基于动态纹理和投影贴图技术的水刻蚀效果仿真

付浩生,李文庆,张 涛,陈 戈
(中国海洋大学 信息科学与工程学院,山东 青岛 266100)

摘 要:水刻蚀效果作为一种常见的自然现象,而通常基于物理原理进行实时计算的水刻蚀效果模拟方法存在计算量大及逼真度不够等问题,在虚拟现实及游戏场景中很少使用。文中提出了一种基于动态纹理和投影贴图技术的水刻蚀效果模拟方法。用投影贴图技术把预先生成的经过精心处理的水刻蚀纹理投影在场景中,再以动态纹理技术定时替换光影贴图。省去了对反射和折射光线的实时计算,有效地提高了场景的真实度与实时性。同时,还可以灵活设置投影区域及光影效果,生成的光影效果可投射在运动的物体表面上。实际项目应用表明,该方法在大规模场景模拟效果令人满意。

关键词:水刻蚀;动态纹理;投影贴图;仿真
中图分类号:TP391.41 **文献标识码**:A **文章编号**:1673-629X(2010)09-0014-05

Caustic Effect Simulation Based on Dynamic Texture and Projection Texture Technology

FU Hao-sheng, LI Wen-qing, ZHANG Tao, CHEN Ge
(College of Information Science & Engineering, Ocean University of China, Qingdao 266100, China)

Abstract: Caustic effect is a common natural phenomenon. For caustic simulation based on real-time computation have many problems, such as time-consuming calculation, lack of real-time and verisimilitude, etc. This method is rarely used in virtual reality and game scenarios. A new simulation technique based on dynamic texture and projection texture was proposed. With this method, lighting effects is projected to the objects in the scene, the calculation quantity and complexity are reduced. At the same time, you can also change the lighting effect by using different textures. The actual project application show that the method greatly improved the verisimilitude of caustic simulation and it can be used in large scale scene simulation.

Key words: caustic; dynamic texture; projection texture; simulation

0 引 言

在计算机图形学领域,对水刻蚀效果的仿真一直是一个热门话题。水刻蚀是一种很常见的自然现象。夏日里,在山间的小溪,常常会看到清澈的水底一圈圈随着水面舞动而又变幻无穷的美丽花纹,这就是水刻蚀现象。水刻蚀效果可以大大增强虚拟现实或游戏场景的沉浸感,给用户带来极其美妙的视觉体验。它产生的原理并不复杂:太阳光照射在水面上经水面折射与反射,一部分光在水底会聚起来,形成耀斑,当水面波动时,耀斑随之变幻,把一个凸透镜放在太阳下也会形成这种耀斑。从物理角度讲,光线的反射遵从反射定律,即:

$$Out = In - 2 * N * In * N$$

在上式中 In 为入光线,Out 为出射光线,N 为反射光线的单位法向量。光线的折射遵从 Snell 折射定律,计算如下:

$$\mu_1 * \sin\Theta_1 = \mu_2 * \sin\Theta_2$$

μ_1 空气对光的折射率,μ_2 为水对光的折射率,Θ_1 为入射角,Θ_2 为出射角。In 为入射光线,N 为入射光线的单位法向量,Out 为出射光线,这三个向量是共面的。反射与折射的示意图如图1、图2所示。

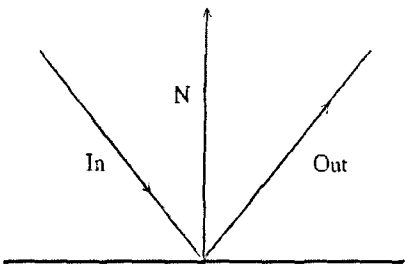

图1 反射的计算

收稿日期:2009-12-19;修回日期:2010-04-09
基金项目:国家自然科学基金(40971207)
作者简介:付浩生(1984-),男,硕士研究生,研究方向为计算机图形学,虚拟现实技术;陈 戈,博士生导师,研究方向为海洋遥感与海洋地理信息系统,虚拟现实技术。

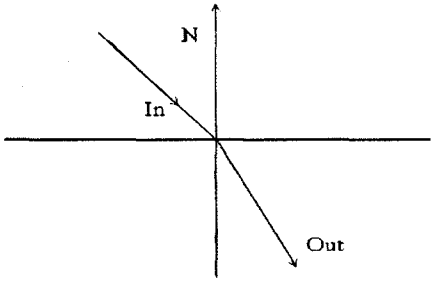

图 2　折射的计算

但是在计算机中按其产生的物理过程对水刻蚀效果进行高真实度模拟却并非易事,因为水刻蚀效果是由水面波浪反射太阳光产生的,首先要建立一个比较真实的波面模型。徐迎庆等人[1]运用一个能够描述水流特征的物理模型,然后据此模型计算不同时间各个点的位置进行模拟。鄢来斌等[2]运用一种虚拟海战场景中快速度高效海浪建模算法,建立了一个依赖于视点的网格成生方法对波面进行仿真。杜岩等[3]从海浪形态的 N－S 模型出发,加入了对海浪形态的随机性和复杂性对波面的影响,还引入正函数,对模型构造出的海浪形态做进一步修正,从而更真实地模拟出海浪随面波动的情形。陈和平等[4]在分析粒子系统和水波原理基础上,还考虑了光折射,波扩散与衰减等因素,最终给出了水波振幅的线性近似算法与水波特效模拟快速实现方法。Peachey[5]用高度场来表示海面,用正余弦函数叠加拟波浪,同时还对波面振进行分析。Foster 等[6]使用有限差分法的似求解流体 Navier－Stokes 方程,根据流体力学模型真实地模拟了水面效果。Lengyel[7]采用二维离散模型模拟水波,较为复杂。Promoze 等[8]将海洋相关理论与物理方法相结合也得到了真实感效果。但模拟水刻蚀效果还得进一步根据上述波面模型分解各个小三角面,再根据三角面来计算最终折射与反射光线。光线经过这些小三角面发生了无数次折射与反射,并且每发生一次折射与反射,光线强度都会随之衰减,各种光线交织在一起相互影响,要对大量光线分别运用上述定律,运算量将会十分巨大。

2001 年,Jensen 和 Golias 根据水刻蚀产生的物理过程对其进行仿真。从太阳发出的光线出发,射向水面三角形网格,再依据 Snell 定律计算折射光线,可参考文献[9],对于含三角面比较少的水面模型,这种方法能够得到不错的效果,但要对大规模水面场景进行实时仿真,即使采用最新硬件进行渲染,也会对系统带来沉重的负担。这种方法还有一个缺陷,现实中的水面曲率是平缓的,而虚拟现实或游戏场景中,水面都是用三角网格表示的,每一个三角形是平面的,这样照在每一个三角形上的光线其反射与折射光线方向相同,这与事实不符,从而导致对水刻蚀模拟的真实度不够。因此在现实中这种方法很少使用。Stam 等[10]用波理论同时模拟水波,用贴图模拟刻蚀刻蚀效果,但相关变换较为复杂,要在真实度足够好的情况下达到可以接受的帧速率,必须另寻其它途径。文中在总结前人成果的基础上提出了一种基于动态纹理和投影贴图技术实现的水刻蚀效果仿真,将水刻蚀贴图的纹理投影到场景中的物体上,省去了对大量光线折射与反射的实时计算。由于水刻蚀贴图来自自然界真实场景,故能非常逼真地模拟水刻蚀效果,经过性能优化,使得该方法能够在当前绝大多数计算机上流畅地运行。

1　水刻蚀仿真工作原理

在讲述其原理之前,先回顾一下投影仪的工作原理:一束强光照射在胶片上,在屏幕上将会出现胶片上的图像,当不断更换胶片时,屏幕上投下的图像也随之变动,从而可以看到一幅幅连贯的动作。投影仪所投下的光影,不仅仅出现在屏幕上,如果有人在屏幕前走动,光影会投射在人身上。试想:如果投影仪播放的是一幅幅预先录制好的水刻蚀画面,那么,在屏幕上,人身上将会投下这种交相辉映的光斑。这里要采用的方法与此类似,所不同的是,投影仪工作在现实世界中,而我们所要进行的模拟工作在虚拟世界里,下面将要在 OpenGL 的三维世界里实现这个投影过程。

2　纹理贴图的准备

纹理映射技术是计算机图形系统中广泛采用的一项技术,利用纹理映射,可以简化建模过程,生成比较真实的场景。对于同一模型的同一三角面片,可是贴一张贴图,还可以贴多张贴图,这里贴两张贴图(即二重纹理)。一张用来表现物体本身的颜色属性,另一张用来表现水刻蚀光影效果。单张水刻蚀纹理取自自然场景中水刻蚀照片经 Photo Shop 处理去除噪点,如图 3 所示。

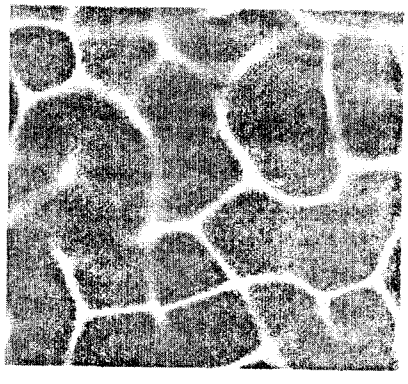

图 3　水刻蚀纹理

3 投影纹理的实现

纹理映射允许我们将一幅图片映射到一个多边形的表面，最常见的是二维纹理，可以通过给一个赛车模型的每个点的指定纹理坐标，将一个图片贴在上面，当赛车模型运动时，贴在上面的图片随之运动。但纹理也可以是一维的、三维的，纹理还可以是自动生成的。所有这些都涉及到一个最重要的问题，即如何指定纹理坐标。如果把投影仪及其周围的物体放置在一个三维坐标系统中，从物体沿着到投影仪的投射光线做逆向投影，可以看出物体上的点与投影仪胶片上的纹理坐标是一一对应的，据此，可以把物体位置坐标变换为纹理坐标，现在利用 OpenGL 纹理自动生成原理来做这个变换。

3.1 纹理坐标自动生成

在 OpenGL 中，指定纹理坐标有两种方式：一种是二维的并且是非投影的，由 glTexCoord*( )为每个点显示地指定两个纹理坐标值：s 和 t 纹理坐标。由这种方式指定的纹理坐标值在程序运行过程中是固定不变的。另一种是被称作齐次纹理坐标，由四个分量组成：s,t,r,q，允许坐标值在纹理空间用 4X4 矩阵进行投影、旋转、平移和缩放变换，一般由 glTexGen*( )让 OpenGL 自动生成纹理坐标。这里先让 OpenGL 自动生成纹理坐标，再对坐标值进行变换，代码如下(详细设置请参见文献[11])：

```
GLint loadCausticTexture(const std::string fileName)
{
static GLfloat planeS[4]= {1.0f, 0.0f, 0.0f, 0.0f};
static GLfloat planeT[4] = {0.0f, 1.0f, 0.0f, 0.0f};
static GLfloat planeR[4]= {0.0f, 0.0f, 1.0f, 0.0f};
static GLfloat planeQ[4]= {0.0f, 0.0f, 0.0f, 1.0f};
...
glTexGeni(GL_S, GL_TEXTURE_GEN_MODE, GL_EYE_
LINEAR);
glTexGenfv(GL_S,GL_EYE_PLANE,planeS);
glTexGeni(GL_T,GL_TEXTURE_GEN_MODE, GL_EYE_
LINEAR);
glTexGenfv(GL_T, GL_EYE_PLANE,planeT);
glTexGeni(GL_R, GL_TEXTURE_GEN_MODE,GL_EYE_
LINEAR);
glTexGenfv(GL_R, GL_EYE_PLANE, planeR);
glTexGeni(GL_Q, GL_TEXTURE_GEN_MODE,GL_EYE_
LINEAR);
glTexGenfv(GL_Q,GL_EYE_PLANE,planeQ);
...
}
```

在创建纹理阶段，GL_EYE_LINEAR 指定纹理坐标在眼坐标空间按线性方式生成，这样生成的纹理不会附着在物体上。自动纹理参考平面为以平面方程标准形式 $Ax + By + Cz + D = 0$ 给出，plane$S$，plane$T$，plane$R$，plane$Q$ 分别指定四个参考平面参数。

做出如上设置后 OpenGL 会把物体在眼坐标系位置距指定参考平面距离赋给纹理坐标，运算过程：

$$S = x * SA + y * SB + z * SC + w * SD = x$$

$$T = x * TA + y * TB + z * TC + w * TD = y$$

$$R = x * RA + y * RB + z * RC + w * RD = z$$

$$Q = x * QA + y * QB + z * QC + w * QD = w$$

至此，物体的纹理坐标(s,t,r,q)和物体在眼坐标空间中距指定参考平面的距离(实际上是物体在眼坐标系空间的坐标值)相等，但纹理坐标值在(0,1)之间，被变换成物体在眼坐标空间的坐标值，这还不是想要的结果。接下来计算纹理变在矩阵，将其变换到[0,1]之间。

3.2 动态贴图加载

为了能够表现出光影不断变幻的效果，采用动态纹理技术，即：预先加载多张纹理贴图，每渲染一帧图像，换一张贴图，这样不断重复下去，像放电影一样。相邻两张贴图及第一张和最后一张贴图的光影轮廓逐渐过渡，从而形成连续的光影效果。在实际编程中，为了让程序在不同配置的电脑上以相同频率更新光影，可以预先设置一个定时器，每秒置换若干张贴图。

3.3 求纹理变换矩阵

OpenGL提供了对纹理坐标进行坐标变换的功能，对于自动生成的纹理坐标在应用到物体之前需要乘以一个 4X4 矩阵。在默认情况下，纹理矩阵为单位矩阵，对纹理坐标不做任何变换，在这里需要显式指定纹理矩阵对纹理坐标进行变换，将物体在眼坐标空间中的位置坐标变换为纹理坐标。其过程与将物体从世界坐标变换系到屏幕所做的变换类似，变换流程：

(1) 模型视图变换：放置投影仪，决定水刻蚀纹理将要投影到那些物体上。投影仪在世界坐标系位置为 $\boldsymbol{P}$，向上向量 $\boldsymbol{U}$，纹理投射点 $\boldsymbol{D}$，以 $\boldsymbol{P}$ 为原点建立投影仪坐标系。如下所示：

$$\boldsymbol{Z}' = \text{normal}(\boldsymbol{D} - \boldsymbol{P})$$

$$\boldsymbol{X}' = \text{normal}(\text{cross}(\boldsymbol{U}, \boldsymbol{Z}'))$$

$$\boldsymbol{Y}' = \text{cross}(\boldsymbol{Z}', \boldsymbol{X}')$$

假设世界坐标系中任意一点 $Q(x,y,z)$，它在投影仪坐标系中的坐标为 $Q'(x',y',z')$。

Q 与 Q' 有如下关系：

$$x' = (Q - \boldsymbol{P}) * \boldsymbol{X}' = x * \boldsymbol{X}'.x + y * \boldsymbol{X}'.y + z * \boldsymbol{X}'.z - \boldsymbol{X}' * \boldsymbol{P}$$

$$y' = (Q - \boldsymbol{P}) * \boldsymbol{Y}' = x * \boldsymbol{Y}'.x + y * \boldsymbol{Y}'.y + z * \boldsymbol{Y}'.z - \boldsymbol{Y}' * \boldsymbol{P}$$

$z' = (Q - P) * Z' = x * Z'.x + y * Z'.y + z * Z'.z - Z' * P$

用向量与矩阵乘法形式表示如下:

$$(x', y', z', 1) = \begin{matrix} X'.x & X'.y & X'.z & -X' * P \\ Y'.x & Y'.y & Y'.z & -Y' * P \\ Z'.x & Z'.y & Z'.z & -Z' * P \\ 0 & 0 & 0 & 1 \end{matrix}$$

由此可得投影仪的模型视图矩阵为:

$$\begin{bmatrix} X'.x & X'.y & X'.z & -X' * P \\ Y'.x & Y'.y & Y'.z & -Y' * P \\ Z'.x & Z'.y & Z'.z & -Z' * P \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(2) 投影变换:投影变换用来将三维物体"压扁"为平面,这里采用投影变换标准变换矩阵(推导过程见文献[12]):

$$\begin{bmatrix} \cot(fov/2) * aspect & 0.0f & 0.0f & 0.0f \\ 0.0f & \cot(fov/2) & 0.0f & 0.0f \\ 0.0f & 0.0f & zf/(zf - zn) & -zf * zn(zf - zn) \\ 0.0f & 0.0f & 1.0f & 0.0f \end{bmatrix}$$

fov 为视角,aspect 为宽高比,zf 和 zn 分别为远近裁剪面。

(3) 视口变换:视口变换把投影变换结果进一步变换到目标平面上,变换矩阵如下:

$$\begin{bmatrix} w/2 & 0.0f & 0.0f & x0 + w/2 \\ 0.0f & -h/2 & 0.0f & y0 + h/2 \\ 0.0f & 0.0f & \max Z - \min Z & \min Z \\ 0.0f & 0.0f & 0.0f & 1.0f \end{bmatrix}$$

(4) 缩放变换:经过述变换后,纹理坐标值被限制在[-1,1]之间,正常纹理坐标在[0,1]之间。故还要进行进一步变换,将纹理坐标变换到[0,1]之间。缩放矩阵如下:

$$\begin{bmatrix} 0.5f & 0.0f & 0.0f & 0.0f \\ 0.0f & 0.5f & 0.0f & 0.0f \\ 0.0f & 0.0f & 1.0f & 0.0f \\ 0.0f & 0.0f & 0.0f & 1.0f \end{bmatrix}$$

经过上述变换,物体位置坐标从三维空间线性变换到[0,1]之间,被当做纹理坐标来处理,正如投影仪前的物体被反向投射到胶片上。变换流程如图 4 所示。

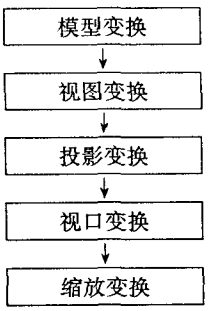

图 4　纹理矩阵变换流程

4　消除背投影

当一个纹理坐标(s,t,r,q) 中 q 分量为负值时,会发生背投影现象,背投影指纹理被投影在投影仪背后的表面上。这不符合实际情况,为消除背投影,可在渲染物体前使用剪裁平面删除投影仪后面的几何体。

5　实验结果

为了验证该技术在大场景中模拟效果及渲染效率,我们在实验室自主开发的测试平台上进行测试。导入海洋动漫场景数据(三维数据 20MB,纹理数据 90MB),打开水刻蚀效果及雾效。由图 5,6 中可以看到,在中低端配置的电脑上该方法能够非常逼真地模拟水刻蚀效果,生成的投影能够投射到运动着的鱼身上,平台能够以较高的帧率稳定地运行,完全能够满足应用需要。

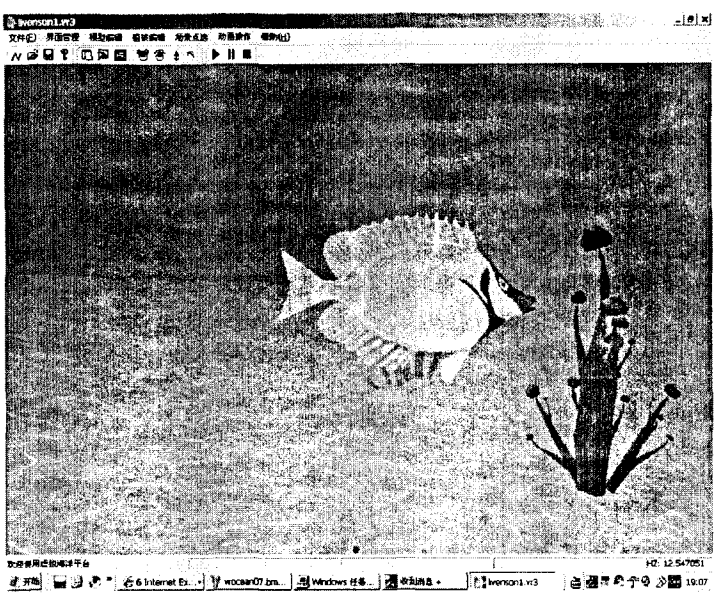

图 5　光影投射在海底及鱼身上

测试所用电脑配置如下:

CPU	Intel CPU 3.0G
显卡	Nvidia Elsa 6600 显卡
内存	三星 DDR2 2G
硬盘	日立 160G

6　结束语

虚拟现实技术在未来有着广阔的发展前景,实时性及真实性一直是虚拟现实技术追求的目标。对自然

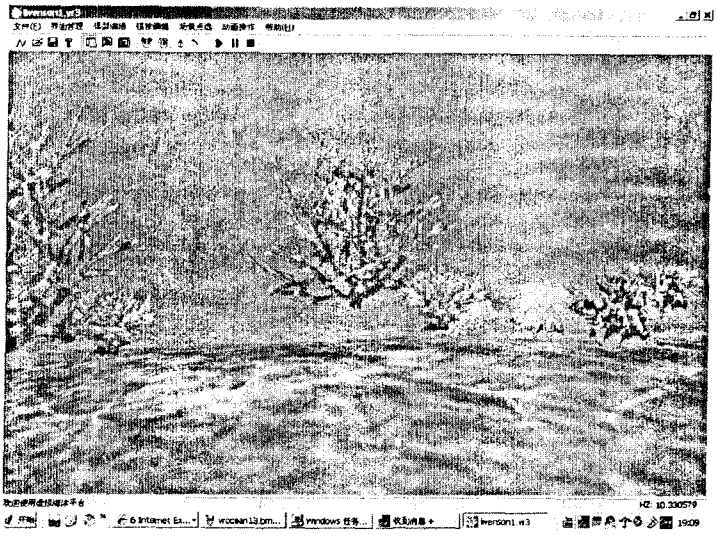

图 6 光影投射在植物上

场景中的光影效果可大大增加三维场景的真实感，但对自然场景中的光影效果的模拟通常涉及大量计算。特别是在对大规模场景中，受限于计算机硬件的发展，实时性和真实性通常无法同时得到满足，这时候，就需要另辟蹊径寻找合适的替代方案。文中以用动态纹理和投影贴图技术代替实时计算对水刻蚀效果进行模拟。该方法在国家自然基金项目《城域景观 VR－GIS 一体化仿真平台》中得到应用，极大地提高了三维场景的沉浸感，取得了令人满意的效果，其原理和方法也可用于对其它场景的模拟。

参考文献：

[1] 徐迎庆，苏 成，李 华，等. 基于物理模型的流水及波浪模拟[J]. 计算机学报，1996(增刊)：153－160.

[2] 鄢来斌，李思昆，张秀山. 虚拟海战场景中的海浪实时建模与绘制技术研究[J]. 计算机研究与发展，2001，38(5)：268－273.

[3] 杜 岩，张晓宇，李文秀. 虚拟现实场景中海浪形态的计算机模拟[J]. 哈尔滨工程大学学报，2001，22(3)：26－30.

[4] 陈和平，王 早. 水波特效模拟原理及期快速实现方法[J]. 计算机应用研究，2005，22(4)：252－255.

[5] Peachey D. Modeling waves and surf[J]. Computer Graphics，1986，20(4)：65－74.

[6] Foster N，Metaxas D. Realitic animation of liquids [J]. Graphical Models and Image Processing，1996，58(5)：471－483.

[7] Lengyel E. Mathematics for 3D game programming & computer graphics[M]. Hingham：Charles River Media，Inc，2002.

[8] Premoze S，Ashikhmin M. Rendering natural waters[J]. Computer Graphics Forum，2001，20(4)：189－200.

[9] Fernando R. GPU 精粹[M]. 姚 勇，王小琴，译. 北京：人民邮电出版社，2006.

[10] Stam，Jos. Random Caustics：Wave Theory and Natural Textures Revisited [C]//Technical Sketch. In Proceedings of SIGGRAPH. [s. l.]：[s. n.]，1996.

[11] OpenGL Architectrure Review Board，Shreiner D，M Woo，et al . OpenGL 编程指南[M]. 徐 波，译. 北京：机械工业出版社，2009.

[12] LaMothe A. 3D 游戏编程大师技巧[M]. 李祥瑞，陈 武，译. 北京：人民邮电出版社，2005.

(上接第 13 页)

参考文献：

[1] Kitano H，Asada M，Kuniyoshi Y，et al. RoboCup：The robot world cup initiative [C]//Proceedings of the International Conference on Autonomous Agents. [s. l.]：[s. n.]，1997：106－107.

[2] Kitano H，Asada M. RoboCup Humanoid Challenge：That's One Small Step for A Robot，One Giant Leap for Mankind [C]//Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS－98). Victoria，Canada：[s. n.]，1998：149－164.

[3] 罗 青，吕恬生，费燕琼. 足球机器人仿真系统的研究与开发[J]. 计算机仿真，1999，16(4)：27－30.

[4] 郭叶军. 机器人足球仿真比赛中多智能体系统的构建[D]. 杭州：浙江大学，2004.

[5] Alebaran Co. Ltd. Nao Acadamics Ed[EB/OL]. 2009. Available online http：//www. aldebaran－robotics. com.

[6] 侯清涛，厉广伟，李金屏. Robocup 中型组机器人足球技术探讨[J]. 济南大学学报：自然科学版，2008，22(3)：270－275.

[7] 林 昊，赵明国. Robocup 类人组技术挑战赛标志柱识别新探[J]. 计算机系统应用，2009(2)：167－170.

[8] Menegatti E，Kulvanit P，Von Stryk O. RoboCup Soccer Humanoid League Rules and Setup [EB/OL]. 2009. http：//www. tzi. de/humanoid/bin/view/Website/WebHome.

[9] 刘光宇，刘国栋，韩云生. 一种基于全景视觉系统的 Robocop 机器人定位方法[J]. 江南大学学报：自然科学版，2009，8(3)：279－282.

[10] Bao Dunqiao，Wang Hao，Fang Baofu，et al. HfutEngine3D Soccer Simulation Team Description 2008[C]//In：Proeeedings CD RoboCup 2008. Suzhou，China：Springer－Verlag，2008.

[11] 程显毅，王 军，张 俊. RoboCup 3D Server 分析[J]. 计算机工程与科学，2006，28(3)：119－122.

[12] Boedeeker J，Dorer K，Rollmann M，et al. SimSpark User's Manual[EB/OL]. 2009－06. http：//simspark. soureeforge. net/.