基于双 Kinect 传感器定标和配准的研究

文/张永涛[1] 赵立宏[1] 郭会娟[2]

摘要

Kinect 原本是微软公司开发的 Xbox360 主机的周边外设，主要应用于实时的人机交互工程，也有相关研究人员将其成功应用于三维重建系统。但目前利用 Kinect 实现三维重建大多数是基于单个 Kinect 传感器的应用，多 Kinect 传感器联合实现大型复杂环境的三维重建技术日益显现出优势，多传感器坐标系标定和配准的实现是第一个技术难点，本文研究了两个 Kinect 传感器定标和配准的实现过程，为实现后续的多传感器联合大型复杂环境三维重构做了基础研究。本文主要分为两步：1）同一台计算机分别对两个传感器的启用并获取两个传感器的数据进行坐标系标定；2）采用迭代就近点法 ICP（Iterative Closest Point）对两个传感器的图像进行配准。本文采用的定标和配准方法简单快捷，并且能够准确的对复杂环境进行定标且配准结果比较理想。

【关键词】Kinect 多传感器标定 三维重建 图像配准

随着信息技术和科学技术的不断发展，三维重建也在工业测量、航空航天、逆向工程、医学等领域得到了快速发展与应用，并将应用于越来越复杂的领域。Kinect 传感器是一种 RGB-D 传感器，在获得环境颜色值（RGB）还可获得深度值（depth）。并且作为一种价格相对低廉的深度传感器，在构建多传感器复合三维重构系统中有较大的优势，对于大型工件与复杂环境的三维建模，单个传感器的扫描重建速度严重限制了三维重建的效率，多传感器联合扫描重建将成为大型复杂工件与环境三维重建的一个新的发展方向，本文研究了基于双 Kinect 传感器在复杂环境下的坐标系标定和图像配准，并实现了复杂环境下坐标系的快速准确标定以及获得比较理想的图像配准结果。

1 Kinect坐标系标定现状

确定一个传感器的参数，称为传感器标定。标定目的在于确定传感器的图像坐标系与物体在空间中的三维参考坐标系的坐标对应关系，即图像间对应点的匹配关系，以及传感器自身的内部参数和不同图像间的传感器的外部参数。目前关于 Kinect 坐标系标定的方法常用的一般为定标物定标，对 Kinect 深度传感器定标时，使用棋盘格作为定标物定标，棋盘格如图 1 所示。

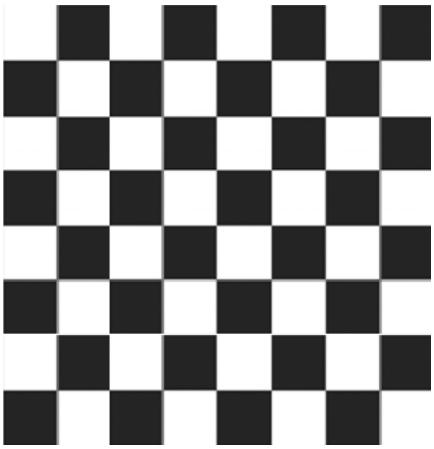

图 1

Kinect 深度传感器定标的主要步骤如下：

（1）准备棋盘格标定板，连接深度传感器。

（2）前后转动棋盘格标定板，使用 Kinect 传感器扫描棋盘格定标板在各个位置的图像，并进行图像数据存储。至少在不同平面内扫描 3 幅棋盘格在不同方位的图像，图 2 为某次扫描中的一个位置上所获得的图像。

（3）利用 Camera Calibration Toolbox，运行 calib 命令对 Kinect 传感器拍摄的图片定标，将定标结果文件存储为 Calib_Results_Kinect.mat，完成对 Kinect 传感器的定标。

2 双Kinect传感器定标方法设计

双 Kinect 传感器定标基本方法如上对每个 Kinect 传感器分别进行定标，完成每个 Kinect 传感器的内部参数矩阵标定，双 Kinect 传感器定标设计主要是两个 Kinect 传感器外部旋转矩阵 R 和平移矩阵 T 参数的定标，标定的外部参数有 6 个，即旋转矩阵 R 中绕三个坐标轴的旋转角以及平移矩阵 T 中沿三个坐标轴方向的位移，空间三维参考坐标系到摄像机坐标系的变换，从 (x_w, y_w, z_w) 到 (x, y, z)

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = R_i \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + T_i \qquad (1)$$

假设两个传感器的外部参数分别为 R_l、T_l 和 R_r、T_r，则 R_l、T_l 表示左边传感器与世界坐标系的相对位置，R_r、T_r 表示右边传感器与世界坐标系的相对位置，两个传感器之间的集合关系 R、T 如下关系式表示：

$$R=R_lR_l^{-1} \quad T=T_r-R_lR_l^{-1}T_l \qquad (2)$$

矩阵 R_l、T_l 和 R_r、T_r 是采用同一棋盘格同时分别对每个传感器进行定标，从而获得的对应传感器与世界坐标系的外部参数，由式（1）可以计算两个 Kinect 传感器之间的对应几何位置。

图 2

3 程序方法实现

首先，启动程序运行至如下程序段分别启动第一、第二 Kinect 传感器。打开过程中如果第一个 Kinect 传感器未连接、出现故障或没有选择画面分辨率时，输出“打开左 Kinect 传感器失败”提醒，打开成功则不提醒；当要打开第二个人 Kinect 传感器时首先定义与第一个 Kinect 传感器的设备号不能相同，然后打开第二个 Kinect 传感器。

```
void CStereoVisionDlg::OnBnClkRunCam()
{   if (m_nImageWidth * m_nImageHeight * m_nImageChannels == 0)
{   AfxMessageBox(_T(" 请选择 Kinect 传感器画面的分辨率！"));
return;
}
if (m_nCamCount > 0)
{   // 打开第一个 Kinect 传感器
if ( !lfCam.open(m_lfCamID) )
{   AfxMessageBox(_T(" 打开左 Kinect 传感器失败 ."));
return;
}
lfCam.set(CV_CAP_PROP_FRAME_WIDTH , m_nImageWidth);
lfCam.set(CV_CAP_PROP_FRAME_HEIGHT, m_nImageHeight);
}
if (m_nCamCount > 1)
{   if (m_lfCamID == m_riCamID)
{   AfxMessageBox(_T(" 左右 Kinect 传感器的设备序号不能相同！"));
return;
}
// 打开第二个 Kinect 传感器
if ( !riCam.open(m_riCamID) )
{   AfxMessageBox(_T(" 打开右 Kinect 传感器失败 ."));
```

```
return;
}
riCam.set(CV_CAP_PROP_FRAME_WIDTH, m_nImageWidth);
riCam.set(CV_CAP_PROP_FRAME_HEIGHT, m_nImageHeight);
GetDlgItem( IDC_BN_CompDisp )->EnableWindow( TRUE );
}
```

其次，通过程序对传感器定标量进行初始化和棋盘格参数的设定，确认两个 Kinect 传感器的定标次序，当只有一个传感器时不能进行双目定标。再提取数据正确并保存的前提下控制 Kinect 传感器是否开始读取下一帧数据。

最后，通过以下程序实现双 Kinect 传感器的定标过程。

```
// 进行摄像头标定
if (optCalib.doStereoCalib) // 双目标定和校正
{   stereoParams.cameraParams1.flags = optCalib.flagCameraCalib;

stereoParams.cameraParams2.flags = optCalib.flagCameraCalib;
stereoParams.flags = optCalib.flagStereoCalib;
stereoParams.alpha = optCalib.alpha;
bool needCalibSingleCamera = (CALIB_SINGLE_CAMERA_FIRST == optCalib.calibOrder);
m_stereoCalibrator.calibrateStereoCamera(cornerDatas, stereoParams, needCalibSingleCamera);
// 计算标定误差
double avgErr = 0;
m_stereoCalibrator.getStereoCalibrateError(cornerDatas, stereoParams, avgErr);
char info[50];
sprintf_s( info, " 标 定 误 差 = %7.4g", avgErr );
CString ss;
ss = info;
AfxMessageBox(ss);
// 执行双目校正
m_stereoCalibrator.rectifyStereoCamera(cornerDatas,stereoParams,remapMatrixs, ptCalib.rectifyMethod);
AfxMessageBox(_T(" 已完成双目校正 "));
// 保存摄像头定标参数
filePath = m_workDir;
filePath.AppendFormat("calib_paras.xml");
m_stereoCalibrator.saveCalibrationDatas(filePath.GetBuffer(0),optCalib.rectifyMethod, ornerDatas, stereoParams, remapMatrixs);
AfxMessageBox(_T(" 已保存定标参数 "));
```

4 双Kinect传感器图像配准研究

4.1 图像的配准

图像配准是将不同时间、不同传感器或不同条件下（如光照、气候、位置角度等）获取的两幅或者多幅图像进行匹配和叠加的过程。本文选用迭代就近点法 ICP（Iterative Closest Point），即基于自由形态曲面的配准方法。以点集对点集（PSTPS）配准方法为基础，阐述了一种曲面拟合算法，该算法是基于四元数的点集到点集配准方法。从测量点集中确定其对应的就近点点集后，运用 Faugera 和 Hebert 提出的方法计算新的就近点点集。用该方法进行迭代计算，直到残差平方和所构成的目标函数值不变，结束迭代过程。

4.2 ICP算法

ICP 算法是最常用的数据精确配准方法，该算法在每次的迭代过程中，对数据点云中每一点，在模型点云中寻找欧式距离最近点作为对应点，在双 Kinect 传感器采集图像基础上，以其中第一个 Kinect 传感器采集的图像作为模型点云，以第二个 Kinect 传感器采集到的图像作为数据点云， 通过这组对应的数据点使得目标函数最小化：

$$R_m=\min\sum_{i=1}^{N}\| Q_i - R_mP_i+t_m \|^2$$

从而得出最优的配准旋转矩阵 R_m 与配准平移向量 t_m。

4.3 双Kinect传感器图像的配准过程

将上述得出的配准旋转矩阵 R_m 与配准平移向量 t_m 作用于数据点云进行相对应的旋转和平移，得出新的数据点云并进入下次迭代过程之中，最后转化到模型数据点云参考系中，实现两个 Kinect 传感器采集的图像之间的配准。

5 实验结果验证

为验证双 Kinect 传感器定标完成后两个传感器所采集的图像配准结果，在 Intel(R)Core(TM)i5-3230M CPU @2.6GHz 主频 4.00GB 内存的计算机上应用 PCL（Point Cloud Library）开发平台进行验证。

图 3.1 为配准前第一个（左）Kinect 传感器采集的模型点云图像，图 3.2 为配准前第二个（右）Kinect 传感器采集的点云图像。图 3.1 和图 3.2 的点云数据经过 ICP 算法的配准处理后，输出图像如图 3.3 和图 3.4 所示，其中图 3.3 为配准处理后输出图形，而图 3.4 是在图 3.3 的基础上为显示配准后的三维效果手动旋转某一角度后所得图像，由配准后图像可知配准结果相对较为理想，达到了预期目的。

6 结语

实验结果表明，对两个 Kinect 传感器经过棋格盘定标后，可以实现双 Kinect 传感器采集图像并实现两个传感器所采集图像的配准，为将来多传感器的标定和配准打下基础。并为大型工件与复杂环境的三维重构提供了一种新的路径，即采用双传感器或多传感器联合实现快速而准确的三维重构。

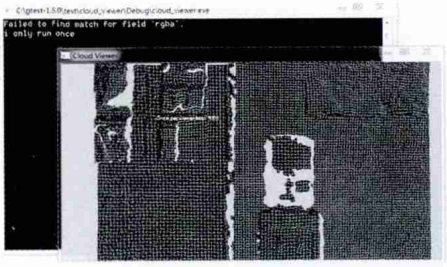

图 3.1：配准前第一个 Kinect 传感器采集的图像

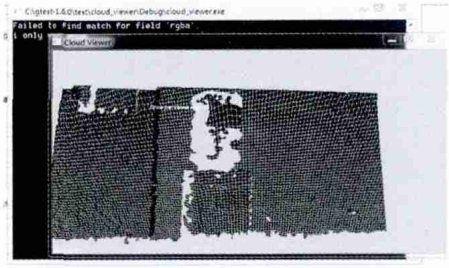

图 3.2：配准前第二个 Kinect 传感器采集的图像

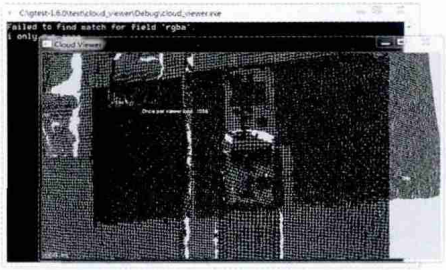

图 3.3：配准后的图像

图 3.4：配准后旋转某一角度的图像

（通讯作者：郭会娟）

参考文献

[1] 刘鑫，许华荣，胡占义．基于 GPU 和 Kinect 的快速物体重建 [J]．自动化学报，2012(08)．

[2] 杨现辉，王惠南．ICP 算法在 3D 点云配准中的应用研究 [J]．计算机仿真，2010(08)．

作者单位

1. 南华大学机械工程学院　湖南省衡阳市 421001

2. 黄河科技学院工学院　河南省郑州市 450000