Java-Linux Team 生成的 LINUX 下的 JDK1.2 和基于 Mesa（OpenGL 在 LINUX 下的实现版本）的 JAVA3D1.2，以及用于加载 VRML 文件的加载器 vrml97.jar。图 9 所示为运行在 LINUX（RedHat 6.2）操作系统下的仿真与监控系统。

七、结论

由于引入了网络通信技术，能使机器人与其他设备方便地实现连接及通信，这样就能使机器人具有被远距离控制功能，同时保证机器人在未来应用中发挥更大的作用。本文介绍的基于网络和 Java3D 的机器人仿真与监控系统不但能够实时接收机器人控制器发来的当前机器人状态数据，将它们动态地以三维图形方式显示，实时监视机器人的运动状态，而且有权限的用户还能够通过网络对机器人的动作进行示教和机器人指令方式的控制。该系统不仅实现了 LINUX 和 WINDOWS 操作系统上的应用，而且分别用于 PT500 和 PT600 不同的机器人本体，这展示了基于网络和 Java3D 的机器人仿真、监控系统的平台无关性、通用性和实用性的特点，在性能、价格和发展潜力上都具有竞争力，代表了未来的发展方向，有很好的应用前景。

参考文献

[1] Hirzinger G., Brunner B., Koeppe R.,Landzettel K., Vogel J., Teleoperating space robots, Impact for the design of industrial robots, ISIE'97, Proceedings ofthe IEEE International Symposium on Industrial Electronics, IEEE, New York, NY, USA, 1997.3, Vol.1, pp.250-256

[2] Australia's Telerobot on the Web, http://telerobot.mech.uwa.edu. au/index.htm

[3] Shannon R.E., Introduction to the Art and Science of Simulation, Proceedings of the 1998 Winter Simulation Conference, 1998, pp.7-14

[4] Brown L., The Development of Software to Assist in Off-line Programming for Robotic Fettling of Cast Components, Industrial Robot, Vol.25, No.4, 1998, pp. 282-287

[5] Jern M., 3D data visualization on the Web, Proceedings 1998 Multimedia Modeling.MMM'98 (Cat. No.98EX200), IEEE Computer, Soc, Los., Alamitos, CA, USA, 1998, x+231 pp.90-99

[6] Vogel, J., VRML and Java3d Robots, http://www.robotic.dlr.de/Joerg.Vogel/Vrml/lib.html

[7] Vogel J., Brunner B., Landzettel K., Hirzinger G., VRML Telerobotics Client, http://www.robotic.dlr.de/Joerg.Vogel/Vrml/Rotex/

[8] VRML Consortium, The Virtual Reality Modelling Language ISO/IEC 14772-1:1997, http://www.vrml.org/Specifications/

[9] Johnson C.G., Marsh D., A robot programming environment based on free-form CAD modeling, Proceedings of the 1998 IEEE International Conference on Robotics & Automation,1998,pp.194-199

[10] Kiczales G., Lamping J., et al, Open implementation design guidelines, Proceedings of the 1997 International Conference on Software Engineering, 1997, pp.481-490