

基于深度图像的三维场景重建系统

张志林, 苗兰芳

(浙江师范大学数理与信息工程学院, 浙江 金华 321004)

摘 要: 针对计算机图形学和视觉领域研究热点——三维场景重建, 首先分析了 Kinect v2 (Kinect for Windows v2 sensor) 获取深度图像的原理, 说明深度图像噪声的来源。然后根据获取深度图像的原理设计一种算法对点云采样范围进行裁剪。其次对点云离群点进行去除, 填补点云孔洞, 以提高重建质量。常见的三维场景重建大都采用了 KinectFusion 的一个全局立方体方案, 但只能对小范围内的场景进行重建。对此设计了一种对大场景进行点云匹配的 ICP 算法。最后对点云进行曲面重建, 实现一套低成本、精确的针对大场景的三维重建系统。

关 键 词: Kinect v2; 三维场景重建; 点云去噪; 离群点去除; ICP

中图分类号: TP 391

DOI: 10.11996/JGJ.2095-302X.2018061123

文献标识码: A

文章编号: 2095-302X(2018)06-1123-07

3D Scene Reconstruction System Based on Depth Image

ZHANG Zhilin, MIAO Lanfang

(College of Mathematics, Physics and Information Engineering, Zhejiang Normal University, Jinhua Zhejiang 321004, China)

Abstract: The reconstruction of three-dimensional scenes is currently a research hotspot in computer graphics and visual fields. Firstly, this paper analyzes the principle of depth image acquisition by Kinect v2 (Kinect for Windows v2 sensor), and explain the source of depth image noise. Then according to the principle of obtaining the depth image, an algorithm is designed to crop the point cloud sampling range. Secondly, the point cloud outliers are removed to fill the voids of the cloud to improve the reconstruction quality. Common three-dimensional scene reconstruction mostly uses a global three-dimensional body scheme of Kinect Fusion. This scheme can only reconstruct scenes in a small area. An ICP algorithm for point cloud matching of large scenes is designed for this purpose. Finally, surface reconstruction is performed on the point cloud to realize a low-cost and accurate three-dimensional reconstruction system for large scenes.

Keywords: Kinect v2; 3D scene reconstruction; point cloud denoising; outlier removal; ICP

根据相机的移动, 对变换中的三维场景进行重建, 一直是计算机视觉和图形学领域的热点和难点。基于 KinectFusion 方案的三维场景重建, 虽然可以实时, 准确的进行场景重建。但是由于受到全局立方体的限制, 只能进行小规模场景重建。Kinect v2 采用不同于 v1 结构光的 TOF(time-of-flight)深度

测量方法。虽然测出来的深度数据精度不能达到工业制造水平, 但是在无人驾驶和机器人定位导航、文物三维取样保存、3D 游戏和影视建模等领域中有广泛的应用。

基于 Kinect v1 的三维重建有许多优秀的解决方案, 其中微软研究院 NEWCOMBE 等^[1]利用

收稿日期: 2018-04-15; 定稿日期: 2018-06-21

基金项目: 国家自然科学基金项目(61170315)

第一作者: 张志林(1992-), 男, 安徽合肥人, 硕士研究生。主要研究方向为计算机图形学。E-mail: jsjzhangzhilin@163.com

通信作者: 苗兰芳(1963-), 女, 浙江慈溪人, 教授, 博士。主要研究方向为计算机图形学。E-mail: mlf@zjnu.cn

Kinect v1 提出一种可以在不受光照条件下的室内场景的实时三维重建技术。NIEBNER 等^[2]将全局立方体所占用的物体用 Hash 函数进行索引,充分利用了全局立方体中的空间,在同等的显示内存中增加了重建范围,但不能做到既增大范围,又提高精度。WHELAN 等^[3]提出 Kintinuous 的一种空间扩展模型,设置一个平移阈值,全局立方体随着 Kinect v1 移动而移动,并且可以将移除的立方体数据导入到内存中。ZENG 等^[4]使用八叉树(Octree)数据结构来节省内存使用,通过对全局立方体进行八等分,直到找到最小并且合适的空节点来存储场景中不空的部分。由于受到全局立方体的限制,其改进非常有限。HENRY 等^[5]使用彩色和深度信息之间的映射,实现了带有彩色纹理的三维场景重建,但是重建效果比较粗糙。谭歆^[6]在 KinectFusion 的基础上,提出一种保留设备精度,可应用于大场景离线的三维物体重建的解决方案,并且可以实现全局优化,但是不能做到全自动闭环。周瑾等^[7]提出使用 ICP 算法输入点云与可变模型之间的对应点对,用可变形的人体模型来拟合采样的点云,通过迭代获取高精度的三维人体模型。朱笑笑等^[8]提出使用环境中边线特征点匹配来提高其定位的鲁棒性,与此同时在点云模型中预设一个地面点云来降低累积误差提高精度。李诗锐等^[9]利用光滑表面作为物理表面平滑程度的先验知识,修正后面数据流的噪声,解决了累积误差问题,并且基于 Kinect v2 实现了一套实时、精确地重建物体,可以广泛应用的低成本快速三维重建系统。但此方法建立在 KinectFusion 的基础上,并不适合大场景的三维重建。胡正乙等^[10]提出一种带有深度约束和局部近邻约束的基于 RGB-D 室内场景实时三维重建算法,增加了特征点对的匹配准确率。

本文在构建基于 Kinect v2 三维场景重建系统时,摒弃了传统的 KinectFusion 算法框架。因此实时性并不强,适合做离线的点云处理。本系统的贡献主要有:①利用 Kinect v2 SDK 的 API,构建了场景重建的点云获取模块。②分析 Kinect v2 的噪声类型和来源,并且根据所采集点云的分布特点,设计一种包围盒对点云进行裁剪,获得了高质量的点云。③将两两点云配准的 ICP 算法进行改进,利用队列数据结构,对多片点云进行配准,配准之后场景大小不受全局立方体的限制,适合大场景的三维重建。④实现了一套可以用于离线的大场景的三维重建系统。

1 Kinect v2 与 Kinect v1 获取深度图像原理

Kinect v1 的深度传感器,采用了 Light Coding^[11]的方式,将红外线光均匀分布投射到测量空间中,通过红外线摄影机记录空间中的每个散斑,获取原始资料后,再通过芯片计算获得具有深度信息的图像。Kinect v2 使用 TOF^[12]技术,其原理是通过对目标场景发射连续的近红外脉冲,然后用传感器接收由物体反射回的光脉冲,比较发射光脉冲之间的传输延迟计算物体相对于发射器的距离,从而得到一幅深度图像。TOF 技术比 Light Coding 技术在深度图像上拥有 3 倍的精确度。其数据更加稳定,细节更多,被其他光源(红外线)影响的概率也更低,捕捉到的红外线影像不会有散斑。表 1 是两代 Kinect 技术参数的比较。

表 1 Kinect v1 与 Kinect v2 的技术参数对比

项目	Kinect v1	Kinect v2
视角	水平 57.5°垂直 43.5°	水平 70°垂直 60°
深度有效值	0.5~5.0 m (近场 0.35 m)	0.5~5.0 m
彩色相机分辨率	1280×960	1920×1080
深度相机分辨率	640×480	512×424
红外相机分辨率	1280×960	512×424

Kinect v2 使用 ToF 深度摄像机获取深度图像,受到曝光时间、被测目标的材质颜色,甚至是温度等因素的影响,导致其深度图像存在许多噪声点。图 1 可以看出在相同距离和曝光时间下, Kinect v2 比 v1 采集了更多的细节,但在物体的边缘处的深度值出现误差,其原因通常是由于一个像素点对应的场景涵盖了不同的物体表面所引起的。

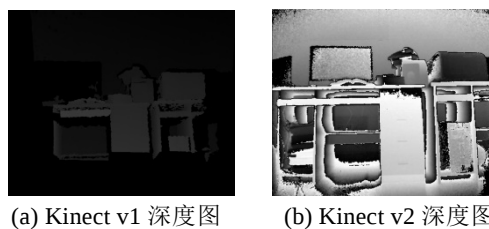
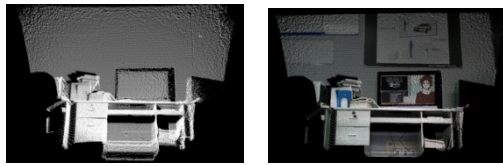


图 1 两代 Kinect 深度图比较

2 Kinect v2 获取点云

在使用 Kinect v2 设备获取深度图像之前,首先要对其进行标定。目前最为成熟的是 ZHANG 标定法^[13],被广泛应用于相机标定中。文献[14]通过对比标定前和标定后的深度图像发现,误差很小,说明 Kinect v2 内部芯片已经对彩色摄像头和红外摄像头进行了标定。本文采用的是 Kinect for Windows

SDK v2.0 内部中 `MapDepthPoint ToCameraSpace()` 函数, 直接将深度图像转换为三维点云, 故没有对 Kinect v2 进行额外的标定。图 2 是获取的点云图像, 图 2(a)是具有三维坐标 XZY 的点云, 图 2(b)是具有颜色值的 $XYZ-RGB$ 的点云。从图 2 中可以看出电脑屏幕和书本遮挡的部分以阴影的形式表现出来, 将获取的点云旋转和平移到和 Kinect v2 相对应的视角时, 可以在视野中最大限度地消除阴影部分。本文使用普通点云进行重建, 原因是彩色点云在进行曲面重建时, 颜色部分会被截断, 最后和灰度点云得到相同的结果, 但彩色点云更消耗存储空间。



(a) 普通点云 (b) 包含颜色信息的点云

图 2 Kinect v2 获取的点云

3 点云处理

在获取点云数据时, 由于设备精度、操作者经验、环境因素等带来的影响, 以及电磁波衍射特性、被测物体表面性质变化和拼接配准操作过程的影响, 点云数据中将不可避免地出现一些噪声点。实际应用中除了测量随机误差产生的噪声点之外, 由于受到外界干扰(如视线遮挡、障碍物等), 点云数据中往往存在着一些离主体点云即被测物体点云较远的离散点, 即离群点。同时场景点云数量庞杂, 需要对点云进行向下采样精简点云, 加快速度。本文将以此对采样的点云进行处理, 优化系统重建效果。

3.1 点云裁剪

从图 2 中可以看出, 物体边缘位置出现跳边的现象, Kinect v2 有效深度值为 0.5~5.0 m, 在 Kinect v2 视角的边缘位置出现了大量的噪声点和凸起的情况, 采样效果较差。点到平面距离的标准方差 $x(\text{mm})$ 与传感器到平面的距离 $y(\text{m})$ 成一种线性关系^[9], 即

$$y = 0.5472x + 1.258 \quad (1)$$

表明 Kinect v2 在深度有效值范围内, 随着距离的增大, 在采样场景的边缘处残差也增大。本文设计一种算法将点云采样范围设置在最佳的区域内, 选取质量好的点云进行场景重建。假设点云集合中点云坐标为 (x, y, z) , 其中 x, y 为点云所在深度图中的坐标; z 为点云的深度值; Kinect v2 与所构建场景的距离为 d 。从表 1 中可以得出 Kinect v2 的视角为

水平 70° , 垂直 60° , 则 x, y, z 的取值范围为

$$\begin{aligned} -d \times \tan 70^\circ &\leq x \leq d \times \tan 70^\circ \\ -d \times \tan 60^\circ &\leq y \leq d \times \tan 60^\circ \\ 0.5 &\leq z \leq 5.0 \end{aligned} \quad (2)$$

Kinect v2 最小距离可以为 0.5 m, 但该距离会由于没有深度数据而产生较大的空洞。由上文可知距离越近采样的点云越精确, 故本文将 Kinect v2 的距离场景(即最近的桌子边缘)的长度设置为 1.0 m。将点云的 x, y 坐标范围裁剪为整个场景中一半即中心位置。原因是 TOF 相机在发射红外光时在边缘位置反射回来的光振幅较小^[15]和深度相机畸变导致场景边缘产生较大的残差。因此本文设置场景的三维包围盒范围为

$$\begin{aligned} -0.5 \times d \tan 70^\circ &\leq x \leq 0.5 \times d \tan 70^\circ \\ -0.5 \times d \tan 60^\circ &\leq y \leq 0.5 \times d \tan 60^\circ \\ 1.0 &\leq z \leq 5.0 \end{aligned} \quad (3)$$

对点云进行裁剪后的效果如图 3 所示, 其中蓝色为裁剪去除的点云, 绿色为保留下来的点云, 从图 3 中可以看出, 该算法有效地去除了一些畸变的边缘噪声和大面积的空洞。

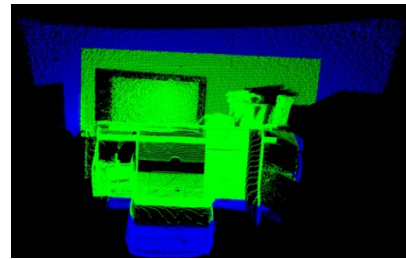


图 3 点云裁剪

3.2 点云离群点去除

本文使用 PCL1.80 版本点云库中的 `Statistical OutlierConditionalRemoval` 滤波器来移除离群点。其主要的思想是对每个点的 K 邻域进行一个统计分析, 首先计算这个点到其 K 邻域的平均距离。假设得到的结果是一个高斯分布, 其形状由均值和标准差决定, 平均距离在标准范围之外的点, 可被定义为离群点, 并可从数据集中移除。如图 4 所示, 通过调节标准范围的大小, 确定一个最佳的采样效果, 绿色部分为保留的点云, 蓝色部分是被移除的离群点, 表 2 给出了具体的分析, 本文选择图 4(d) 的标准范围。

3.3 点云重采样

本文使用 `VoxelGrid` 的方法, 即根据给定的点云构造一个三维体素栅格并进行下采样达到滤波效果。`VoxelGrid` 通过输入的点云数据创建一个三维

体素栅格,然后将每个体素内所有的点都用该体素内的点集的重心来近似,大大减少了数据量,减少点云在配准、曲面重建时所消耗的时间,提高速度。本文通过大量实验得出使用 $0.4\text{ cm} \times 0.4\text{ cm} \times 0.4\text{ cm}$ 的立方体来过滤点云,可以将孔洞数量减小到较小的范围,且点云数量保留较少,其点云数量减少到原来的 60%,由于篇幅原因不做展开。

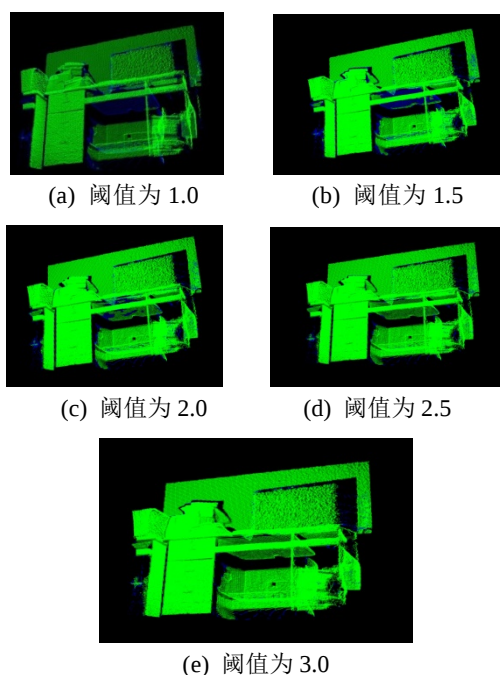


图 4 点云离群点去除

表 2 阈值标准的调节

邻域 K	阈值	保留点数	图示	分析
50	1.0	135 924	图 4(a)	桌面没有保存下来
50	1.5	141 848	图 4(b)	桌面大面积残缺
50	2.0	146 875	图 4(c)	桌面小面积残缺
50	2.5	149 657	图 4(d)	桌面保存完整
50	3.0	150 763	图 4(e)	两桌之间离群点没有去除

4 点云配准

在本文中点云配准分为 3 步:①对两片点云进行粗配准。首先是对两片点云按照相同的关键点选取标准,提取关键点。然后对选择的所有关键点分别计算其特征描述子,最后结合特征描述子在两个数据集集中的坐标位置,以二者之间特征和位置的相似度为基础,来估算其对应关系,完成粗配准。②利用经典的 ICP 算法对两片点云进行细配准。③设计一种利用队列来进行多对点云配准算法。

4.1 ICP 算法简介

ICP 算法(iterative closest point)^[16]的基本思想

是给定两个来自不同坐标系的三维数据点集,找出两个点集之间的空间变换,以便能进行空间匹配。迭代最近点算法的基本思想是:

已知两个对应点集合

$$X = \{x_1, x_2, \dots, x_n\} \quad (4)$$

$$P = \{p_1, p_2, \dots, p_n\} \quad (5)$$

求解旋转矩阵 R 和平移向量 t , 目标函数为

$$E(R, t) = \arg \min_{R, t} \frac{1}{N_p} \sum_{i=1}^{N_p} \|x_i - R p_i - t\|^2 \quad (6)$$

算法的一般流程如下:

步骤 1. 在点集 P 中寻找与点集 X 中的对应点

$$y = \{y_1, y_2, \dots, y_n\} \quad (7)$$

步骤 2. 求解旋转矩阵 R 和平移向量 t , 目标函数为

$$E(R, t) = \arg \min_{R, t} \frac{1}{N_p} \sum_{i=1}^{N_p} \|y_i - R x_i - t\|^2$$

步骤 3. 更新点云模型

$$y^{k+1} = \{y_i^{k+1} | y_i^k = R^k x_i^k + t^k, x_i^k \in X\} \quad (8)$$

步骤 4. 计算对应点集间的距离

$$d^{k+1} = \sum_{i=1}^n \|y_i^{k+1} - p_i^k\| / n \quad (9)$$

步骤 5. 若 $d^{k+1} < l$ (l 为给定阈值)则算法结束, 否则返回步骤 1。

4.2 多对点云配准

ICP 算法是对点云进行两两配准, 利用 Kinect v2 可以获取多方位的多片点云, 本文设计一套对多片点云进行配准的算法。考虑到 Kinect v2 在获取点云时, 各片点云随时间的变化, 相邻点云具有大量重叠的部分, 满足 ICP 算法的要求。由于获取的相邻点云具有时序性, 因此, 提出使用队列这种先进先出的线性表来维持场景规模。当点云拼接的规模达到设定的阈值即场景点云拼接的最大值时, 则将场景点云集合 Sum 减去队列中的队尾点云, 然后再对当前点云与前一帧的点云进行 ICP 配准, 最后再拼接到场景点云集合 Sum 中。具体流程如图 5 所示。

本算法通过可变长的队列来维持和改变场景的大小, 实现大场景的三维重建。为了加快配准速度, 本文采用的是当前点云和前一帧的点云进行配准, 而没有让当前帧和当前的场景点云 Sum 进行一个全局的配准。当然也会出现一定的累积误差。

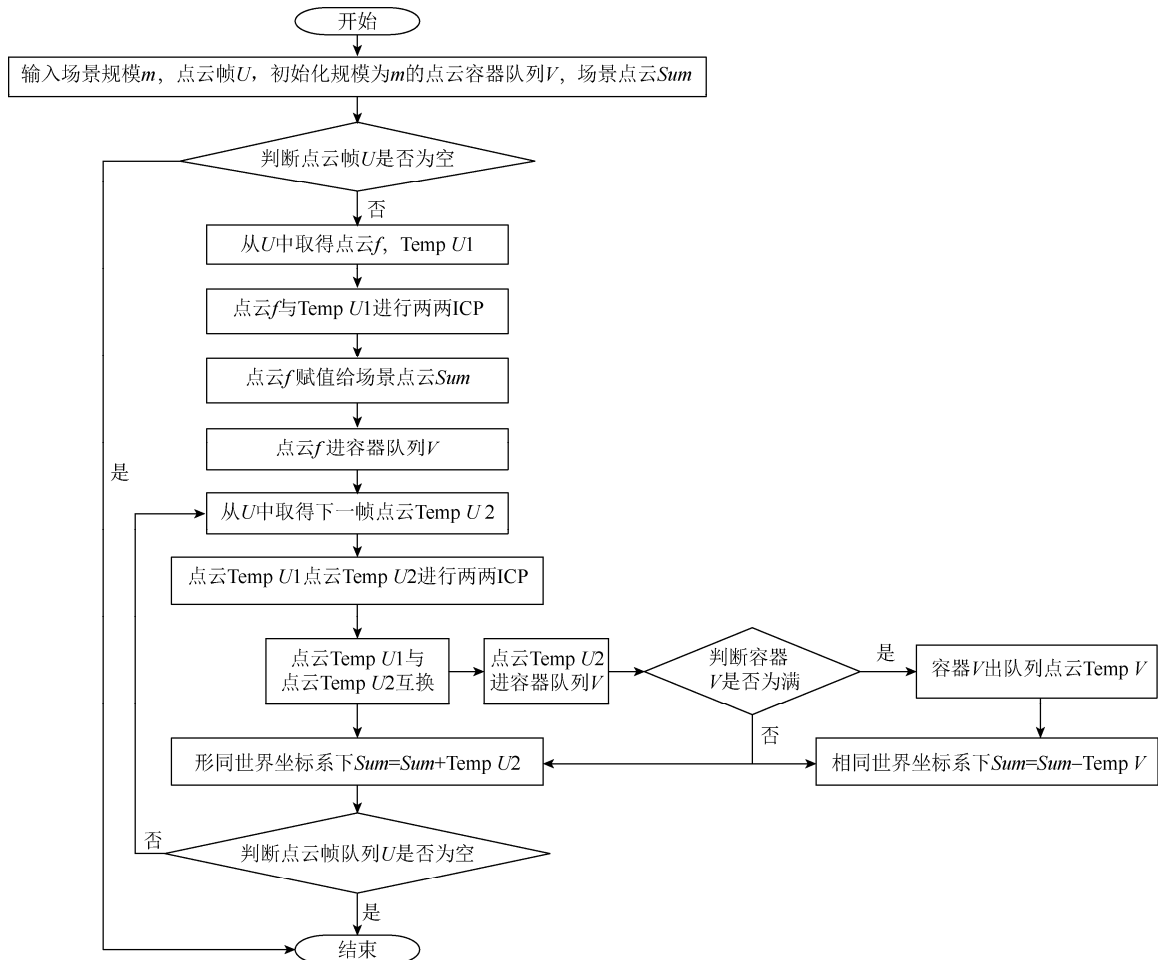


图5 多片点云配准算法流程图

5 Kinect v2 大场景的重建系统

Kinect v2 对平台的要求比较高, Windows8/Windows8.1 操作系统, Visual Studio 2012 以上软件开发平台, 显卡支持 DX11, 内置 USB 3.0, 4 G 以上内存, 64 位的 CPU, i7, 2.5 GHz 以上更佳。本文系统构建的环境是, Windows8.1/Visual Studio 2013/AMD RADEON R9 M290X/USB3.0/8G/i5-4690 3.5 GHz

图6是整个场景重建系统的算法框图, 本系统使用 C++ 程序开发语言, 利用 Kinect for Windows SDK 2.0 提供的接口获取数据和进行相机标定, 以及 PCL 1.8.0 版本的点云库对点云数据进行处理, 都需要在 Visual Studio 2013 的开发环境下进行编译使用。具体步骤为:

步骤 1. 获取深度图像, 本文通过 Kinect v2 获取深度相机中的每一帧, 获取图像信息。

步骤 2. 深度图像转换为点云, 本文并没有对 Kinect v2 进行额外的标定, 而是直接通过 Kinect for Windows SDK 2.0 获取相机内置标定参数进行标

定, 转换为点云数据。实验结果表明获得了较好的采样效果。

步骤 3. 点云去噪, 通过分析点云噪声分布规律, 实现一种类似包围盒的去噪算法。

步骤 4. 通过调整滤波阈值的参数来进行调优, 实现点云离群点的去除和向下采样。

步骤 5. 将两两配准的 ICP 算法进行改进, 使其可以进行多片点云配准, 并且可以通过设置点云场景参数, 来动态控制场景的大小。使构建场景不受 KinectFusion 全局立方体体积的约束。

步骤 6. 对点云进行贪婪投影三角化实现曲面的重建。本文利用 PCL 点云库实现了该算法。

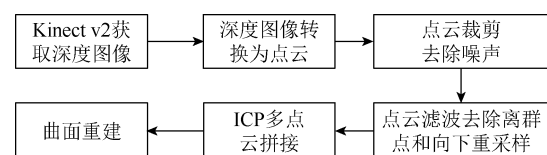


图6 基于 Kinect v2 的重建系统算法框图

5.1 实验结果

本文通过获取关键帧的形式对场景进行重建, 本文场景是一个长 1.2 m、宽 0.6 m、高 0.75 m 的桌

子。将 Kinect v2 放在一个高 1.2 m, 距离桌子边距为 1.0 m 的载物台上。ICP 算法对两片点云的初始值有很高的要求, 否则难以得到闭合解, 导致 ICP 算法失效。将 Kinect v2 从右至左每次平移 0.03 m, 来获取 6 片关键帧的点云, 进行场景重建。ICP 配准后的效果如图 7 所示, 配准后点云的个数见表 3。

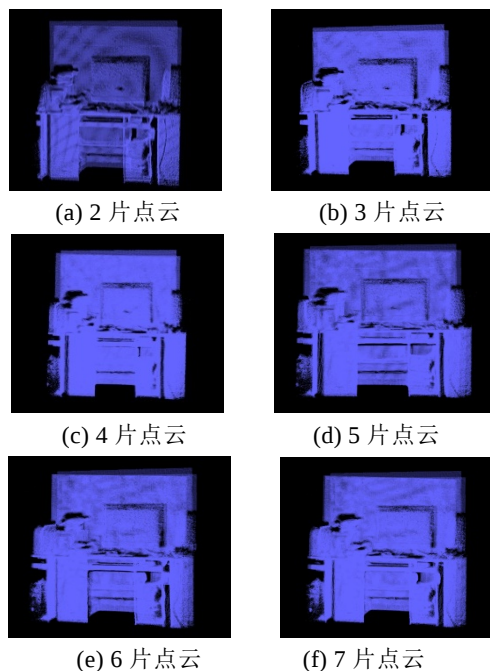


图 7 配准的点云图

表 3 配准后点云个数一览表

图示	点云片数	配准后点数
图 7(a)	2	369 930
图 7(b)	3	493 507
图 7(c)	4	616 455
图 7(d)	5	740 153
图 7(e)	6	863 610
图 7(f)	7	986 836

本文使用贪婪投影三角化算法对配准后的点云进行曲面重建, 在重建时通过扩大连接点之间的最大距离来确定 K 近邻的球半径, 使更多的点连接起来, 适当减少孔洞。KinectFusion 和本实验结果对比如图 8 所示。

5.2 结果分析

从图 7 可看出随着 Kinect v2 的移动, 左边放置打印机的桌子开始逐渐显现出来。同时, 随着维持点云队列长度的增加, 桌上的细节开始逐渐地模糊起来, 与此同时, 点云的场景也开始增大。除了增大点云队列的长度, 还可以通过以下方式来增大场景: ①通过在不影响 ICP 算法有效性的前提下, 提高间距, 获得更大的场景。②本文所裁剪的场景是 Kinect v2 当前视角下的一半, 可以在避免获得

噪声点云的情况下, 扩大裁剪场景的包围盒来扩大重建场景。③可以通过直接拉大采样距离来获得更大的场景。本文的采样距离是 1.0 m, 只是 Kinect v2 获取场景有效深度的 20%。通过裁剪获得一个和 KinectFusion 极限范围同样大小的场景。综上所述, 本文算法在获得重建场景范围上, 具有相当大的优势。

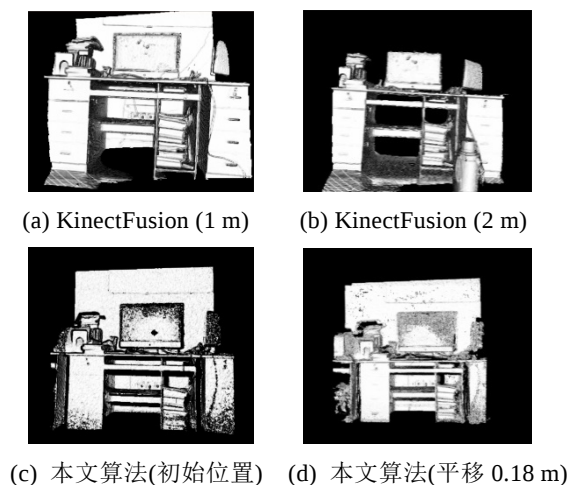
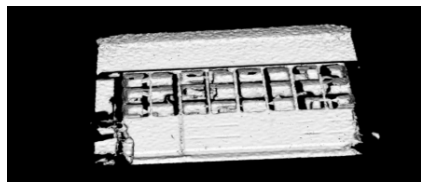


图 8 KinectFusion 与本文算法重建效果对比

图 8(a)、(b)是 Kinect for Windows SDK v2.0 中集成的 KinectFusion 算法所构建的场景, 可看出全局立方体的限制, 图 8(a)是距离场景 1 m 时所构建的, 图 8(b)是距离场景 2 m 时所构建的, 超出重建范围的场景在屏幕中都是不可见的, 在可见的范围内, 可以通过拉近 Kinect v2 与场景的距离不断刻画场景中的细节, 并且全局立方体适合做 GPU 的并行计算, 为 KinectFusion 进行实时场景重建提供了有利条件。图 8(c)是初始位置时所获得的场景, 图 8(d)水平向左平移 0.18 m 六片点云融合后所获得的场景。由于 Kinect v2 所采集的点云在整个场景中密度分布不均匀, 所采用的贪婪投影三角化算法对场景点云分布密度要求很高, 因此在场景中出现了很多孔洞。图 8(d)的细小孔洞明显比图 8(c)多, 如电脑后面的墙体。这是由于多片点云在进行拼接时, ICP 算法没有得到完美的闭合解, 点云叠加时破坏了原来曲面的光滑度, 同时也改变了点云均匀的变化密度, 再加上贪婪投影三角化算法不能在三角化的同时对曲面进行平滑和孔洞修复, 使得曲面重建的效果不是特别理想。图 9(a)是实验室中的一面墙的书橱重建效果图重建有 8 m²左右的面积。图 9(b)是实验室一角, 图 9(c)是实验室用于呈现虚拟现实场景的展示台, 中间放一把椅子来判断配准的

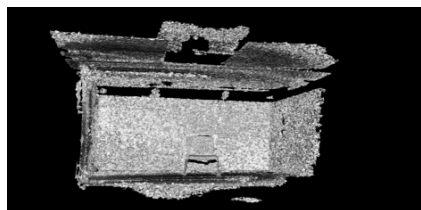
效果, 由于 kinect v2 的远距离取样精度的问题, 效果比较粗糙。图 9(d)~(f)是由 15 片点云从不同角度重建出来的椅子效果图。



(a) 实验室墙面



(b) 实验室一角



(c) 展示台



(d) 角度 1 (e) 角度 2 (f) 角度 3

图 9 大面积场景和单个物体的重建

6 总结与展望

通过分析, Kinect v2 获得的深度图像噪声主要集中在离深度相机较远的边缘地区。本文摒弃了 KinectFusion 三维重建的算法框架, 提出一种新的适合大场景的三维重建方案。着重去除了点云场景中边缘的噪声点, 提高了采样的质量。然后将两两配准的 ICP 算法, 扩展到多片点云配准, 设计一个队列来维持场景中点云的个数, 以此来控制场景的大小, 获得了较好的效果。下一步的工作将重点放在解决 Kinect v2 在重建大场景的实时性和修复面孔洞的问题上。

参考文献

[1] NEWCOMBE R A, IZADI S, HILLIGES O, et al.

KinectFusion: real-time dense surface mapping and tracking [C]//2011 10th IEEE International Symposium on Mixed and Augmented Reality (ISMAR). New York: IEEE Press, 2011: 127-136.

- [2] NIEBNER M, ZOLLHÖFER M, IZADI S, et al. Real-time 3D reconstruction at scale using voxel hashing [J]. ACM Transactions on Graphics (TOG), 2013, 32(6): 1-11.
- [3] WHELAN T, MCDONACD J, KAESSE M, et al. Kintinuuous: spatially extended kinectfusion [EP/OL]. [2018-03-08]. <http://www.thomaswhelan.ie/whelan12rssw.pdf>.
- [4] ZENG M, ZHAO F K, ZHENG J X, et al. A memory-efficient kinectfusion using octree [C]//First International Conference on Computational Visual Media. Berlin: Springer, 2012: 234-241.
- [5] HENRY P, KRAININ M, HERBST E, et al. RGB-D mapping: using Kinect-style depth cameras for dense 3D modeling of indoor environments [J]. The International Journal of Robotics Research, 2012, 31(5): 647-663.
- [6] 谭歆. KinectFusion 三维重建的再优化[D]. 杭州: 浙江大学, 2015.
- [7] 周瑾, 潘建江, 童晶, 等. 使用 Kinect 快速重建三维人体[J]. 计算机辅助设计与图形学学报, 2013, 25(6): 873-879.
- [8] 朱笑笑, 曹其新, 杨扬, 等. 一种改进的 Kinect Fusion 三维重构算法[J]. 机器人, 2014, 36(2): 129-136.
- [9] 李诗锐, 李琪, 李海洋, 等. 基于 Kinect v2 的实时精确三维重建系统[J]. 软件学报, 2016, 27(10): 2519-2529.
- [10] 胡正乙, 谭庆昌, 孙秋成. 基于 RGB-D 的室内场景实时三维重建算法[J]. 东北大学学报: 自然科学版, 2017, 38(12): 1764-1768.
- [11] NITZAN D, BRAIN A E, DUDA R O. Measurement and use of registered reflectance and range data in scene analysis [J]. Proceedings of the IEEE, 1977, 65(2): 206-220.
- [12] SALVI J, FERNANDEZ S, PRIBANIC T, et al. A state of the art in structured light patterns for surface profilometry [J]. Pattern Recognition, 2010, 43(8): 2666-2680.
- [13] ZHANG Z. A flexible new technique for camera calibration [J]. IEEE Computer Society, 2000, 22(11): 1330-1334.
- [14] KONOLIGE K, MIHELICH P. Technical description of Kinect calibration [EB/OL]. (2012-08-10). [2018-02-20]. <http://www.ros/wiki/kinect-calibration/technical>.
- [15] FOIX S, ALENYA G, TORRAS C. Lock-In time-of-flight (ToF) cameras: a survey [J]. IEEE Sensors Journal, 2011, 11(9): 1917-1926.
- [16] BESL P J, MCKAY N D. A method for registration of 3-D shapes [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence. 1992, 14(2): 239-256.