基于Kinect体感交互的多人在线虚拟实验系统

孙博文 张佳梁 蔡亚飞 郭文兰
(哈尔滨理工大学计算机科学与技术学院 哈尔滨 150080)

摘 要 为了满足多人异地进行真实感虚拟实验的需求,使用Kinect体感设备和Unity 3D引擎搭建了一个多人在线虚拟实验系统。在该系统中,使用Unity 3D引擎搭建虚拟实验场景,通过导入3D Max制作的实验器材模型进行实验搭建,并通过网络通信技术实现远距离多人在线操作。对于真实感部分,采用Kinect体感技术捕捉的身体姿势被用来控制虚拟场景中第一人称角色的走动、抓取和操作实验器材以及选取虚拟场景中的菜单。实验结果证明,Kinect姿势识别具有很高的准确性和鲁棒性,并且不容易被光照条件和复杂的背景所影响,服务器与客户端的通信对于建立远程虚拟实验系统来说足够稳定。该系统具有成本低、真实感较强的优点。

关键词 多人在线,虚拟实验,Kinect,体感交互,Unity 3D
中图法分类号 TP37 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2016.9.057

Multiplayer Online Virtual Experiment System Based on Kinect Somatosensory Interaction

SUN Bo-wen ZHANG Jia-liang CAI Ya-fei GUO Wen-lan
(School of Computer Science and Technology, Harbin University of Science and Technology, Harbin 150080, China)

Abstract In order to meet the needs of allowing more than one person conducting a certain virtual experiment at the same time in different places, we proposed a multiplayer online virtual experiment system built by Kinect somatosensory equipment and Unity 3D engine. In this system, the virtual experiment scenes are built by Unity 3D engine and experimental equipments are made by using 3D Max which is well-known as a modeling software. The multi-player online function is completed by the network communication technology. To make users feel real, the body gesture caught by Kinect somatosensory technology can be used to control first-person-character to walk around, grab experimental equipments and manipulate the menu in the virtual scene. The practical result shows that the Kinect gesture recognition has good accuracy and robustness, and is hardly affected by light situations and complex background. At the same time, the communication between the server and client is stable enough to build a remote virtual experiment system, and the system can achieve a high immersion with lower cost.

Keywords Multiplayer, Virtual experiment, Kinect, Somatosensory interaction, Unity 3D

1 前言

在本虚拟实验系统的虚拟环境中,用户可以进行一些化学实验来增强对化学原理的理解和实验步骤的掌握。用户不仅可以通过视觉、听觉等方式与虚拟世界进行交互,还可以通过做出一些肢体动作直接投身于虚拟实验中,并体会这种沉浸的乐趣。这种基于虚拟现实技术的虚拟实验系统在实验教学中起到了很重要的作用[1]。

通常情况下,虚拟实验系统需要借助价格昂贵的硬件支持才能完成人机交互的任务,比如数据手套、3D头戴式显示头盔、位置追踪器等设备。尽管这些设备的效率很高,并且能够带来很好的沉浸感受,但是不菲的价格以及相对于某些目的的专用性使得它们难以在普通院校推广。本文提出了一种基于普通台式电脑建立廉价虚拟实验系统的方法。该系统可以收集人体动作数据,并且提供了一种能够对虚拟实验设备进行体感操作的方法。它摆脱了鼠标键盘的束缚,使通过自然习惯行为来完成虚拟实验任务成为了可能[2]。

Kinect传感器是微软推出的体感游戏外设,它为游戏世界提供了一种无需手持设备的交互方式[3]。借助于红外线收发装置以及图像捕捉处理技术,Kinect能够完成对人体动作的识别、追踪和捕捉。Unity 3D是一款高集成度的游戏开发引擎,同时它也能提供多平台的游戏开发功能。开发人员可以很容易地编辑和渲染虚拟场景,可以使用Java、C#或者Boo语言来开发实现游戏逻辑功能。

为培养团队配合能力,一些化学实验项目可以采用多人配合实验的方式进行,为此在虚拟实验系统中添加多人在线功能显得尤为重要。本文使用远程过程调用控制(RPC)的方法来实现多人操作的统一控制,当多个用户加入局域网以后,

到稿日期:2015-08-24 返修日期:2016-01-20

孙博文(1963—),男,副教授,主要研究方向为机器视觉和计算机图形学,E-mail:sunbw01@163.com;**张佳梁**(1990—),男,硕士,主要研究方向为机器学习与体感交互;**蔡亚飞**(1990—),女,硕士,主要研究方向为机器学习与体感交互;**郭文兰**(1962—),女,硕士,教授,主要研究方向为计算机辅助设计与图形学。

任一用户的操作都会通过网络实时显示在所有终端设备上。

目前已有很多基于 Kinect 体感技术开发的应用，比如 Haggag H 等人开发的基于 Kinect 的人体工程学评估系统[4]可以帮助完成人工学分析；Chen Y Y 等人研究的基于 Kinect 的虚拟装配技术[5]可以允许工人在真实操作之前进行工作情景模拟。但是他们都没有注意到 Kinect 技术在虚拟实验系统上的应用。而 Liao Hongjian 和 Qu Zhe 等人在基于 Kinect 的减速器虚拟组装系统[6]和基于 Kinect 的虚拟电学训练实验系统[7]中，将 Kinect 很好地应用到了教学实践中，但却没有解决多人异地同时进行虚拟实验操作的问题。因此，本文研究的基于 Kinect 技术的多人在线虚拟实验系统具有一定的创新性和推广性。

2 虚拟实验系统框架

2.1 进行虚拟化学实验的目的

众所周知，化学实验总是充满了危险，如果实验人员尤其是年龄较小的学生不了解化学反应的负面作用，或者没有进行有效的保护，可能会在真实实验中受到伤害。虚拟化学实验系统可以允许实验者进行多次练习以熟悉实验步骤，并且可以帮助实验者避免人为失误最终完成实验。另一方面，有些实验药品比较昂贵，而虚拟实验可以帮助降低实验成本。此外，操作是化学实验中很关键的问题，如果操作有误会直接影响到实验现象和结果。真实实验前进行多次虚拟实验练习使实验者熟悉操作步骤，可保证实验结果的相对准确性。

2.2 实验场景界面设计

虚拟实验系统大致分为 4 个场景：菜单选择场景、实验演示场景、实验操作场景和实验评分场景。在以上 4 个场景中，实验者可以摆脱鼠标键盘而通过身体姿势和手部姿势来控制操作。这 4 个场景的关系如图 1 所示。

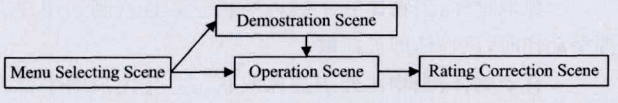

图 1 虚拟实验的 4 个场景

系统欢迎界面和菜单选择场景如图 2 所示。

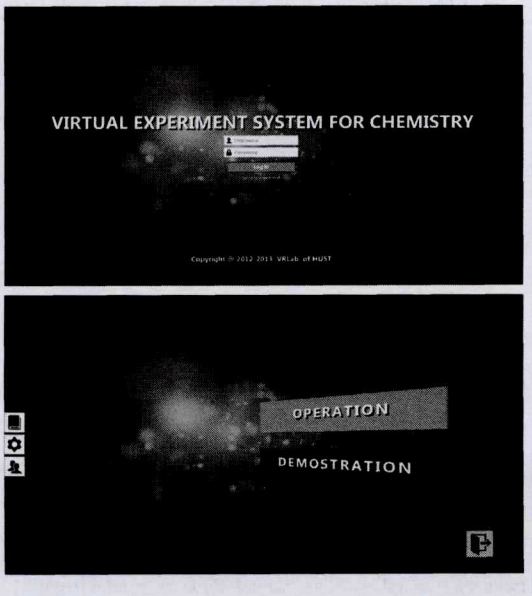

图 2 欢迎界面和菜单选择场景

实验操作场景如图 3 所示。

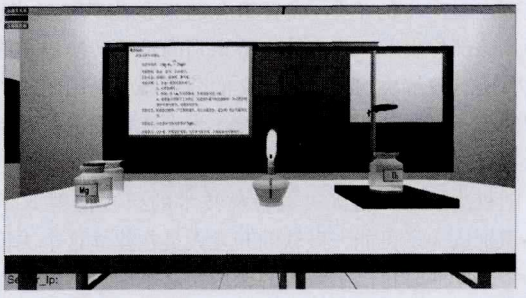

图 3 实验操作场景

2.3 开发环境与实验介绍

Unity 3D 游戏引擎在这里被用作三维空间的管理和显示平台。在开发过程中，使用微软的 Visual Studio 2012 编译器和 Kinect SDK 2.0 开发包在 Windows 7 环境下定义扩展了 Kinect 的姿势识别。在这个虚拟环境中，场景和器材模型由 3D Max 软件制作并导入 Unity 3D 引擎中。

以“镁条在氧气中燃烧”这个实验为例，真实实验步骤如下：

(1)准备一瓶刚收集的氧气；

(2)点燃酒精灯；

(3)折取一段 7cm 左右的镁条，用坩埚钳夹住一端；

(4)将镁条在酒精灯上点燃后，迅速移开集气瓶的玻璃片，将点燃的镁条伸入集气瓶中，观察实验现象。

3 实验中体感交互的实现

通过分析真实实验可知，虚拟实验中需要用到的人体姿势有以下几种：选择、抓取、旋转、移动和释放，以及在自由漫游模式下身体的倾斜移动控制虚拟角色行走。

Kinect 的人体姿势识别技术分为以下几部分：通过动作捕捉获得大量的不同姿态的人体深度图像，将这些深度图像中的人体各个部位标记出来；建立随机决策树，使用标记过的深度图像对决策树进行训练，让 Kinect 学习计算出人体各个部位的特征值；使用算法对人体各部位中的关节点的空间位置进行定位。经过以上几步，Kinect 就可以做到在实时采集到的环境深度图像中分离出人体图像，在人体图像上划分出各个部位，并在各个部位上确定关节点位置。

3.1 人体部位标签

Kinect 姿势识别技术中最关键的部分是人体部位表示法。如图 4 中灰度块所示，微软的开发人员定义了一系列的人体各部位标签，并用这些标签密集地覆盖整个人体表面。

图 4 与深度图相对应的人体部位图

在这些人体部位中，其中一部分能够被直接用来对特定

的兴趣关节点进行局部化，而其他的部位则用来弥补预测其他关节点时产生的不足，或者组合起来预测其他关节点[8]。这种人体部位表示法将人体姿势识别的难题转换成了能够被分类算法有效解决的问题。下文的分类器就是用图 4 所示的成对的带有人体部位标签的图像作为训练数据进行训练的。

3.2 随机决策森林

随机决策树和随机决策森林技术的引入使得 Kinect 的姿势识别成为了可能。微软的开发人员在建立这个决策森林的训练数据库时付出了巨大的代价。这个庞大的数据库是由来自诸如驾驶、跳舞、踢球、跑步和操作菜单等几百个人体动作序列的大约 50 多万帧图像组成[8]。对于数据库中每一帧深度图上给定的像素点，它的深度比较特征值由式(1)计算得出：

$$f_\theta(I,x)=d_I\left(x+\frac{u}{d_I(x)}\right)-d_I\left(x+\frac{v}{d_I(x)}\right) \tag{1}$$

其中，$d_I(x)$是图像 I 中像素点 x 的深度值，偏移量 u 和 v 由参数 $\theta=(u,v)$给出。偏移量的标准化$\frac{1}{d_I(x)}$保证了深度不变性。通过确定的世界空间偏移量，就可以判断出人体表面上某一点与 Kinect 传感器之间的距离是近还是远。

当某个像素点的特征值被计算出来之后，随机决策森林将会确定这个像素点属于人体的哪一个部分。随机决策森林是一系列决策树 T 的集合，每棵决策树都由子树和叶节点组成。子树中含有特征值 f_θ 和阈值 τ。为了对图像 I 中的像素点 x 进行分类，分类器从根节点开始在每一个分支处都反复对式(1)进行计算，将计算结果与阈值进行比较，从而决定是剪掉左子树还是右子树。在决策树 T 的叶节点处，得出了图像 I 中像素点 x 属于人体部位 c 的概率分布 $P_t(c|I,x)$。最终的分类结果由决策森林中的所有树的平均分布概率得出：

$$P(c|I,x)=\frac{1}{T}\sum_{t=1}^{T}P_t(c|I,x) \tag{2}$$

微软的技术人员在每一帧图像中的 2000 个样本像素点中随机选择一个子集，用这个子集来确保人体各部位的概率分布大致平均。使用下面的算法对每棵树进行训练[9]。

1)随机选出一套剪枝候选参数 $\Phi=(\theta,\tau)$(f_θ 为特征值，τ 是阈值)。

2)通过每个 Φ 来决定样本 $Q=\{(I,x)\}$属于左子集还是右子集：

$$Q_l(\Phi)=\{(I,x)\mid f_\theta(I,x)<\tau\} \tag{3}$$

$$Q_r(\Phi)=Q\backslash Q_l(\Phi) \tag{4}$$

3)根据式(5)和式(6)得出的最大增益来计算 Φ：

$$\Phi^*=\operatorname{argmax} G(\Phi) \tag{5}$$

$$G(\Phi)=H(Q)-\sum_{s\in\{l,r\}}\frac{Q_s(\Phi)}{|Q|}H(Q_s(\Phi)) \tag{6}$$

其中，香农熵 $H(Q)$是由身体部位标签 $l_I(x)$的标准化直方图计算得到的。

4)如果最大增益 $G(\Phi^*)$是足够的，并且决策树的深度低于最大值，那么就递归左右子集 $Q_l(\Phi^*)$和 $Q_r(\Phi^*)$。

3.3 关节定位方案

基于上述人体部位识别技术，可以推断得出各像素点的信息。若要生成可靠的 3D 关节点位置方案，必须将像素合并起来分析它们的信息。一个简单的方法是使用已知的矫正深度对每个部位进行概率累加，从而得到该部位的 3D 中心。然而，距离中心较远的像素严重降低了这一估算的准确性。因此，微软的开发人员采用一种基于具有高斯核的均值漂移的查找方法来代替上述过程[10]。

他们将每个身体部位的密度估算器定义为：

$$f_c(\hat{x})\propto\sum_{i=1}^{N}\omega_{ic}\exp\left(-\left\|\frac{\hat{x}-\hat{x}_i}{b_c}\right\|^2\right) \tag{7}$$

其中，$\hat{x}$是 3D 世界空间中的坐标，N 是图像像素的个数，ω_{ic}是像素的权重，$\hat{x}_i$是图像像素 x_i 根据深度 $d_I(x_i)$向世界空间的二次投影，b_c 是通过学习得到的每个部分的带宽。像素点权重 ω_{ic}可以认为是在像素点上推断得出的人体部分概率和像素点的世界表面区域：

$$\omega_{ic}=P(c|I,x_i)\cdot d_I(x_i)^2 \tag{8}$$

根据式(7)计算得出的人体各部位的密度，Kinect 能够在它捕捉的每一帧深度图像中确定每个关节点的最可能的位置。

3.4 人体动作定义

以上的所有工作成就了 Kinect 姿势识别技术。而普通开发人员借助微软发布的 Kinect 软件开发程序包(SDK)，就可以在 Windows 7 或者其他操作系统上开发可以支持姿势或语音识别的应用程序[11]。借助该 SDK 提供的人体 20 个(Kinect 2.0 可以提供 25 个)关节点空间位置坐标，我们定义了几种肢体动作来驱动虚拟实验系统中虚拟人物角色的移动；并且 SDK 也给出了对用户握拳情况的检测结果。本文定义了以下几种动作姿势。

- 选择物体：当虚拟场景中的手部模型与其他物体的碰撞体发生接触时，或者从摄像机到手部模型的射线穿过物体的碰撞体并停留 2 秒以上时，该物体即被选择。
- 抓取物体：当物体处于被选择状态并且检测到用户有握拳动作时，该物体即被抓取。
- 移动物体：当物体处于被抓取状态并且用户的手臂在向某个方向挥动时，物体将跟随手部模型一同移动。
- 释放物体：当用户的手张开时，物体将会被释放。
- 身体倾斜和移动：头部关节与髋部中心关节的位置差决定了虚拟角色向哪个方向行走。

当 Kinect 采集到用户手部位置后，系统程序会将手部位置进行一定的矫正以符合虚拟场景中的世界坐标系，然后绑定到用以表示手部的模型上。而当多个用户建立局域网时，系统需要从网络读取其他在线用户的手部位置及相关操作，并将这些信息绑定到当前场景中代表其他用户手部的模型上。本文使用 Unity 3D 的 NetworkView 组件和 RPC 功能来实现局域网内网络数据的传输。

3.5 Kinect SDK 与 Unity 3D 引擎

Kinect SDK 支持 WPF/Windows Form/Console Application 程序，但这三者属于二维平面图形，不能满足虚拟实验对真实性和沉浸感的要求，所以使用游戏开发者熟知的 Unity 3D 引擎进行虚拟实验场景的开发。Kinect SDK 对开发者开放的接口调用方式有两种：一种是基于 C++的头文件和静态链接库的方式，另一种是程序集的方式。而 Unity 3D 引擎中内置的 MonoDevelop 开发环境无法直接调用微软的托管

SDK，但它支持调用非托管的动态链接库的接口方式，以实现对数据的调用处理。在项目中导入 Dll 文件，用户通过自定义数据结构和算法，能够实现对 Kinect 中获得的数据进行重新组织，从而可在 Unity 3D 中使用。

在使用 Unity 3D 引擎开发 Kinect 相关应用时，应单独创建一个 Kinect 控制脚本，其功能包括：启动 Kinect 设备及设备初始化（"NuiInitialize"）、读入彩色图像流（"CopyKinectColorStream"）、读入深度图像流（"CopyKinectDepthStream"）和读入骨骼信息流（"CopyKinectSkeletonStream"）、获取用户握拳信息（"CopyKinectHandEventReslut"）、关闭设备（"NuiShutdown"）。

因为在 Unity 3D 内部不直接支持微软的 SDK，所以借助调用非托管函数的 DllImportAttribute 属性对 Kinect10. dll 进行调用，举例如下：

[DllImportAttribute (@"Kinect10. dll", EntryPoint = "NuiInitialize")]

public static extern int NuiInitialize (NuiInitializeFlags dwFlags);

经 Kinect 传感器采集到的信息数据类型主要为浮点型（float）或字节型（byte），在 Unity 3D 中可以编写脚本对 Kinect 采集的数据加以处理，然后实现用户的体感控制。比如，将 float 类型的骨骼关节点三维坐标转存入三维向量 Vector 类型的变量中，即可捕捉人体所有关节位置；将 byte 类型的用户握拳数据转存入 bool 类型的变量中，即可实现用户抓取操作的检测。

体感虚拟实验的人机交互模块就是基于上述用户动作捕捉和抓取检测实现的，其余操作逻辑则是根据真实实验步骤设计，通过 C# 脚本语言编写实现。

4 多人在线网络通信

4.1 网络协议

Unity 3D 的网络协议属于高层网络协议，它与引擎的游戏对象结合在一起，而非独立存在。它能为游戏提供网络功能支持，而且是可以跨平台的，可运行于 PC，IOS，Android，并且支持不同平台的客户端通信。Unity 3D 的网络协议由网络游戏对象管理、状态同步和 RPC 组成。

网络游戏对象管理：在 Unity 3D 网络模式下使用 Network. Instantiate() 函数创建游戏对象实例。该函数在本地创建一个游戏对象实例的同时，也在所有互联的终端上同时创建具有相同 NetworkView ID 的同样的游戏对象实例，同时这些游戏对象之间的通信关系也被建立起来，NetworkView ID 就是对象之间通信所用的身份标识，如图 5 所示。每个被创建出来的游戏对象都使用 NetworkView. isMine 来标识自己在当前场景中的身份。如果 isMine 为 true，则表示此游戏对象是本机创建的；false 则表示是其他主机创建并同步到本机的。游戏对象的 isMine 身份与网络底层的 server 和 client 身份没有任何关系，在 server 和 client 上都可以允许任何身份的游戏对象存在，这主要取决于游戏对象是谁创建的。最后调用 Network. Destroy() 对所有主机上的该对象进行销毁。

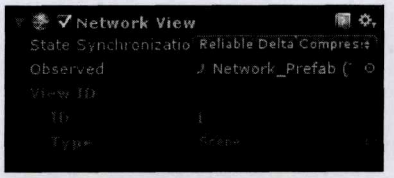

图 5 NetworkView ID

状态同步：当网络中某个对象的属性发生变化，所有具有相同 NetworkView ID 的对象的状态都会随之更新，更新方向是从当前终端上 isMine 为 true 的对象到其他终端上 isMine 为 false 的对象。

远程过程调用协议（RPC）：RPC 是一种从远程计算机程序请求服务的协议，它不需要开发者了解底层的网络通信技术。RPC 假设某种协议的存在，比如 TCP 或者 UDP，在这种假设下在通信程序之间传递数据。RPC 采用的是客户端/服务器模式。程序的请求者是客户端，而程序的提供者是服务器端。

在计算机网络中，数据的存放和处理地点可能不在同一主机上。通常情况下，是将数据从存放主机发送到处理主机上进行处理，处理机返回距离结果。但若数据量较大，这种方式既消耗网络资源又保证不了可靠性。而 RPC 执行过程是：调用过程暂停执行，将参数通过网络送至被调用者以执行此过程调用，执行结束后结果被返回给调用者，调用者恢复执行。

具体调用过程是：当一个局域网建立起来之后，RPC 协议将会根据程序代码在整个局域网中查找调用被[RPC]标记的函数。这个标记的存在是为了引导如何找到远程过程，根据 ONC RPC 标准：设备必须为每个在其上执行的远程程序分配一个唯一的 32bit 整数，调用者通过该整数来标识该远程程序。由于一台机器上可能同时运行多个远程程序，因此每个远程程序将使用不同的传输层端口。而传输层使用的端口是 16bit 整数，为此不能将一个远程调用唯一地映射到一个传输层端口上。即使能够做到这样的映射，也将浪费大量的传输层端口资源。所以远程程序的端口是由系统根据传输层端口的使用情况动态分配的。远程程序提供服务之前，会先将自己的信息在端口映射器注册。应用程序在调用远程程序之前，会先请求端口映射器，查询所要调用的远程程序使用的端口号。端口映射器维护一张表，将本机运行的远程程序机器端口号保存在该表中。应用程序根据映射器查表得出的端口信息，向具体的远程程序发起调用请求。整个过程如图 6 所示。

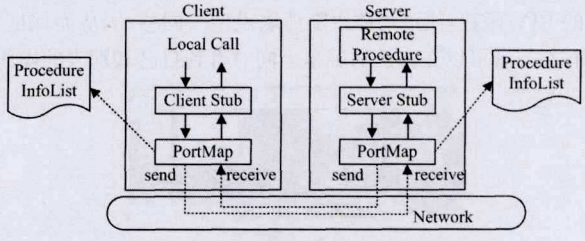

图 6 RPC 通信示意图

远程过程调用原语是在报文传递原语的基础上产生的，它使用 XDR 语言来定义报文。使用 XDR 定义的 RPC 报文类型如下：

enum msg_type

```
{ /* RPC 报文类型 */
  CALL=0;
  REPLY=1;
};
```

RPC 报文的类型只有两种：RPC CALL 和 RPC REPLY，分别用于远程调用和远程调用结果的返回。RPC CALL 是把可靠的阻塞原语 SEND 和 RECEICE 结合起来形成 SEND-GET，用于客户端进程向服务器进程发送一个请求，之后等待服务器进程的回答。而 GET-REQUEST 用于服务器得到一个报文，报文告诉服务器要做的工作。当服务器完成该工作时，RPC REPLY 用原语 SEND-REPLY 发回一个回答报文。远程过程调用可以看作是由这些原语结合起来完成的过程，具有对用户更方便的句法。与报文传递方式相比，RPC 具有许多优点：语义清楚、易用；简化了通信，效率高。

RPC 不是 Unity 3D 独有的功能，但是 Unity 3D 引擎中的 Network View 组件可以实现 RPC 功能。使用 RPC 可以在其他主机的相同游戏对象上执行一个远程函数调用，一般用来通过网络通知一次性的游戏事件。RPC 的传输方向非常自由，可以在一个主机上向包括自己在内的任何其他主机发送 RPC 调用。在编写实现建立局域网(LAN)功能的脚本文件时，需要在脚本文件中引用命名空间：System. Net 和 System. Net. NetworkInformation。了解以上原理之后，就可以搭建系统的网络通信模块。

4.2 虚拟实验系统的网络通信模块

单人操作：当用户进行单人操作时，在虚拟场景中会有两个绿色的小球实时随着用户的手部移动。用户可以控制小球移动到场景中的虚拟仪器上进行抓取等操作。

多人操作：当用户选择多人操作时，他会在屏幕的左下角看到其电脑的 IP 地址，如图 7 所示。如果有两位用户或者更多用户想同时进行一项虚拟实验，那么他们中的一个人需要挥舞他的手臂使屏幕上的光标位于“创建服务器”的按钮上，然后握拳，而其他人需要在他们各自的屏幕上输入服务器用户的 IP 地址，最后在“连接服务器”的按钮上握拳。这样就建立了一个可以允许多人同时操作的局域网。

用户加入局域网时，会在全网内的虚拟场景创建具有特定 NetworkView ID 的浅色小球(自己设备上的小球为深色，浅色小球为其他用户在当前场景中创建的)。用户移动手臂时，根据 Kinect SDK 提供的骨骼数据，当前场景中的深色小球会相应移动，同时在局域网中其他设备上具有与这个小球相同 NetworkView ID 的浅色小球也会相应移动。当用户抓取某个物体时，被抓取的物体会调用其他用户场景中该物体上的 RPC 函数，使该物体跟随抓取模型一同移动，从而保证了所有参与者可以在自己的屏幕上同时看到自己和别人的操作。

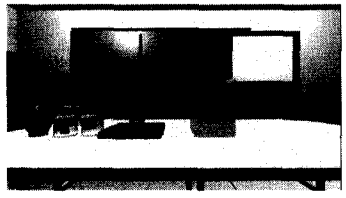

图 7 不同颜色的小球代表不同的用户

结束语 本文主要讨论了将姿势识别应用到虚拟实验系统中和如何实现多人同时在线进行虚拟实验的关键问题。根据系统需要较为详细地分析了 Kinect 传感器的技术原理，包括使用经过人工标定各部位标签的深度图像对随机决策树进行训练，当 Kinect 识别出人体的各部位之后，根据算法推算出各个关节点的位置。将 Kinect 体感技术作为人机交互的一种方式是本系统的核心部分，姿势识别和人体动作定义降低了系统对传统的鼠标键盘的依赖。其次，介绍了为实现多人异地在线进行虚拟实验所使用的 Unity 3D 网络通信技术。在这个基础上，使用有限状态机来进行实验逻辑任务的智能化管理。实践证明，以上方法使该系统的人机交互比以往更加自然和高效，可以满足虚拟实验过程中的交互需求。局域网技术的加入，一定程度上填补了体感虚拟实验系统在多人异地操作方面的空白，对于完成虚拟实验任务和提高用户体验起到了至关重要的作用。

参 考 文 献

[1] Lu Miao. The application of the virtual experiment in teaching researching[J]. Chinese Journal of ICT in Education, 2014, 12: 79-80(in Chinese)
卢苗. 虚拟实验在教学中的应用限度研究[J]. 中国教育信息化, 2014, 12: 79-80

[2] Brough J E, Schwartz M, Gupta S K. Towards the Development of a Virtual Environment-based Training System for Mechanical Assembly Operations[J]. Virtual Reality, 2007, 11(4): 189-206

[3] Garcia J, Valencia E S, Zalevsky Z, et al. Range Mapping using Speckle ecorrelation: US Pat 7,433,024 B2[P]. 2008-10-07

[4] Haggag H, Hossny M, Nahavandi S, et al. Real Time Ergonomic Assessment for Assembly Operations Using Kinect[C]// International Conference on Computer Modeling and Simulation, 2013. Hong Kong, 2013: 495-500

[5] Chen Y Y. Gesture Recognition based on Kinect and Application in the Virtual Assembly Technology[J]. Electronic Design Engineering, 2013, 21(10): 4-7

[6] Liao Hong-jian, Long Xiao-li. Study on Virtual Assembly System Based on Kinect Somatosensory Interaction[C]// ISCC. 2013: 55-60

[7] Liao Hong-jian, Qu Zhe. Virtual Experiment System for Electrician Training based on Kinect and Unity 3D[C]// MEC. 2013: 2659-2662

[8] Shotton J, Fitzgibbon A. Real-Time Human Pose Recognition in Parts from Single Depth Images[DB/OL]. http://www.microsoft.com/en-us/research/people

[9] Lepetit V, Lagger P, Fua P. Randomized Trees for Real-time Key point Recognition[C]// CVPR. 2005: 775-781

[10] Comaniciu D, Meer P. Mean shift: A robust approach toward feature space analysis[C]// IEEE Trans. PAMI. 2002: 603-619

[11] 余涛. Kinect 应用开发实战，用最自然的方式与机器对话[M]. 北京：机械工业出版社，2012：141-145，296-300