文章编号: 1671-0444(2015)04-0448-07

多 Kinect 实时室内动态场景三维重建

邓念晨, 杨旭波
(上海交通大学 软件学院,上海 200240)

摘要: 采用多个 Kinect 从不同角度同时捕获场景,将它们的深度图和彩色图结合在一起,通过数据预处理、顶点构建、点云注册和表面重建等步骤得到场景三维模型. 整个流程均在 GPU 上实现以加速运算,实现了基于 GPU 的迭代最近点算法、基于 GPU 的八叉树构建、基于有向距离函数的表面重建等关键算法. 试验中,整个算法运行帧率达到 8.74 f/s;重建分辨率达到约 5.9 mm. 试验表明,算法基本满足实时动态场景重建的要求,重建模型的精度满足非精确计算类应用的需求.

关键词: 三维重建;室内动态场景;多 Kinect 设备;迭代最近点配准;Marching Cubes 算法
中图分类号: TP 391　　文献标志码: A

Real-Time Dynamic Indoor Scene 3D Reconstruction Using Multiple Kinect

DENG Nian-chen, *YANG Xu-bo*
(School of Software, Shanghai Jiaotong University, Shanghai 200240, China)

Abstract: Multiple Kinect devices are used to capture a scene from different views at the same time. Depth and RGB images of the scene are combined to model the scene. The reconstruction process mainly includes data preprocessing, vertex construction, point clouds registration and surface reconstruction. All processes are GPU implemented to speed up the calculating. Key algorithms implemented in this subject include iterative closest point based on GPU, octree construction based on GPU, surface reconstruction based on the signed distance function. Experimental results show that the algorithm efficiency reaches 8.74 f/s and the resolution of models reconstructed are about 5.9 mm. The experimental results prove that, the reconstruction process implemented meets the requirements of the real-time dynamic scene reconstruction and the precision of reconstructed models meets the need of non-exact calculation application.

Key words: 3D reconstruction; indoor dynamic scene; multiple Kinect devices; iterative closest point registration; Marching Cubes algorithm

场景三维重建技术是利用传感器和计算机,通过对场景的纹理、深度等元数据进行分析和处理后自动(或半自动)地构建场景的三维模型. 目前,场景重建技术在工业模拟、智能机器人等诸多方面都发挥着十分重要的作用. 然而,较长的计算时间和昂贵的设备极大地阻碍了这一技术在协作办公和日常生活领域的推广应用. 因此,基于廉价设备的实时场景三维重建技术具有巨大的研究意义和应用价值. 本文以远程协作应用为背景,采用 Kinect 设备捕获场景数据,研究如何实时重建满足一般用户视觉精度要求的场景模型.

人们对三维重建技术的研究已达数十年,也提

收稿日期:2014-11-14
基金项目:国家自然科学基金资助项目(61173105, 61373085)
作者简介:邓念晨(1990—),男,山东潍坊人,硕士研究生,研究方向为计算机图形学. E-mail:dengnianchen@sjtu.edu.cn
杨旭波(联系人),男,教授,E-mail:yangxubo@sjtu.edu.cn

出了许多方法. 根据原始场景数据的类型，三维重建技术主要分为基于深度数据的三维重建[1-5]和基于纹理数据的三维重建[6-7]，其中大多数方法是基于深度数据的. 基于纹理数据的方法具有硬件成本低廉的优势，但物体表面的纹理本身受环境光影响较大，而且重建过程包含大量经验和假设，一般情况下的重建精度往往不能满足要求. 而基于深度数据的三维重建方法更为简单、稳定，易于推广应用. 根据重建模型的抽象等级，三维重建技术可以分为 3 个层次[4]：高层次关注的是一种或者一类物体，主要用于物体识别[8]；中层次使用基本几何图元构建模型，主要用于物体分割[1]；低层次则完全构建模型网格表面，不受任何假设的限制，用于对物体或场景精确建模以便重现[2, 4-5]. 本文重点研究基于深度数据的网格模型精确建模方法.

近几年，由于通用图形处理器(GPGPU)的推广应用以及类似 Kinect 这样具有实时深度获取能力的设备的出现，实时三维重建技术开始成为研究热点. 文献[9]利用 Kinect 从多个角度拍摄物体，然后离线重建完整的物体模型. 而文献[10]则在 Kinect 输入数据上提出了更好的预处理方法. 文献[5, 11]利用 Kinect 设备和 GPU 运算达到了实时重建的效率. 文献[12]继续进行了改进，使得重建算法能较为快速地响应场景中发生的变化. 而文献[13]则结合图像追踪技术来重建和追踪非刚体物体的变化. 值得一提的是，上述方法都仅使用了单台 Kinect 设备，由于受限于单一设备的视野局限，需要手持设备移动扫描才能获取物体的多个视角，以便重建出较为完整的模型. 本文采用多台 Kinect 设备从不同视角捕获场景并进行融合，用户不需要手持设备. 这种配置更适合于远程协作的应用环境，使用户可以解放双手进行更好的交流和协作.

1　总体系统设计

1.1　硬件系统设计

在远程协作的工作平台中，用户不便手持 Kinect设备进行工作，因此，本文系统使用 2～4 个固定摆放在场景周围不同方位的 Kinect 设备拍摄场景(如图 1 所示)，采用普通的个人计算机进行计算，得到场景的三维网格模型，且固定 Kinect 设备能简化重建动态场景的算法，也使重建结果更加稳定.

本系统在部署完成后需要进行一次标定操作，

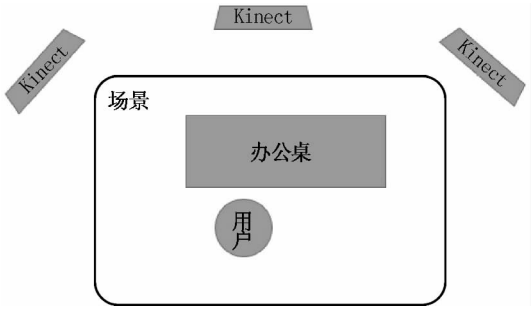


图 1　硬件系统布局示意图
Fig. 1　Hardware system layout

包括对所有 Kinect 设备分别进行内参标定，对所有 Kinect 设备统一进行外参标定. 内参和外参标定均采用棋盘格标定法.

1.2　算法流程概览

实时动态场景重建算法的主要流程为获取数据(深度数据和纹理色彩数据等)、对数据进行预处理、根据深度数据构建三维点云、对多组点云进行注册、从点云构建网格表面，具体如下所述.

(1) 对数据进行预处理. 从传感器获取的数据存在一定的噪声和误差分布特性，因此，需要对原始的深度数据进行特殊的预处理以降低噪声的干扰，从原始数据的层面减少误差.

(2) 根据深度数据构建三维点云. 对于每个传感器获得的深度数据和彩色图像数据，根据传感器的内参和外参信息计算点云中各点的三维坐标、颜色和法向量. 这时将它们可视化已经可以看出一定的重建效果，但是由于基于棋盘格的外参标定并不十分精确，从结构和纹理上可以看出多组点云并不能完全匹配.

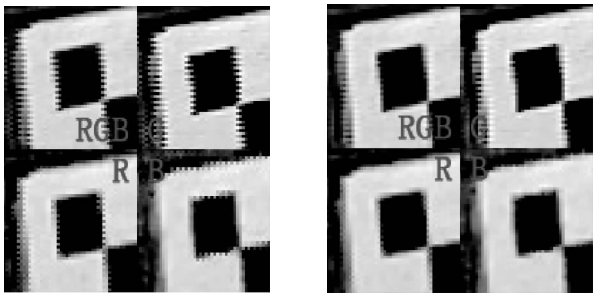
(3) 对多组点云进行配准. 旨在减小多组点云之间存在的变换偏差. 许多有关三维重建的研究(例如文献[5, 11])都使用迭代最近点配准算法(ICP)[14]，本文亦在该算法基础上提出了改进方案.

(4) 从点云构建网格表面. 从三维点云中构建较为完整的表面网格模型，具体涉及基于 GPU 的八叉树构建、有向距离函数计算和三角面片生成等算法.

2　彩色图和深度图的预处理

由于硬件层面的原因，Kinect 设备获得的彩色图呈现出明显的条纹状噪声，深度图中也存在较为明显的随机噪声，影响重建效果. 因此，在使用这些数据生成三维点云前，首先需要进行二维图像层面的降噪处理. 针对色彩图的条纹噪声特征，本文采用

反交错技术处理色彩图,基本消除了这一噪声,去噪效果如图 2 所示.

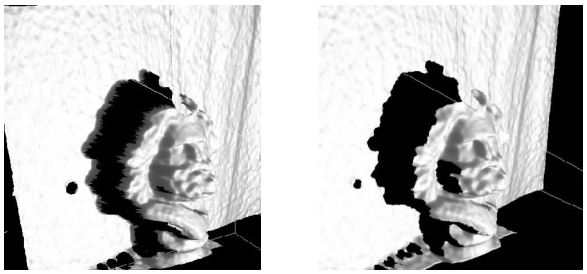


(a)预处理前 (b) 预处理后

图 2 彩色图预处理前后的效果对比

Fig. 2 Color images before and after preprocess

针对深度图中的随机噪声,如采用传统高斯滤波方法在降噪的同时会导致边缘模糊,因此,本文采用带边界检测的高斯滤波方法. 该方法能够在消除内部高频噪声的同时,保持边界的清晰. 经过这一处理的深度图所构建的三维模型不会在边缘部分产生毛刺,克服了传统高斯滤波方法的不足(如图 3 所示).



(a) 传统高斯滤波 (b) 带边界检测的高斯滤波

图 3 使用经过传统高斯滤波和带边界检测的高斯滤波处理的单深度图的重建结果对比

Fig. 3 The reconstruct result from single depth images processed by Gauss filter without and with edge detection

3 基于改进的 ICP 算法配准三维点云

在外参标定过程中的微小误差会被构建三维点云时的逆投影变换放大,因此,本文利用 ICP 算法实时地改进外参标定的精度. 为了增强算法的鲁棒性,本文采用位置-色彩混合空间下的距离定义[11].

另外,考虑到 GPU 的并行计算特性以及由深度图生成的点云具有高度的结构化,本文提出了基于投影关系的快速最近邻搜索算法来加速最近邻搜索. 该算法的主要依据是在深度图中排列的像素通过逆投影生成的三维顶点也具有一定的排列关系,即两个距离相近的顶点,其在深度图上的投影之间的距离也很接近(只要这两个点深度值不是很小). 因此,若假设某个顶点 v_1 的最近邻顶点是 v_2,v_2 的投影 p_2 有很大的概率落在 v_1 的投影 p_1 的小邻域内,则在计算 v_1 的最近邻时搜索范围可以被缩减到 p_1 的小邻域内.

4 从三维点云提取表面

场景重建的最后一个步骤是将点云变成表面模型. 离线重建一般采用泊松表面重建方法[15],但是该方法计算复杂度较高. 本文所实现的表面提取算法采用八叉树结构组织点云和划分空间,使用有向距离函数模型计算零值面,最后利用基于八叉树的 Marching Cubes 算法构建三角面片.

4.1 基于 GPU 的八叉树构建算法

本文参照文献[16]中所述设计并实现了基于 GPU 的八叉树构建算法. 为了使八叉树构建和查找能在 GPU 上高效执行,本文采用了广度优先的构建算法,并使用预先计算好的邻居查找表查找节点的邻居,将带有数据依赖而无法并行执行的操作降到最少.

4.2 基于有向距离函数的等值面计算

本文采用有向距离函数计算等值面,定义类似文献[17]中所使用的定义. 对于空间中的某点 P 和点云中的某点 V,有向距离被定义为向量 $\overrightarrow{VP}$ 在 V 的法向量上的投影. 而空间中某点 P 的有向距离函数值被定义为 P 到点云中所有与 P 邻近点的有向距离之和. 因此,有向距离函数的零值面可以近似看作表面模型.

4.3 Marching Cubes 表面提取算法

Marching Cubes 算法[18]是基于网格划分的表面提取算法,其基本思想是在每个网格中的小立方体单元中用平面结构近似表面. 根据立方体上的 8 个顶点在模型的内部或外部的情况可以将立方体分为 256 类,又由于内外对称、立方体旋转对称,这 256 个种类可以规约到 15 个基本种类[18](如图 4 所示). Marching Cubes 算法的基本流程对于网格中的每个单元,以 6 个顶点的正负值(表示在三维模型的外部或内部)组成索引查表获得该单元需要形成的一系列三角形顶点索引.

基于八叉树的 Marching Cubes 算法和传统的 Marching Cubes 算法相似,由于八叉树的叶子节点仅存在于有原始点云中的点的区域,因此,会有部分三角面片因节点的缺失而无法构建,导致最终生成的表面中包含许多小洞(如图 5 所示).

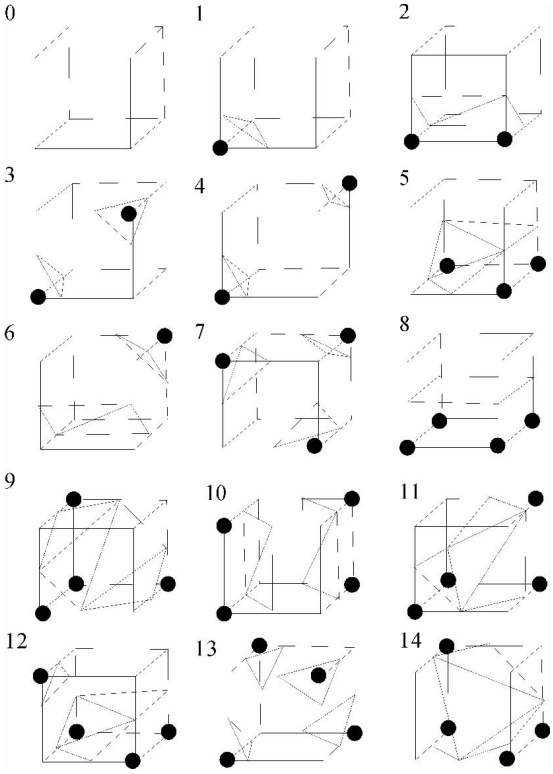


图 4　15 类立方体可以形成的三角面片[18]

Fig. 4　Fifteen kinds of triangle generation in a cube [18]

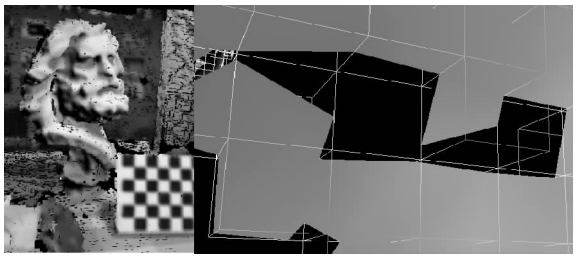


图 5　基于八叉树的 Marching Cubes 算法重建的表面

Fig. 5　Surface reconstructed from octree-based Marching Cubes algorithm

为了解决这一问题，本文提出了构造伪节点的方法，其基本思路是对于所有在面 f 上与表面存在交线的节点 n，构造一个伪节点 n_f，与 n 在 f 上邻接，并将在结构关系上属于 n_f 的顶点和边赋给 n_f. 然后再对所有伪节点执行 Marching Cubes 算法生成三角面片，便可补上这些空洞（如图 6 所示，图中顶点上的黑色标记表示该顶点为正值，反之为负值，下同）.

由伪节点的构造而引出的一个问题，即伪节点中不包含任何原始点云中的点，因此由伪节点而新增的顶点（可以称其为伪顶点）并不存在一个有向距离值（可以假定其为无穷大）. 一个无穷大的值

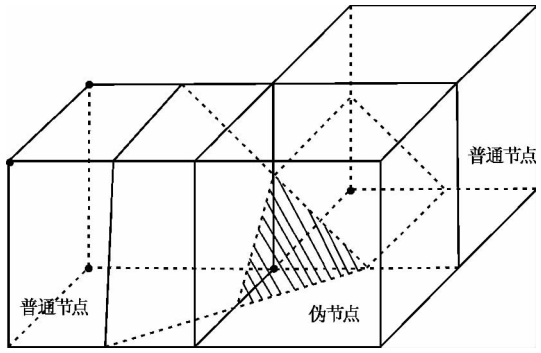
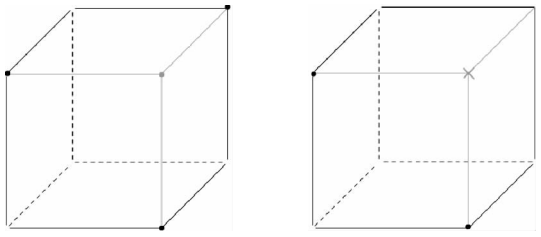


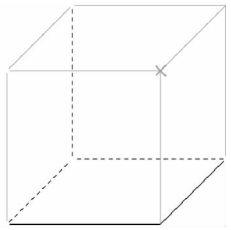
图 6　伪节点与在伪节点上构造的面片（图中画阴影的三角形）

Fig. 6　Fake node and the patch on it (the triangle marked with shadow)

并不属于正值或负值，这导致无法明确识别该立方体的类型. 因此，本算法采用就近原则，将一个无穷大值的符号定义为该顶点的 3 个邻接顶点中非无穷大值的符号（如图 7(a)）. 若 3 个邻接顶点中存在不同的符号（如图 7(b)）或 3 个邻接顶点均为伪顶点（如图 7 (c)），则抛弃该伪节点，不生成任何三角面片.



(a) 3 个相邻顶点符号相同　　(b) 3 个相邻顶点符号不同



(c) 3 个相邻顶点均为伪顶点

图 7　就近原则处理无穷大值的符号

Fig. 7　The sign of infinite decided by principle of proximity

而在生成三角面片方面，由于为伪顶点赋予了符号，Marching Cubes 算法生成的三角面片中可能会存在顶点在伪边上的情况. 对于这种情况，仅需要将该三角面片删除即可. 使用构造伪节点的方法后则重建结果中含有的空洞明显减少了（如图 8 所示）.

图8 用构造伪节点方法改进后的重建结果

Fig. 8 The improved result by using fake nodes

5 试验结果和性能

通过一系列试验验证本文所设计的场景重建算法的执行结果,测试和优化算法执行性能.所有试验均在同一台普通的家用型笔记本电脑上进行,其主要硬件参数为:Intel® Core™ i7-2670QM 处理器,8.00 GB 内存以及 AMD® Radeon™ HD 6770M 图形显示卡.所有试验均采用两台 Kinect 设备捕获场景,两台 Kinect 设备中心轴之间的夹角约为 45°,使用的深度图和彩色图分辨率均为 320×240.

试验场景为桌面物体及人物雕像,Kinect 设备与场景主要物件的距离范围为 60～100 cm.试验包括不同八叉树层级的重建结果比较、时间和空间测试的性能分析.

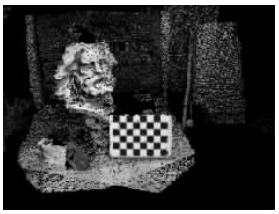
5.1 不同八叉树层级的重建结果比较

该试验目的是对比在不同的八叉树层级上执行表面提取的效果.用 8～11 层八叉树(分别对应叶节点边长为 23.4, 11.7, 5.9 和 2.9 mm)重建的结果如图 9 所示.从图 9 中可以看出,使用 8 和 9 层八叉树重建的结果十分粗略,尤其是使用 8 层时,雕像的五官几乎不可分辨.而使用 11 层八叉树重建结果虽然保留了非常细致的细节,但由于原采样点的密度不足,该重建结果也存在大量的空洞.使用 10 层八叉树重建的结果虽然细节较 11 层有所缺失,但在大部分区域都没有空洞,是最为适合当前所使用的采样分辨率的.因此,以下试验均采用 10 层八叉树重建.

(a) 8层八叉树

(b) 9层八叉树

(c) 10层八叉树



(d) 11层八叉树

图9 使用不同层数的八叉树重建场景的结果

Fig. 9 The reconstruct results using different levels of octree

5.2 时间性能分析和优化

优化前后主要步骤所用时间如表 1 所示.鉴于优化前算法在清空八叉树和提取表面花费的时间最多,优化主要针对这两个步骤进行.

表1 优化前后主要步骤所用时间

Table 1 Time spent in key steps before and after optimization

步骤	时间/ms		加速比
	优化前	优化后	
获取帧	31.69	2.18	14.54
清空八叉树	112.74	7.07	15.95
预处理	0.54	0.56	0.96
变换估算	26.43	18.52	1.43
建八叉树	65.40	55.58	1.18
提取表面	106.74	28.90	3.69
总体	345.15	114.36	3.02

清空八叉树的时间耗费过长主要是由于内存到显存的带宽相对较窄,采用从内存中将初始数据重新导入 GPU 缓存中的开销巨大.因此,针对这一步骤的优化采取了通过执行 shader 为缓存写入初始数据的方法.

对于提取表面步骤,首先进一步分析了表面提取算法中的各个步骤的时间开销.优化前后表面提取算法的各步骤时间开销对比如图 10 所示.由图 10 可知,计算每个八叉树顶点的有向距离花费了绝大多数的时间,而优化该步骤的代码后表面提取算法的时间开销明显缩短.

```
##################################
  All: 100.025ms
##################################
CalcValue: 67.8943ms
GenVertices: 7.38622ms
GenTriangles: 1.73522ms
GenFakes: 14.261ms
SetFakesVerticesEdges: 2.05489ms
GenFakeTriangles: 2.87133ms
```

(a) 优化前

```
##################################
# All: 33.2514ms
##################################
CalcValue: 2.949ms
GenVertices: 7.38411ms
GenTriangles: 1.71456ms
GenFakes: 13.3897ms
SetFakesVerticesEdges: 2.07211ms
GenFakeTriangles: 2.84956ms
```

(b) 优化后

图 10　优化前后表面提取算法的各步骤时间开销对比

Fig. 10　Time spent in each step of surface extraction before and after optimization

由表 1 可知，优化后重建算法执行时间是原来的 33%，加速比达到 3.02. 优化后的运行帧率可达到 8.74 f/s.

5.3　GPU 内存空间性能分析与优化

场景重建算法所占用的 GPU 内存空间如表 2 所示，所列数据为分析具体实现后计算而得.

表 2　优化前后主要数据结构空间的使用效率

Table 2　The space usage of main data structures before and after optimization

数据结构	优化前			优化后		
	有效空间/MB	占有空间/MB	有效率/%	有效空间/MB	占有空间/MB	有效率/%
模型顶点	3.48	70.31	4.94	3.48	17.23	20.19
面索引	1.78	8.79	20.20	1.63	4.31	37.93
节点	18.75	103.71	18.08	18.85	33.88	55.65
叶子节点	12.59	34.57	36.43	12.66	16.94	74.72
节点边	10.86	56.25	19.31	10.90	13.78	79.08
节点顶点	7.06	56.25	12.55	7.08	8.27	85.65
总计	54.52	329.88	16.53	54.60	94.41	57.83

由表 2 可知，优化前执行场景重建算法需要 329.88 MB 的显存空间，有效率仅有 16.53%. 主要原因是动态分配 GPU 缓存资源的开销极大，一般都是在初始化时就创建了足够大的空间. 例如基于极端情况下每个八叉树的叶子节点可能只包含一个点云中的点，因此，优化前为八叉树叶子节点分配的空间与分辨率和 Kinect 的个数的乘积一致. 但是事实上由于点集分布的集中性，上述极端情况是不可能出现的. 另外由于分辨率的增加只会导致点云的密度增加，并不会增加许多覆盖区域，因此，实际上分辨率的增加并不会导致叶子节点的大幅增加. 经过对试验数据的分析后，本文进行了 3 步优化：

(1) 对点云数据进行压紧处理，使得有效数据都集中在数组前端；

(2) 将为节点分配的空间大小与分辨率的关系从线性关系修正为与输入分辨率的二次根呈线性关系，可以有效减少使用高分辨率输入数据时的空间浪费；

(3) 将节点边和节点顶点改为紧凑排列，并减少分配给节点边和节点顶点的空间.

经过 3 步优化，算法所耗费的 GPU 内存空间仅为优化前的 1/3，有效率提升至 57.83%.

5.4　办公场景的重建

场景包含人物和放置着一些办公物品的桌面. 使用两台 Kinect 设备拍摄场景，其中心轴之间的夹角约为 30°，与场景中主要物体的距离范围为 80～150 cm. 重建结果如图 11.

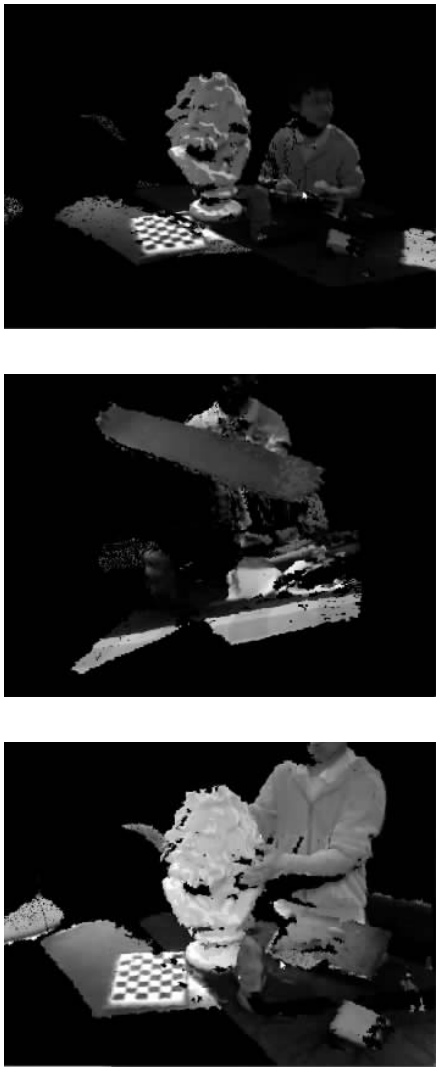


图 11　办公场景的重建结果

Fig. 11　Reconstruction of work scene

6　结　语

本文设计并实现了基于多个 Kinect 设备的实时动态场景重建技术，性能基本满足实时要求. 主要

研究成果包括:提出了基于投影关系的快速最近邻搜索算法;用构造伪节点法改进了基于八叉树的Marching Cubes算法的重建结果;实现了完整的场景重建算法,并利用GPU并行技术达到了三维重建实时性.

未来工作将主要集中于以下几点:深入研究更多应用在表面重建中的数学模型,以期提出一个更好的数学模型减少空洞和噪声敏感度,并提升重建模型的细节;引入纹理色彩分析算法,考虑研究色调补偿和表面材质提取等方面的内容,并改进模型纹理的生成方法,提高模型纹理的真实感;进一步优化性能.

参 考 文 献

[1] DENG S W, YUAN B Z. A method for 3D surface reconstruction from range images[C]// 1994 International Symposium on Speech, Image Processing and Neural Networks. 1994:429-432.

[2] CURLESS B, LEVOY M. A volumetric method for building complex models from range images[C]// Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, ACM. 1996: 303-312.

[3] SCHUTZ C, JOST T, HUGLI H. Free-form 3D object reconstruction from range images[C]// 1997 International Conference on Virtual Systems and MultiMedia. 1997: 69-70.

[4] WHITAKER R T. A level-set approach to 3D reconstruction from range data[J]. Int J Comput Vision, 1998, 29(3): 203-231.

[5] IZADI S, NEWCOMBE R, KIM D, et al. Kinect Fusion: Real-time dynamic 3D surface reconstruction and interaction [C]//ACM SIGGRAPH 2011 Talks (SIGGRAPH' 11). Vancouver, British Columbia, Canada: ACM, 2011.

[6] 杨敏,沈春林.基于场景几何约束未标定两视图的三维模型重建[J].中国图象图形学报:A版, 2003, 8(8): 872-876.

[7] REMONDINO F, EL-HAKIM S F, GRUEN A, et al. Turning images into 3-D models[J]. Signal Processing Magazine, IEEE, 2008, 25(4): 55-65.

[8] BESL P J, JAIN R C. Three-dimensional object recognition [J]. ACM Computing Surveys (CSUR), 1985, 17(1): 75-145.

[9] 刘田间,郭连朋,陈向宁,等.基于Kinect传感器多深度图像融合的物体三维重建[J].应用光学, 2014, 35(5):811-816.

[10] 应忍冬,陈晓明,蒋乐天.基于Kinect深度信息的实时三维重建和滤波算法研究[J].计算机应用研究, 2013, 30(4):1216-1218.

[11] NEUMANN D, LUGAUER F, BAUER S, et al. Real-time RGB-D mapping and 3-D modeling on the GPU using the random ball cover data structure[C]// 2011 IEEE International Conference on Computer Vision Workshops, 2011: 1161-1167.

[12] KELLER M, LEFLOCH D, LAMBERS M, et al. Real-time 3D reconstruction in dynamic scenes using point-based fusion [C]//2013 International Conference on 3D Vision. IEEE, 2013: 1-8.

[13] ZOLLHÖFER M, NIEßNER M, IZADI S, et al. Real-time non-rigid reconstruction using an RGB-D camera[J]. ACM Transactions on Graphics (TOG), 2014, 33(4):

[14] BESL P J, MCKAY N D. A method for registration of 3-D shapes[J]. IEEE Trans Pattern Anal Mach Intell, 1992, 14(2): 239-256.

[15] KAZHDAN M, BOLITHO M, HOPPE H. Poisson surface reconstruction[C]//Proceedings of the Fourth Eurographics Symposium on Geometry Processing. Cagliari, Sardinia, Italy: Eurographics Association, 2006: 61-70.

[16] ZHOU K, GONG M, HUANG X, et al. Highly parallel surface reconstruction[J]. Microsoft Research Asia, 2008.

[17] HOPPE H, DEROSE T, DUCHAMP T, et al. Surface reconstruction from unorganized points [C]//Computer Graphics (SIGGRAPH '92 Proceedings). 1992: 71-78.

[18] LORENSEN W E, CLINE H E. Marching cubes: A high resolution 3D surface construction algorithm[C]//Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '87). 1987: 163-169.