基于 WebGL 的三维 GIS 空间算法的研究与实现

王星捷[1,2] 卫守林[3]

[1](核工业西南物理研究院 四川 成都 610041)

[2](成都理工大学工程技术学院 四川 乐山 614007)

[3](昆明理工大学信息工程与自动化学院 云南 昆明 650093)

摘 要 WebGL 被广泛运用到 Web 的三维场景可视化和三维 WebGIS 场景实现中。目前将 WebGL 应用在三维 GIS 中的应用研究主要集中在处理图形引擎、三维建筑加载和三维场景可视化等方面,而在采用 WebGL 技术实现三维 GIS 以及空间算法上的研究较少。采用 WebGL 技术实现三维 GIS,在进行数据测试的过程中,出现二、三维空间数据不同步、错位情况严重、路径分析三维展示平滑性差、场景漫游旋转时三维对象发生位置偏移和坐标错位等问题。针对以上问题,重点研究二、三维空间数据同步算法、三维运动轨迹处理算法和三维漫游处理算法。以实际的城市三维数据为例进行算法实验分析,结果表明:利用该方法保证了三维 GIS 中二、三维空间数据的同步,减少了三维漫游过程三维对象位置偏移和坐标错误情况,在路径分析的三维效果中体现了良好的视觉感知。

关键词 WebGL WebGIS 二、三维空间数据同步 三维运动轨迹 三维漫游处理

中图分类号 TP319 **文献标识码** A DOI:10.3969/j.issn.1000-386x.2019.04.008

3D GIS SPACE ALGORITHM BASED ON WEBGL

Wang Xingjie[1,2] Wei Shoulin[3]

[1](*Southwestern Institute of Physics*, *Chengdu 610041*, *Sichuan*, *China*)

[2](*The Engineering and Technical College of Chengdu University of Technology*, *Leshan 614007*, *Sichuan*, *China*)

[3](*Faculty of Information Engineering and Automation*, *Kunming University of Science and Technology*, *Kunming 650093*, *Yunnan*, *China*)

Abstract WebGL has been widely utilized in the 3D scene visualization on Web and the implementation of 3D WebGIS scene. At present, in the area of 3D GIS, WebGL is mainly studied in graphics engine processing, 3D architecture loading and 3D scene visualization, but lack of research on 3D GIS realization and space algorithm by applying WebGL. In this paper, 3D GIS was implemented by adopting WebGL. During the data test, data from 2D and 3D space was unsynchronized and seriously mismatched, and 3D display indicated poor smoothness during path analysis. In the process of rotating 3D objects and scenes, position deviation and coordinate dislocation occurred. In view of the above problems, we studied 2D and 3D space synchronization algorithm, 3D object trajectory processing algorithm and 3D rotation processing algorithm. With the example of actual 3D city data, the algorithm experimental analysis indicates that the proposed method guarantees the data synchronization of 2D and 3D space, reduces the position deviation and coordinate dislocation and reflects well visual perception in the 3D effects of path analysis.

Keywords WebGL WebGIS 2D-3D space data synchronization 3D motion trajectory 3D rotation processing

---

收稿日期:2018-11-25。国家自然科学基金项目(11403009)。**王星捷**,副教授,主研领域:智慧城市,三维空间数据处理,大数据应用。**卫守林**,讲师。

0 引 言

当前多数的 Web 端的 GIS 平台主要以二维的为主,许多成熟的三维 GIS 平台,都已经开发成产品,不仅加载 Web 三维场景地图大多需要下载额外的插件来实现,而且需要昂贵的平台使用费用。WebGL 技术[1]可以直接利用硬件来渲染三维场景,不用下载额外的插件。现在虽然有许多关于利用 WebGL 技术来渲染[2]三维 GIS 场景的研究论文,但实现效果不太理想,大多只提供场景模型加载功能,没有具体的二维地图矢量数据的支撑,不能进行空间分析。而且只能实现单一的三维场景[3],不能实现与二维数据的联动等多方面的空间功能。

目前浏览器对 WebGL 提供支持,在未来 WebGL 与地理信息服务技术相结合是发展的趋势。本文基于 Web Graphics Library(WebGL)结合 Geographic Informatica Service(地理信息服务)对三维数字城市技术进行了深入研究,设计了一套基于 WebGL 开发三维 GIS 的技术[4],充分利用了 WebGL 与 ArcGIS Server 的结合,方便地实现了三维编辑、三维查询[5]和三维分析等功能。由于三维模型[6]和三维场景的复杂性[7],在实现的过程中出现了一些问题:(1) 系统在平移和旋转时,二维空间数据处理和三维模型的显示发生了位移,没有保持空间信息的同步;(2) 实现路径分析数据三维体现时,轨迹显示效果差;(3) 在实现二、三维视图切换,鹰眼滑动,三维漫游以及三维场景旋转时,发现旋转控制效果不理想,出现了偏移,坐标发生错位。

针对上述的问题,本文重点研究和分析了二、三维空间数据同步算法,三维对象运动轨迹处理算法和三维漫游处理算法。

1 二、三维空间数据同步算法

二、三维空间数据同步是三维 GIS 空间分析的基本功能,基本原理是将二维空间数据与三维模型通过空间坐标进行同步的显示。在小范围的数据中,可以通过坐标实现同步,而面对较大的三维模型和二维空间数据时,当进行三维操作时,二、三维出现明显的坐标偏差和错位,当进行漫游和旋转时,情况会更为严重。

为了让二维空间数据与三维模型数据的同步[8]显示,在通过空间坐标绑定的基础上,结合在二维空间数据的 map 对象中的地图范围变化事件中绑定三维场景的相机同步控制函数。三维模型显示二维数据需要对相机进行控制,实现的过程比较复杂。本文对二维空间数据控制三维场景相机的位置进行了算法计算。

在三维场景的使用的相机为透视投影相机[9],因为相机视野中的物体尺寸会随着与相机的距离变远而变小,更接近于人眼观察的效果,所以选用透视投影相机会达到更加逼真的效果。透视相机原理如图 1 所示。

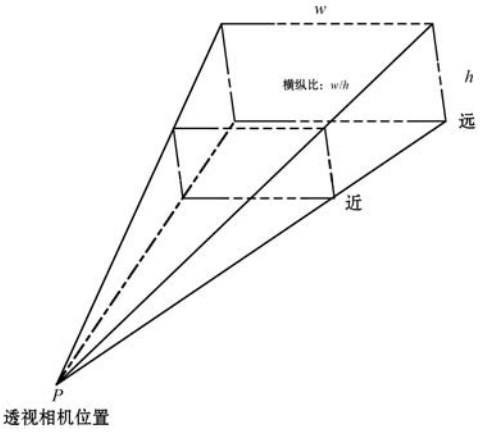


图 1 透视相机的观察事物的原理

二维空间数据中的展示的地图相当于三维场景的地面,为了保证相机俯视时看到的效果能与二维地图显示范围一致,必须保证网页中放置 map 对象的控件元素和 Three. js 的三维场景 scene 控件容器元素的长宽比例 w/h 保持一致。当二维地图显示范围与 scene 控件显示的地图范围一致时,假设相机的位置为 P,相机的视角为 θ,此时地面范围的宽度 W 与相机高度 H 的关系如图 2 所示。

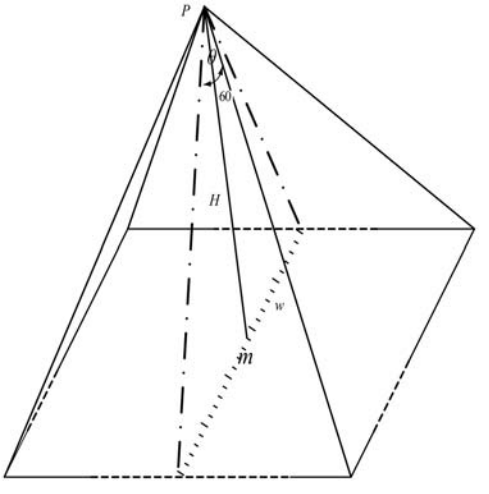


图 2 根据地图范围确定相机位置

WebGL 场景使用的是右手坐标系,Y 轴向上。为了方便计算,将 WebGL 场景中的 XZ 坐标作为的经纬度坐标的映射。$m(X_m, 0, Z_m)$ 点为地图范围的中心点,同时也是相机位置 P 点在地面上的投影点。通过二维地图 map 控件可以获取到当前显示的二维地图的地理坐标范围,再经过坐标转换得到场景坐标的范

围为:

$$((X_{\min},Z_{\min}),(X_{\max},Z_{\max}))$$

可以得到范围中心点的坐标为:

$$M\left(\frac{X_{\min}+X_{\max}}{2},\frac{Z_{\min}+Z_{\max}}{2}\right)$$

P 点坐标的计算公式如下:

$$X_p=X_m=\frac{X_{\min}+X_{\max}}{2} \tag{1}$$

$$Z_p=Z_m=\frac{Z_{\min}+Z_{\max}}{2} \tag{2}$$

H、W 的计算公式如下:

$$H=\frac{W}{2}\times\tan\theta \tag{3}$$

$$W=Z_{\max}-Z_{\min} \tag{4}$$

最终计算得到相机位置为:

$$P\left(\frac{X_{\min}+X_{\max}}{2},\tan\theta(Z_{\max}-Z_{\min}),\frac{Z_{\min}+Z_{\max}}{2}\right)$$

当三维模型视图与二维空间数据同步时,需要将场景中的相机移动到 P 点位置并向朝向 Y 轴负方向。

二、三维空间数据的同步,可以方便将 ArcGIS Server 服务中的二维地图查询、缓冲查询、编辑、邻近设施分析等功能移植到三维视图,是实现三维数字城市基本功能的基本。

2　三维对象运动轨迹处理算法

物体的运动轨迹主要来自于路径分析结果和二、三维空间数据同步时相机的运动轨迹。这些轨迹都是由分布不均的点集生成的折线,需要对轨迹线进行插值算法分析,使得到的轨迹变得平滑,使物体运动起来更加平缓。本文对运动轨迹进行插值时分别采用了样条函数插值法和 CatmullRom 样条函数插值法,对物体的运动轨迹进行插值和对比。CatmullRom 样条函数[10]与普通样条函数的图解如图 3 所示。

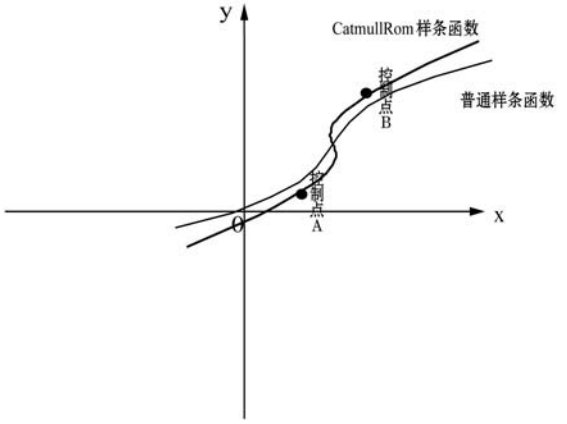


图 3　CatmullRom 样条函数与普通样条函数对比

可以看出一般的样条函数生成的曲线只会接近控制点,不会穿过控制点,CatmullRom 样条函数会穿过每一个控制点,而三维路径分析和漫游需要经过每一个道路线的拐点(控制点),所以选用 CatmullRom 样条函数来对物体的运动轨迹进行插值。

插值完成后得到路径为一条平滑的曲线,这时只需在场景渲染时的每一帧读取样条函数上对应的坐标点坐标,再将移动的物体的坐标设置为读取到的坐标,每一帧都执行此操作,就可以实现物体在场景中按照插值轨迹的进行平滑的移动。

3　三维漫游处理算法

三维漫游无非是平移、缩放、旋转等操作,而旋转是实现三维运动处理的关键的部分。旋转的过程是物体从初始状态,经历旋转,达到结束状态。在三维空间中控制物体旋转有三种方式:欧拉角[11]、旋转矩阵和四元数[12]。

欧拉角:这是最直观也最容易理解的旋转表达方式,物体从初始状态到结束状态可以将物体分别绕 x、y、z 轴按顺序旋转不同的角度完成。一共用三个旋转值和一个旋转顺序表示:

$$(x,y,z,\text{order})$$

其中,x、y、z 分别表示物体绕 x、y、z 轴旋转的角度。order 表示三次旋转的顺序。当用欧拉角控制[13]对象旋转时其实需要将物体绕三个坐标轴的三次旋转,并且如果旋转的顺序不同得到的结果也不相同,一个欧拉角的参数中包含绕三个坐标轴旋转的角度和三次旋转的顺序四个参数。除了三次旋转中两次为零情况,直接进行线性插值无法得到线性旋转轨迹,也就是说当相机按得到的旋转轨迹旋转时可能会出现抖动的现象,出现这种情况的原因被称之为万向节锁。

旋转矩阵:其实无论经历怎样的旋转物体都可以通过绕一个特定的旋转轴按一定方向旋转一定角度来达到结束状态。也就是绕一个向量旋转一个角度,但是旋转矩阵中变化参数比较多,对于计算和控制状态并不是那么容易。

四元数:四元数描述旋转的原理和旋转矩阵类似[14],区别就是四元数就引入了复数的思想,对复数进行了延伸,可以把它看成一个四维复数,同样将一个旋转轴和一个绕它变化的角度进行变形,使之成为一个四元数,一共有四个值。这样可以遵循复数的运算,包括加减乘除以及积分等等。

使用四元数法控制物体旋转时物体只需进行一次旋转,所有的旋转是绕着一条直线旋转一定角度。只

要指定好这条直线,再指定一个角度即可控制物体的旋转。一个四元数需要的参数有三个直线表达式和旋转的角度值,共含四个参数。四元数旋转坐标表示如下:

$$\{x:k_x\times\sin(a/2),y:k_y\times\sin(a/2),$$
$$z:k_z\times\sin(a/2),w:\cos(a/2)\}$$

其中,a 代表绕轴(k_x,k_y,k_z)旋转的角度),转换成复数形式为:

$$\cos(a/2)+(k_x\times\sin(a/2))i+(k_y\times\sin(a/2))j+(k_z\times\sin(a/2))k$$

这里的 i、j、k 表示虚数单位,仅仅代表他们分别处于不同的维度,不代表具体值。这样以来就可以实现对旋转进行叠加、加速等效果的控制,计算起来比欧拉角(三次旋转的合成)和旋转矩阵(四元数变形的更多的变量的矩阵形式)更加简单[15]。

通过上面的解释,四元数的表达式可分解为 k_x、k_y、k_z、angle,设角度 a 为 angle。而欧拉角的表达式无法分解,设它的表达式为(x,y,z,'xyz'),'xyz' 只表示顺序不表示数值,因为三个参数都是角度值,具有实际意义无法分解,那么参数有三个,设置让三位场景中的物体绕指定向量匀速旋转,旋转过程如图 4 所示。

图 4 测试对象 x,y,z 轴的旋转状态

以下四元数表达式各参数和欧拉角表达式各参数数据采集自该对象绕单位向量(0.826 803,0.447 628,0.340 626)匀速旋转一周的不同时间点采集到的参数数值。如表 1、表 2 所示。

表 1 四元数表达式各参数的在旋转中的数值

time	k_x	k_y	k_z	angle
8.890	0.826 803	0.447 628	0.340 626	-2.947 070
9.347	0.826 803	0.447 628	0.340 626	-2.747 930
9.780	0.826 803	0.447 628	0.340 626	-2.558 750
10.214	0.826 803	0.447 628	0.340 626	-2.369 490
10.649	0.826 803	0.447 628	0.340 626	-2.179 660
11.091	0.826 803	0.447 628	0.340 626	-1.986 600
11.518	0.826 803	0.447 628	0.340 626	-1.800 470

续表 1

time	k_x	k_y	k_z	angle
11.969	0.826 803	0.447 628	0.340 626	-1.603 560
12.423	0.826 803	0.447 628	0.340 626	-1.405 770
12.857	0.826 803	0.447 628	0.340 626	-1.216 150
13.289	0.826 803	0.447 628	0.340 626	-1.027 580
13.715	0.826 803	0.447 628	0.340 626	-0.841 870
14.167	0.826 803	0.447 628	0.340 626	-0.644 770
14.592	0.826 803	0.447 628	0.340 626	-0.459 110
15.026	0.826 803	0.447 628	0.340 626	-0.269 790
15.472	0.826 803	0.447 628	0.340 626	-0.075 250
15.948	0.826 803	0.447 628	0.340 626	0.132 373
16.421	0.826 803	0.447 628	0.340 626	0.338 757
16.887	0.826 803	0.447 628	0.340 626	0.542 065
17.330	0.826 803	0.447 628	0.340 626	0.735 379
17.778	0.826 803	0.447 628	0.340 626	0.930 777
18.260	0.826 803	0.447 628	0.340 626	1.141 104
18.719	0.826 803	0.447 628	0.340 626	1.341 419
19.149	0.826 803	0.447 628	0.340 626	1.529 336
19.597	0.826 803	0.447 628	0.340 626	1.724 838
20.074	0.826 803	0.447 628	0.340 626	1.932 968
20.517	0.826 803	0.447 628	0.340 626	2.125 903
20.984	0.826 803	0.447 628	0.340 626	2.329 890
21.430	0.8268 03	0.447 628	0.340 626	2.524 290
21.853	0.826 803	0.447 628	0.340 626	2.709 117
22.297	0.826 803	0.447 628	0.340 626	2.902 617
22.737	0.826 803	0.447 628	0.340 626	3.094 657
23.187	0.826 803	0.447 628	0.340 626	-2.991 910
23.632	0.826 803	0.447 628	0.340 626	-2.797 890
24.063	0.826 803	0.447 628	0.340 626	-2.609 760
24.509	0.826 803	0.447 628	0.340 626	-2.415 090
24.979	0.826 803	0.447 628	0.340 626	-2.210 320
25.418	0.826 803	0.447 628	0.340 626	-2.018 650
25.888	0.826 803	0.447 628	0.340 626	-1.813 540
26.358	0.826 803	0.447 628	0.340 626	-1.608 370

表 2 欧拉角表达式各参数的在旋转中的数值

time	x	y	z
8.890	-2.590 36	0.490 905	-1.133 88
9.347	-2.424 68	0.379 031	-1.135 91
9.780	-2.273 22	0.273 796	-1.120 01
10.214	-2.123 55	0.171 965	-1.088 56
10.649	-1.972 15	0.075 587	-1.042 47
11.091	-1.814 25	-0.014 100	-0.981 40

续表 2

time	x	y	z
11.518	-1.656 470	-0.090 250	-0.909 590
11.969	-1.482 520	-0.157 180	-0.820 790
12.423	-1.300 540	-0.207 670	-0.720 110
12.857	-1.120 440	-0.238 190	-0.615 430
13.289	-0.938 140	-0.249 720	-0.506 880
13.715	-0.758 300	-0.242 200	-0.399 560
14.167	-0.570 270	-0.214 120	-0.289 570
14.592	-0.398 220	-0.170 020	-0.193 240
15.026	-0.229 240	-0.109 340	-0.104 990
15.472	-0.062 660	-0.032 860	-0.026 670
15.948	0.108 209	0.061 584	0.041 812
16.421	0.273 165	0.165 514	0.093 575
16.887	0.433 958	0.274 751	0.127 804
17.330	0.588 810	0.382 326	0.143 707
17.778	0.751 539	0.492 095	0.141 321
18.260	0.939 706	0.608 018	0.114 460
18.719	1.138 651	0.712 199	0.060 484
19.149	1.349 931	0.799 777	-0.020 700
19.597	1.601 364	0.874 417	-0.141 230
20.074	1.905 639	0.927 302	-0.309 410
20.517	2.210 873	0.944 960	-0.489 990
20.984	2.533 968	0.927 634	-0.681 310
21.430	2.819 580	0.879 450	-0.839 910
21.853	3.060 756	0.810 950	-0.957 710
22.297	-3.001 300	0.722 436	-1.045 500
22.737	-2.807 770	0.623 590	-1.100 920
23.187	-2.629 070	0.515 947	-1.130 480
23.632	-2.465 450	0.407 100	-1.137 310
24.063	-2.313 730	0.301 931	-1.125 890
24.509	-2.159 620	0.196 062	-1.097 490
24.979	-1.996 800	0.090 668	-1.050 870
25.418	-1.840 830	0.000 101	-0.992 500
25.888	-1.667 750	-0.085 280	-0.915 030
26.358	-1.486 860	-0.155 730	-0.823 110

从表 1、表 2 中的数据可以看出,从初始到结束状态四元数表达式和欧拉角表达式各个参数值的变化情况,四元数表达式参数中只有一个角度 angle 在呈线性变化,而欧拉角的三个参数都在同时进行变化,但是变化规律并不统一。将表 1、表 2 中的数据绘制成数值变化图,如图 5 所示。

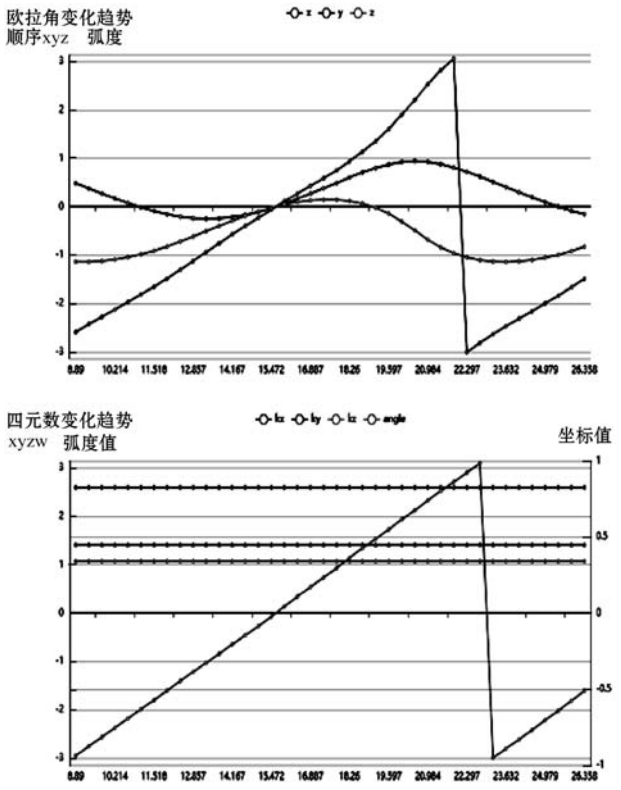


图 5　欧拉角和四元数数值变化图

从图 5 中很直观分析出,四元数可以通过对角度参数进行简单的线性插值就可以实现旋转控制,而使用欧拉角表达式来控制物体旋转状态却不能用简单的线性差值来实现,如果要使用欧拉角来控制旋转,那么就要分别求出每一个 *xyz* 每次旋转角度的变化规律,显然复杂程度比四元数高出了很多。

上述的数据只是简单的绕一根轴匀速旋转,是最简单的旋转过程,即使是最简单的旋转过程也很难使用欧拉角表达式通过特定的差值规则来控制旋转,其实物体在实际情景下的旋转过程是多方位的旋转,在转动过程中并不只是在绕某一个旋转轴匀速旋转,而是不断在改变旋转轴和速度。截取一个尽可能短的时间段,可以把这个时间段的旋转近似看成不变的,可以使用四元数表达式和欧拉角来表达此时的旋转,这个动作就可以看成旋转轴在不断旋转,旋转速度不断变化的旋转组合,同时把这些旋转看成旋转轴也在不停地旋转。使用四元数表示旋转,可以使用四元数特有的加减乘除运算法则来计算出每个时段旋转的四元数最终表达,在三维场景中使用这个最终表达来设置物体在对应时段的旋转状态,就可以达到对物体旋转的很好控制,可以对四元数运用一些现有的插值算法来控制物体平滑旋转。而欧拉角并不能够像四元数这样找出规律并应用运算规则,旋转中,欧拉角三个值都在不停变换,仅仅表示一个旋转状态,用来描述和控制物体旋转过程很难找出其特定的规律。

与欧拉角相比四元数描述旋转的优势如下:插值结果为线性轨迹,没有万向节锁,具有唯一表达式。而欧拉角旋转顺序不同会有多种不同的表达式。

选用四元数法来控制物体的旋转,只需要对物体的旋转状态和最终旋转状态的四元数表达式进行线性插值,场景在渲染每一帧时为物体应用每一个插值点的四元数变换,就能保证物体旋转的平滑过渡。

4 实验与分析

本文测试的数据是一个某小城市的数据,范围大小为 10 平方公里,模型数量超过 5 000 个,其中矢量面拉伸生成的模型 1 421 个,导入外部模型 3 600 多个,按平台设计需要还包含每个建筑物的高程数据、模型参数(路径、大小、旋转)、道路宽度、建筑物名称等必要属性数据。

(1) 二、三维空间数据同步分析　利用二、三维空间数据同步算法计算出二维地图当前显示范围的场景中相机对应的位置,同时利用三维对象运动控制算法对相机的运动过程进行插值计算。让场景中的相机能够动画过渡到根据二维地图显示范围计算出的相机位置。从而实现视图同步。二、三维视图同步效果如图 6 所示。

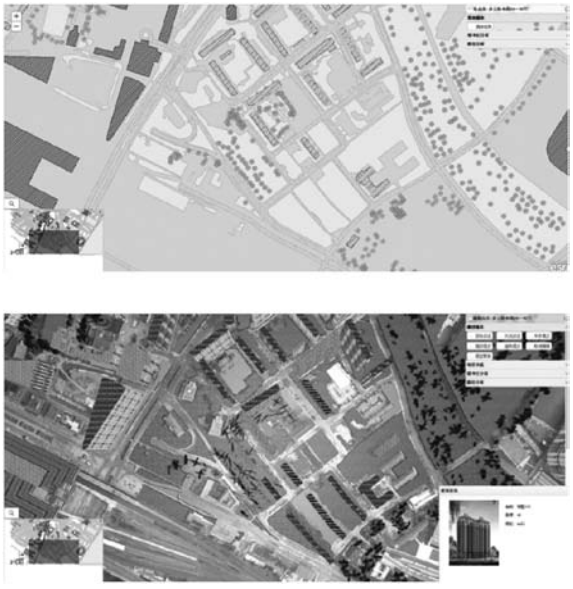
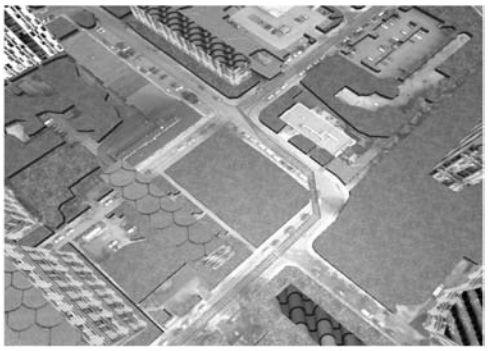
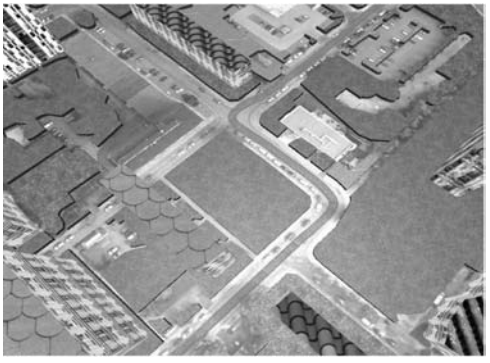


图 6　二、三维空间数据同步效果

(2) 运动轨迹分析对比　用路径分析生成的路线来测试运动轨迹,图 5 中展示了采用普通样条插值处理和 CatmullRom 样条插值处理的不同效果。从图 7(a)中可以看出,普通样条插值处理的效果,体现为折线,特别是在转角处的处理比较粗糙,没有完全经过控制点,只是接近。从图 7(b)中可以看出,CatmullRom 样条插值处理的效果较好,体现了平滑的路径曲线,并且通过了每一个路径坐标控制点。



(a) 普通样条插值处理



(b) CatmullRom 样条插值处理

图 7　三维对象运动轨迹处理效果

5 结　语

本文在基于 WebGL 的实现三维 GIS 技术基础上,对二、三维空间数据同步算法、三维对象运动轨迹处理算法和三维漫游处理算法进行了分析和研究。分析了二、三维视图同步算法的原理,并设计了对应的模型和算法;分析和研究了轨迹处理的算法;详细分析了三维漫游处理算法和实现过程。最终通过实验分析和数据对比,证明了本文设计和研究算法的科学性,同时通过展示的效果,证明了本文研究的场景控制算法达到了预计的效果。本文研究的三维视图算法具有实际应用和研究的价值,实现的技术为三维数字城市的开发提供了技术参考和理论依据。

参 考 文 献

[1] 刘恒星, 范湘涛, 刘健, 等. WebGL 技术下的 SPH 流体模拟方法[J]. 小型微型计算机系统, 2017, 38(10): 2406-2411.

[2] 郑维欣, 贾金原. 基于 PBR 的轻量级 WebGL 实时真实感渲染算法[J]. 系统仿真学报, 2017, 29(11): 2693-2699.

[3] 李兴田,张丽萍. 基于 WebGL 的工程制图网络虚拟模型库的开发[J]. 图学学报,2016,37(6):836-841.

(下转第 85 页)

可使产品在单元间不断转移并选择在高效机器上加工。大量随机数值实验也表明,在不同生产规模的单元化制造实践中,构建合理的单元间产品物流对于有效降低运作成本具有重要的现实意义。

本研究还可从以下方面做进一步探索。首先,可针对工业互联网环境下,考虑设备移动、租赁及共享等因素对单元制造系统资源配置的影响进行研究;其次,可将单周期生产情况拓展至多周期;最后,可设计有效的启发式算法,以改善遗传算法初始种群的质量,实现快速收敛,这对于工业级的大规模问题而言也有重要的研究价值。

参 考 文 献

[1] Liu C, Wang J, Leung Y T. Integrated bacteria foraging algorithm for cellular manufacturing in supply chain considering facility transfer and production planning[J]. Applied Soft Computing, 2018, 62: 602 - 618.

[2] 赵东方. 复杂机电产品高柔性数控生产单元构建与调度研究[D]. 北京:北京科技大学,2017.

[3] 张庆良. 丰田制造系统关键要素[J]. 柴油机设计与制造, 2017, 23(4): 53 - 56.

[4] Liu C, Wang J, Leung Y T, et al. Solving cell formation and task scheduling in cellular manufacturing system by discrete bacteria foraging algorithm[J]. International Journal of Production Research, 2016, 54(3): 923 - 944.

[5] 张媛. 供应链环境下制造业物流成本控制研究[D]. 西安: 长安大学,2013.

[6] Koulamas C, Kyparisis G J. Single-machine and two-machine flowshop scheduling with general learning functions [J]. European Journal of Operational Research, 2007, 178 (2): 402 - 407.

[7] Liu C, Wang J, Leung Y T. Worker assignment and production planning with learning and forgetting in manufacturing cells by hybrid bacteria foraging algorithm [J]. Computers & Industrial Engineering, 2016, 96: 162 - 179.

[8] Toksan M D, Ank O A. Single machine scheduling problems under position-dependent fuzzy learning effect with fuzzy processing times[J]. Journal of Manufacturing Systems, 2017, 45:159 - 179.

[9] Hazarika M, Laha D. Genetic algorithm approach for nachine cell formation with alternative routings[J]. Materials Today Proceedings, 2017, 5(1): 310 - 314.

[10] Soto R, Crawford B, Olivares R, et al. An imperialist competitive algorithm to solve the manufacturing cell design problem[J]. Applied Computational Intelligence and Mathematical Methods, 2018: 102 - 113.

[11] Bychkov I, Batsyn M. An efficient exact model for the cell formation problem with a variable number of production cells [J]. Computers & Operations Research,2017,91:112 - 120.

[12] Aalaei A, Davoudpour H. A robust optimization model for cellular manufacturing system into supply chain management [J]. International Journal of Production Economics, 2017, 183: 667 - 679.

[13] 杨国俊,陈健,孙思蒙,等. 基于遗传算法的车间布局优化研究[J]. 机械研究与应用,2016(1): 12 - 14.

[14] Paydar M M, Saidi-Mehrabad M. A hybrid genetic algorithm for dynamic virtual cellular manufacturing with supplier selection[J]. International Journal of Advanced Manufacturing Technology, 2017, 92(5/8):3001 - 3017.

[15] 韩忠华,朱一行,史海波,等. 基于改进紧致遗传算法的柔性流水车间组批排产优化问题研究[J]. 系统工程理论与实践,2016, 36(6): 1616 - 1624.

(上接第 68 页)

[4] Jenny B, Šavrič B, Liem J. Real-time raster projection for web maps[J]. International Journal of Digital Earth, 2015, 9(3):215 - 229.

[5] 左正,胡昱,段云岭, 等. 基于第 5 代 HTML 标准的拱坝工程三维可视化网络平台[J]. 计算机辅助设计与图形学学报,2014,26(4):590 - 596.

[6] 李海生,刘成,蔡强, 等. 三维模型网格数据压缩技术研究[J]. 系统仿真学报,2013,25(9):2150 - 2156.

[7] 周静,彭冲. 三维复杂场景路径规划仿真系统设计与分析[J]. 计算机仿真,2015,32(6):364 - 367.

[8] 谈心,佘江峰. 二维矢量线符号在三维地形表面的贴合渲染方法[J]. 地球信息科学学报, 2015,17(12):1483 - 1489.

[9] 闫利,费亮. 摄影测量成像原理的相机模拟及其在纹理映射中的应用[J]. 测绘通报,2013(5):28 - 30.

[10] 李军成,刘成志,易叶青. 带形状因子的 C ~ 2 连续五次 Cardinal 样条与 Catmull-Rom 样条[J]. 计算机辅助设计与图形学学报,2016,28(11):1821 - 1831.

[11] 项伟,白征东,汤晓禹. 阻尼最小二乘法在任意欧拉角坐标转换中的应用[J]. 大地测量与地球动力学,2016,36(2):167 - 170.

[12] 冯贺, 常国权, 郭晓波. 超复数 Fourier 变换耦合位置扰乱的彩色图像哈希算法[J]. 计算机科学与探索, 2017, 11(11):1837 - 1848.

[13] Sheng Q H, Shao S, Xiao H, et al. Relative Orientation Dependent on Dual Quaternions[J]. The Photogrammetric Record, 2015, 30(151):300 - 317.

[14] Zha C, Ding X, Yu Y, et al. Quaternion-based nonlinear trajectory tracking control of a quadrotor unmanned aerial vehicle [J]. Chinese Journal of Mechanical Engineering, 2017, 30(1):77 - 92.

[15] Gou X M, Liu Z W, Wei Liu.... Filtering and tracking with trinion-valued adaptive algorithms[J]. Frontiers of Information Technology & Electronic Engineering, 2016, 17(8): 834 - 840.