MFC 框架下 Vega 视景的视点控制技术①

刘元琳 [1,3], 祝 勇 [2], 张海洋 [1,3]

[1](长春理工大学 电子信息工程学院, 长春 130022)
[2](长春理工大学 计算机科学技术学院, 长春 130022)
[3](长春理工大学 空间光电技术研究所, 长春 130022)

摘 要: 大范围战场环境的视景仿真中, 仿真实体运动状态复杂多变. 为解决这种视景仿真的逼真性问题, 通过对观察者和目标间几何对应关系的分析, 给出了三种控制模式: 等距离模式、等速度模式、匀加速模式, 设计了虚拟视景中视点控制的基本算法. 在 MFC 框架下, 应用 Vega API 函数实现 C 语言对 Vega 视景仿真平台的调用, 实现了视点对目标的随动控制方法, 解决了视景仿真系统中视点的随动控制问题. 在视景仿真实验中, 达到了对视点方式的灵活控制, 具有较强的现实直观性, 同时增强了视景的真实感和交互性.

关键词: 视点控制方式; 目标随动; 视景仿真; 微软基础类库; Vega

Visual Control of Vega Observer Under the MFC Framework

LIU Yuan-Lin[1,3], ZHU Yong[2], ZHANG Hai-Yang[1,3]

[1](Electronic and Information Engineering, Changchun University of Science, Changchun 130022, China)
[2](Computer Science and Technology, Changchun University of Science, Changchun 130022, China)
[3](Institute of Space Optoelectronics Technology, Changchun University of Science, Changchun 130022, China)

Abstract: In a wide range of visual simulation battlefield environment, the motion state of simulation entities are complicated and changeable. In order to solve this visual simulation of fidelity problems, this paper analyzes the geometry correspondence relationship between the observer and the target, gives three control modes: equidistant mode, uniform speed mode, uniform acceleration mode, the basic algorithm of observer controlling in virtual scene is designed. Under the MFC framework, applying of Vega API function achieves the C language calling Vega visual simulation platform, achieving the observer servo control method for the target, solving the problem of observer automatic controlling in visual simulation system. In the visual simulation experiments, a flexible controlling to viewpoint way is achieved, with a strong practical intuitive property, while enhancing the realism and interactivity of viewpoints.

Key words: observer controlling approach; target automatic; scene simulation; MFC; Vega

随着科技的飞速发展, 视景仿真技术被广泛应用于航空航天、工程设计、军事训练以及娱乐、教育、医疗等各个领域, 它能够直观地展现事物运动变化规律, 具有不可比拟的信息集成优势.

在虚拟仿真系统中, 视景逼真性主要由三维实体模型、虚拟环境、视点方式以及计算机系统硬件性能等因素决定. 就视点方式而言, 视景仿真软件系统中必须设计出可方便切换并能灵活调整的视点方式, 以使观看者以最佳角度、最佳位置和最佳形式来观察仿真实验.

大范围战场环境的视景仿真, 由于场景范围大、仿真实体多而且运动状态复杂多变, 所以观看的视点方式就要求灵活机动且形式多样[1]. 视点的位置以及运动方式很大程度上决定观察者对目标运动姿态的判别和识别. 为此, 如何使观察者在最佳角度和最佳位置上快速、灵活的观察目标成为虚拟场景仿真实验的

① 收稿时间:2014-09-02;收到修改稿时间:2014-09-26

一个研究重点及难点. 本文通过对观察者和目标之间对应关系的分析与研究, 给出了三种视点控制模式:等距离模式、等速度模式、匀加速模式, 设计了虚拟视景中视点控制的基本算法. 在 MFC 框架下, 应用 Vega API 函数实现 C 语言对 Vega 视景仿真平台的调用, 实现了视点对目标的随动控制方法, 使视点以不同运动方式跟随目标进行不同模式的随动观测. 在实际应用中, 利用这种随动视点方式, 可以更方便用户调整出适合需求的最佳观看方式.

1　视点方式

1.1 基本视点方式

Vega 软件, 是一个面向对象的虚拟现实平台, 它使用户以简单的操作迅速地创建、编辑和运行复杂的仿真程序. Vega 为观察者提供了四种基本视点方式, 它们分别是:手动定位视点模式(Manual), 束缚定位视点模式(Tether), 路径导航定位视点模式(Path Navigator)和运动定位视点模式(Motion Model)[2].

(1)手动定位视点:一种静态定位方式, 需要指定观察者在场景中的位置和观察场景的角度, 将视点固定在该固定位置上, 并且它的位置是固定不变的.

(2)束缚定位视点:将视点束缚在某一个运动的仿真实体上, 完全重复该实体的运动路径, 或者围绕实体旋转, 或与该实体相对位置固定, 这三种束缚视点又分别称为跟踪束缚(Tether-Follow)、旋转束缚(Tether-Spin)和固定束缚(Tether-Fixed).

(3)路径导航定位视点:在仿真开始前人为的设定一段路径, 在仿真中让视点沿着这段路径以某种方式运动(如定时、变速运动等);

(4)运动定位视点:一种最直接的动态定位方式, 视点按照运动模式的控制方式在场景中运动, 且一个观察者只能按照某一种运动模式运动[3].

1.2 视点控制模式

本文针对飞机跟踪的敌我关系, 通过对观察者和目标之间几何对应关系的分析与研究, 为实现最佳观测点观察目标的运动全过程, 设计了三种控制模式:等距离模式、等速度模式、匀加速模式.

(1)等距离模式

等距离模式就是以运动体模型为视场中心位置, 视点始终与运动体模型保持固定距离的偏移. 固定距离的偏移包括前后偏移和左右偏移, 前后偏移就是跟随飞行模式;左右偏移就是伴飞模式. 无论运动目标做何种运动(变速或匀速), 观察者(即视点)始终在固定偏移方向上与运动体模型保持固定距离对运动体模型进行实时观测.

(2)等速度模式

等速度模式里的等速并不是目标与视点速度相等, 而是视点位置在视场中始终以同一固定速度在运动模型的运动方向上跟随目标实时更新. 也就是说, 当视点的运动速度大于目标的运动速度时, 视场中观测到的目标模型就会逐渐增大, 视点与目标之间的距离就会逐渐减小;反之, 当视点的运动速度小于目标的运动速度时, 那么观测到的视景中的目标模型就会逐渐减小, 视点与目标之间的距离就会逐渐增大.

(3)匀加速模式

匀加速模式就是视点在视景仿真中以均匀加速度伴随目标模型进行运动. 在远距离跟踪视景仿真中, 观察者与目标之间的距离逐渐减小, 观察者可以逐渐清晰的观测到目标模型. 这种视点模式可以为大型战场环境的目标跟踪识别提供便利.

2　视点控制技术

2.1 视点与场景坐标的关系

视景仿真中的视点方式就是从空间某一个具体的位置观看某个全局(或局域)场景, 它控制着视锥体内的视觉表现.

场景坐标系是固定坐标系, 永久坐标系;视点的坐标系是一个临时的坐标系. Observer 的位置是视点在场景坐标系中的坐标值(x, y, z), 方位由(h, p, r)确定, 这六个元素构成了视点坐标系[4]. h 是水平旋转角(绕 z 轴, 逆时针为正), p 是以视线为轴的旋转角(绕 y 轴, 顺时针为正), r 是俯仰角(绕 x 轴, 顺时针为正). 如图 1 所示, 图中 XYZ 为场景坐标系, x'y'z'为视点坐标系, 指针方向为 h, p, r 的正方向.

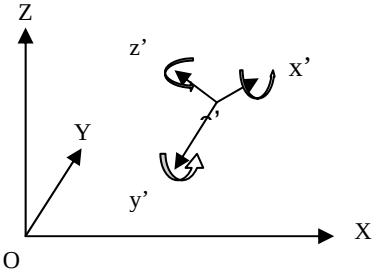


图 1　场景坐标系与视点坐标系的关系图

2.2 视点随动算法

视点随动技术是基于场景中的运动目标和观察者之间的距离及运动模式提出的. 本文将其归纳总结为三类:等距离模式、等速度模式以及加速度模式. 这几种随动视点的实现主要依据目标与观察者之间的相对关系, 其示意图如图 2 所示.

假设t_0点为计算起始点, 其对应的高度为h_{t0}, 并将h_{t0}作为基准高度. 在t_1点, 根据已知测量量, 可得到基准高度h_{t01}的计算公式如下[5]:

$$\frac{h_{t01}}{\tan\ \phi_{t0}} - \frac{h_{t01}+\Delta Y_{01}}{\tan\ \phi_{t1}} = \Delta X_{01} \tag{1}$$

即

$$\begin{aligned} &h_{t01}(\tan\phi_{t1}-\tan\phi_{t0})-\tan\phi_{t0}\Delta Y_{01} \\ &=\Delta X_{01}\tan\phi_{t1}\tan\phi_{t0} \end{aligned} \tag{2}$$

其中, ϕ_{t0}, ϕ_{t2}分别为目标飞行到t_0点, t_1点时的目标与观察者的视线角, ΔX_{t01}是t_0与t_1点的水平距离差, ΔY_{t01}是t_0与t_1点的高度距离差.

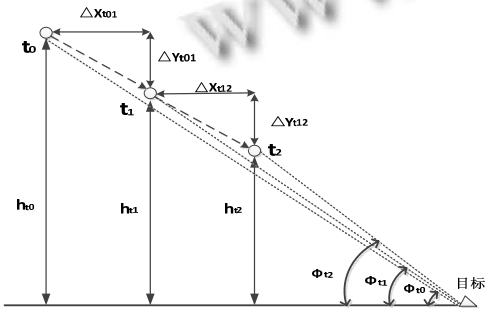


图 2 高度计算示意图

如果ϕ_{t0}与ϕ_{t2}相等或者很接近, 则有$\tan\phi_{t1}-\tan\phi_{t0}=0$, 上式可表示为:

$$\tan\ \phi_{t1} = -\frac{\Delta Y_{01}}{\Delta X_{01}} \tag{3}$$

如果ϕ_{t0}与ϕ_{t2}相差足够大时, 式(2)可表示为:

$$h_{t01}=\frac{\tan\phi_{t1}\tan\phi_{t0}}{\tan\phi_{t1}-\tan\phi_{t0}}(\Delta X_{01}+\frac{\Delta Y_{01}}{\tan\phi_{t1}}) \tag{4}$$

当选取的前后两次采样计算时间间隔内目标与观察者视线角变化足够大时, 由式(4)可完成对初始点高度的计算. 随着目标的飞行, 两者之间的视线角度差不断增大, 计算得到的基准高度值也就越接近准确值. 所以我们采取对当前时刻所有采样点进行计算取均值的方法来计算基准高度值, 即:

$$h_{t0k}=\sum_{j=0}^{k} h_{t0j}/k \tag{5}$$

其中, k为采样次数.

根据基准高度及与基准点高度点之间的高度差Δh_{t0k}, 可得, 当前时刻的高度为:

$$h_{tk} = h_{t0k} - \Delta h_{t0k} \tag{6}$$

则目标与视点之间的距离为

$$D_k = h_{tk}/\sin\phi_{t_k} \tag{7}$$

以上是目标与视点间距离的估算方法, 类似于基于相对位置信息的弹目距离的估算方法[5]. 本文假设目标的飞行速度已知, 我们以目标与视点间的估算距离公式为基础, 视点位置始终与目标固定方向(前后或左右)偏移D_k就是等距离模式;视点始终以目标的飞行速度做匀速运动就是等速度模式;根据目标飞行速度、初始距离D_1以及运动时间我们可以求出一个加速度a, 再根据$V=V_0+at$, 使视点在视场中做匀加速运动就是匀加速模式.

3 视点控制的实现

3.1 场景构建

Lynx 是 Vega 提供的图形用户界面工具集[6]. 提供了许多模块(对应于 Vega 中的类), 并通过设定参数与相互关系, 可以实现简单的仿真应用程序, 为虚拟系统定义窗口, 通道, 模型, 场景等[7]. 通过 Lynx 建立的场景模型通常存放在 ADF 文件中, 要想开发复杂的视景仿真系统, 必须在 ADF 之外, 通过编写自己的程序来满足仿真的各种需求[8]. 本文就采用 Lynx 来配置初始化 ADF 文件, 具体流程如图 3 所示. 这里的 ADF 文件中, 只预先引进所有模型, 但不加载到场景中, 而在后续的编译程序中对模型选择再加入场景.

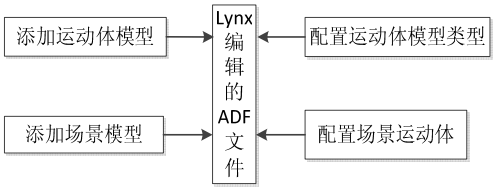


图 3 Lynx 的 ADF 文件配置

3.2 Vega API 编程

对于 Windows 平台上的 Vega 应用程序开发, 有 3 种开发环境分别为:Win32 控制台程序、传统的 Windows 应用程序(带有 WinMain()函数)和基于 MFC(Microsoft Foundation Classes)的应用程序. 相对而言, 应用 MFC 类库是 Windows 平台下开发具有

良好图形用户界面应用程序最方便的途径，Visual C++6.0 中的 MFC 类库已是一个相当成熟的类库，它提供了丰富的窗口和事件管理函数，特别是其基于文档/ 视图结构的应用程序框架，已成为开发 Windows 应用程序的主流框架结构[9]. 因此，本文也选择 MFC 作为 Vega 应用程序的主要开发工具，可直接调用 4.1 节中配置好的 ADF(Application Definition File)文件，其具体程序创建过程如图 4 所示.

图 4　MFC 框架下 Vega 程序的创建过程

3.3 视点控制

视点的控制方式通常以 C 语言的 API 形式出现. 每个 Vega 类都是一个完整的控制结构，用于管理和实现某项特定的功能，比如 vgObserver 类用于管理仿真中的视点, vgMotion 类用于管理运动模式, vglDev 类用于管理输入设备等等[3]. 在前两节的基础上，我们要在创建好的场景模型中建立一个视点(即观察者)，具体过程如图 5 所示.

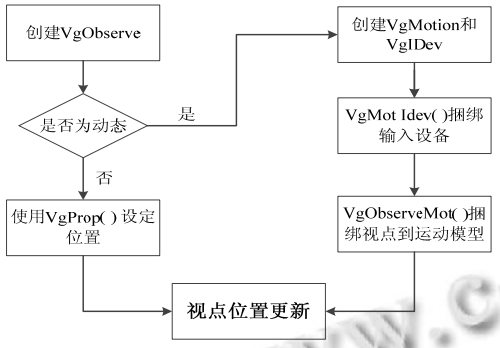


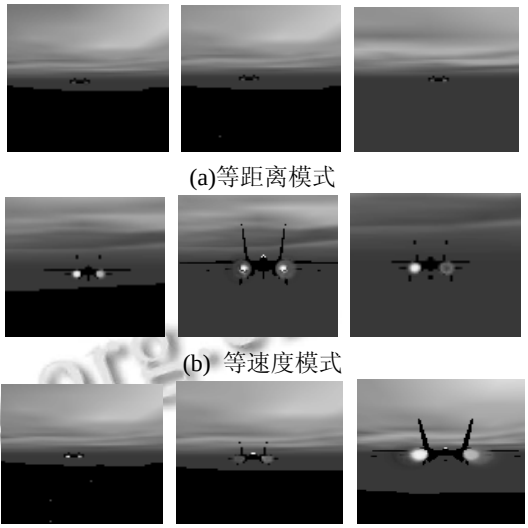
图 5　视点建立过程

视点位置更新是几种视点控制模式的实现的关键，基于 3.2 节中公式(7)对目标与视点距离的计算，然后通过对目标速度以及加速度的计算，可以编写出不同视点控制算法的应用程序. 在 Vega 应用的主循环中包括 vgsyncFrame()和 vgFrame()的调用，在调用期间运动模式被更新. 我们可以首先设定好视点、运动体和运动模型实例，然后将视点绑定在 vgPlayer 实例上，然后根据距离估算方法，分别计算出视点位置的三维坐标，最后应用 vgProp()函数更新视点的三维坐标点，使得视点位置实现不同模式下的实时更新. 调节视点函数的关键代码如下:

```
pos = vgNewPos();
vgPosVec(pos, obsx, obsy, obsz, 0, 0, 0);
vgProp(obs,VGOBS_TETHERSTATE,VGOBS_STATIC);
vgPos(obs, pos);
vgPosVec(pos, X, Y, Z, 0, 0, 0);
vgProp(obs,VGOBS_LOOKAT_TARGET,VGOBS_L_POS);
vgObservLookatPos(obs,pos);
```

4　实验结果及结论

根据上述理论研究，我们进行了不同模式下的视点控制仿真实验. 实验结果图如图 6 所示. 图中(a)(b),(c)分别为等距离模式，等速度模式和匀加速模式. 不同视点模式下，观察者(视点)对目标(飞机)的观察距离不同.



(a)等距离模式

(b) 等速度模式

(c) 匀加速模式

图 6　仿真实验图

为了实验的真实性，以匀加速模式为例，本文在实验中加入了一个 GetDistance()函数，它可以根据目标点位置和观察者位置的坐标差值来计算目标与观察者之间的距离. 本实验预先设定的路径是直线型的，所以这里只计算 Y 方向上的差值. 计算出的距离将实时更新显示在实验窗口状态栏的右下角，如图 7 所示.

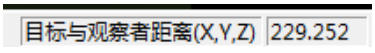


图 7　距离实时显示图

将间隔 1s 的距离数据读出，整理如表 1 所示. 匀加速模式中，视点做匀加速运动，视点跟踪做匀速运动的目标，随着时间的变化目标与视点之间的距离先逐步增大然后再减小. 根据数据做出距离和时间的曲线图如图 8 所示.

表 1 距离采样数据

时间(s)	1	2	3	4
距离(m)	743.253	836.486	856.157	737.347
时间(s)	5	6	7	8
距离(m)	597.219	354.246	19.215	0

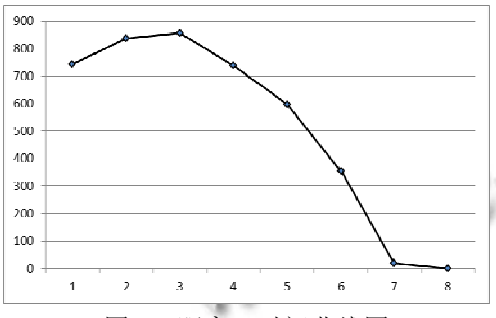


图 8 距离、时间曲线图

为了能够更加准确地观察运动体的运动姿态以及仿真场景，本文采用了多视点观察的方式. 在 Vega 中，场景的绘制是在通道中实现的，并且 Vega 支持多通道显示，因此可以在场景中设置多个视点，再使每个视点与不同的通道绑定，这样就可以实现从不同角度、不同位置观察运动体的运动状态，大大增加了视景仿真的实用价值，具体实验图如图 9 所示.

图 9 中共应用了六个通道，分别显示了六种不同的视点控制模式. 左侧是手动定位视点模式(Manual)，右侧分为五个通道窗口，按照从上到下顺序分别为等距离(跟随)模式，等距离(伴飞)模式，等速度模式，匀加速模式，以及旋转束缚模式.

Vega 提供的基本视点方式及其组合已不能满足用户的观察需求，而本文在 Vega 的基础上，根据实际需求，提出的三种视点控制模式，在视景仿真实验中达到了对视点方式的灵活控制，提高了观测距离的可控性，为大范围战场环境的视景仿真提供了便利，同时具有较强的现实直观性，增强了视景的真实感和交互性.

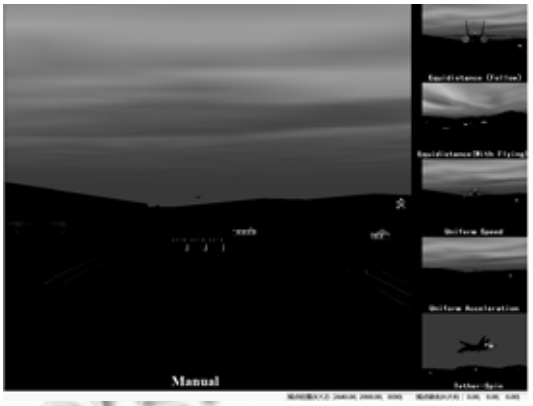


图 9 多通道显示仿真实验图

参考文献

1 宋志明,康凤举,褚彦军,等.Vega 环境中一种自由视点方式的开发.计算机仿真,2004,21(2):109–111.
2 杨洋,刘志勤.Vega 场景中一种自由视点漫游的实现.计算机与现代化,2010,(2):84–87.
3 薛军涛,贺怀清.基于Vega的视景仿真系统中自由视点的开发.2006 系统仿真及其应用学术交流会论文集.2006. 269–272.
4 魏博,吴国平,朱士峰,等.基于MFC框架的Vega虚拟场景设计和实现研究.河南科学,2008,26(4):454–458.
5 温求遒,夏群利,蔡春涛,等.基于图像导引头及惯导相对位置信息的弹目距离估计算法.北京理工大学学报,2012,32 (2):141–145.
6 王传喜,赵刚.基于Vega平台的视点控制技术的研究与设计.电子设计工程,2012,20(5):190–192
7 李瑞,刘鹏远,张锡恩,等.Vega 程序设计在 MFC 中的应用.计算机工程与设计,2002,23(8):55–57.
8 董博,马立元,罗婧,等.基于 MFC 的 Vega 应用程序设计.微计算机信息,2006,22(4):264–265.
9 李明泽,毛学刚,范文义,等.MFC 框架下基于 Vega 的视景驱动程序设计研究.现代计算机(专业版),2007,(10):93–95,112.