# OpenCV 棋盘格外定标对比pcl\_kinfu\_app ICP迭代外定标测试报告

## Changelog

|  |  |  |  |
| --- | --- | --- | --- |
| 版本号 | 变更人 | 变更说明 | 变更时间 |
| V1.0 | 张琛 | 初稿 | 2014/12/6 |
| V1.1 | 张琛 | 添加小节 3.1.4, 表格3; 3.2.3 | 2014/12/9 |
|  |  |  |  |

## 概述

### 编写目的

本报告旨在对比使用pcl\_kinfu\_app 中ICP迭代外定标（记为方法M1），与OpenCV 棋盘格外定标（记为方法M2）两种不同方法的结果，并对产生误差的原因进行分析。

### 测试环境

PC: /i5-3470 CPU @ 3.20GHz /6GB RAM /Windows7 x64

IDE: Visual Studio 2010 sp1

Lang: C++, python (ipython-notebook)

Lib: 3rd party @PCL 1.6.0-x86; Opencv2.4.9;

Src: pcl-master-6d0343d1b7

## 测试过程

### 使用离线数据文件

采集离线ONI数据文件的方式为：在高精度立方体（20\*20\*20 ）侧面粘贴棋盘格，立方体上方放置一个四面体；kinect设备距离棋盘格表面约 80~130cm，进行单轴或者整体的旋转&平移。采集场景如图1所示。注意，所有离线数据一律设定为 mirror = ON; registration = ON 模式。

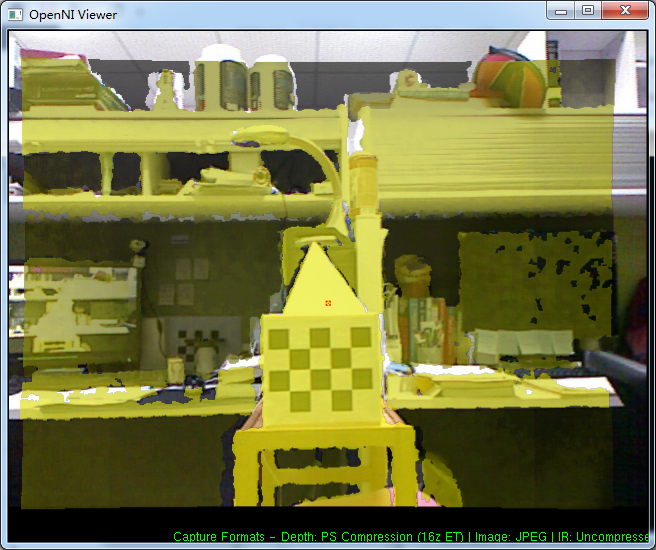


图 1 使用OpenNI 采集离线数据的场景

表格 1 oni数据文件及其描述

|  |  |  |  |
| --- | --- | --- | --- |
| 测试序号 | 文件名 | 棋盘格角点数 | 对应动作描述 |
| 1 | smt-d100-sz35.8-r2l-t2b-refine.oni | 4x3 | 绕立方体从右到左旋转+平移，再从上到下旋转平移 |
| 2  3 | smt-d100-sz35.8-horizon-move-part1.oni | 3x4  4x3 | 由正对面，向右绕行 |
| 4 | smt-d100-sz35.8-vertical-move.oni | 4x3 | 上下绕行，顺序：正对面，向上（俯视），向下（仰视），回到正面 |
| 5 | smt-d100-zrot-7x8.oni | 8x7 | 绕Z轴，逆时针90°，再顺时针90°，回到正面 |
| 6 | smt-d100-xyzrot-7x8-3axes.oni | 8x7 | 先后绕XYZ轴分别单独负向旋转 |
| 7 | smt-d100-xyztrans-7x8-refine.oni | 8x7 | 先后沿XYZ轴做（正向-回归）平移运动。尽量不发生旋转 |
| 8 | bigboard-d130-7x10.oni | 10x7 | 场景内使用大定标板（格子边长50mm），无限制慢速旋转平移 |
| 9 | bigboard-slope-7x10.oni | 10x7 | 确保设备开始采样时倾斜，使相机坐标轴不与棋盘格（世界）坐标系重叠 |

### 导出参数文件

因为vs中C++工程较大，构建调试十分耗时，所以对于离线ONI视频数据，离线对比M1, M2的定标结果。分别导出两组 (R, t) 序列，使用python脚本进行参数绘制，观察对比两个结果。同时，也可以验证坐标转换公式推导是否正确。

表格 2 oni数据对应两种方法导出的外参文件

|  |  |  |  |
| --- | --- | --- | --- |
| 测试序号 | Pcl\_kinfu\_app 导出 csv | OpencvCalib 导出 xml | 备注 |
| 1 | pose\_file-smt-d100-sz35.8-r2l-t2b-refine.csv | smt-d100-sz35.8-r2l-t2b-refine\_calib\extrinsics.xml |  |
| 2 | pose\_file-smt-d100-sz35.8-horizon-move-part1.csv | smt-d100-sz35.8-horizon-move-part1\_calib\extrinsics-3x4.xml | extrinsics 参数设定3x4， 非 4x3 |
| 3 | 同上 | smt-d100-sz35.8-horizon-move-part1\_calib\extrinsics-4x3.xml | 参数4x3, 与测试序号2对比 |
| 4 | pose\_file-smt-d100-sz35.8-vertical-move.csv | smt-d100-sz35.8-vertical-move\_calib\extrinsics-4x3.xml |  |
| 5 | pose\_file-smt-d100-zrot-7x8.csv | smt-d100-zrot-7x8\_calib\extrinsics.xml |  |
| 6 | pose\_file-smt-d100-xyzrot-7x8-3axes.csv | smt-d100-xyzrot-7x8-3axes\_calib\extrinsics1-22.3.xml | 棋盘格一格边长 22.3mm |
| 7 | pose\_file-smt-d100-xyztrans-7x8-refine.csv | smt-d100-xyztrans-7x8-refine\_calib\extrinsics.xml |  |
| 8 | pose\_file-bigboard-d130-7x10.csv | bigboard-d130-7x10\_calib\extrinsics.xml |  |
| 9 | pose\_file-bigboard-slope-7x10**-285**.csv | bigboard-slope-7x10\_calib\extrinsics-50.xml | 棋盘格一格边长 50mm |

## 测试结果

### 结果描述

* + 1. Pcl深度图ICP迭代（记为方法M1）得到的 (R, t) 曲线更光滑，如图2；
    2. OpenCV棋盘格外定标（记为方法M2）得到的 (R, t) 曲线波动较大，如图3、4(CMY曲线)、5(CMY曲线)；
    3. 方法M2在某些地方存在“直角尖峰”现象，如图12；同时，M2生成导出的外参序列元素个数比M1多约 10%
    4. 下表格3展示了采用 RMSE 量化表述两组曲线（**点集**） ， 之间的偏差。其中，

1. E为离线外定标旋转矩阵R的欧拉角表示（XYZ三轴，单位：弧度）；T为其偏移量（XYZ三轴，单位：米）；
2. ;

;

表格 3 两种外定标结果的对比 RMSEe & RMSEt

|  |  |  |
| --- | --- | --- |
| 测试序号 | XYZ（单位：弧度） | XYZ（单位：米） |
| 1 | 0.341, 0.147, 0.182 | 0.086, 0.238, 0.079 |
| 2 | 0.263, 0.109, 0.097 | 0.121, 0.149, 0.140 |
| 3 | 0.133, 0.109, 0.014 | 0.122, 0.122, 0.052 |
| 4 | 0.296, 0.075, 0.167 | 0.043, 0.217, 0.086 |
| 5 | 0.057, 0.098, 0.847 | 0.122, 0.115, 0.012 |
| 6 | 0.174, 0.206, 0.399 | 0.139, 0.092, 0.059 |
| 7 | 0.090, 0.057, 0.008 | 0.064, 0.095, 0.015 |
| 8 | 0.076, 0.078, 0.074 | 0.132, 0.104, 0.083 |
| 9 | 0.043, 0.057, 0.041 | 0.061, 0.052, 0.054 |

### 结论分析

* + 1. M2容易出现角点检测失败（找不到、找不准）的情况。影响方法M2外定标因素有：

1. 棋盘格在场景图片中的大小。整个棋盘格像素宽高较小，导致角点检测不精确；
2. 棋盘格格子数目。即对应角点个数，角点个数太少导致solvePnP目标方程优化解不精确；
3. 设备移动速度。设备移动速度稍快，会造成彩色图模糊，导致彩色图像模糊（原本kinect一代色彩分辨率只有640\*480）；
   * 1. M2“直角尖峰”现象是因为某些时候算法角点检测失败，而目前我的策略为失败时继续使用上一帧 (R, t) 作为当前帧的值。可以看出，视频帧序列中，角点检测失败的情形出现较为频繁，如图12。
     2. 对于表格3所示的结果，目前主要的误差来源是两组数据之间的帧率不同步，而离线导出的外参序列没有对应的时间戳序列，所以两组外参存在时序上的偏移，并不能很好的表示误差。

对于这个问题，目前计划是尝试在 pcl\_kinfu\_app 代码里同时进行 OpenCV 外定标， 以尽量保证两者输出的外参是同步的。这个测试尚未完成。

### 结果附图

因测试太多，本小节截取展示部分代表性结果图。

测试序号1：

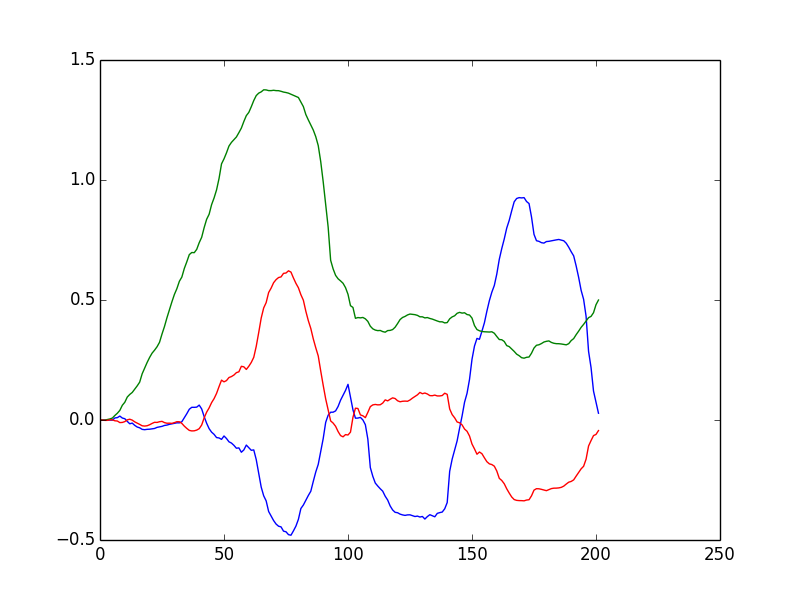


图 2 M1计算得到的欧拉角E1，XYZ三轴分别对应BGR三色

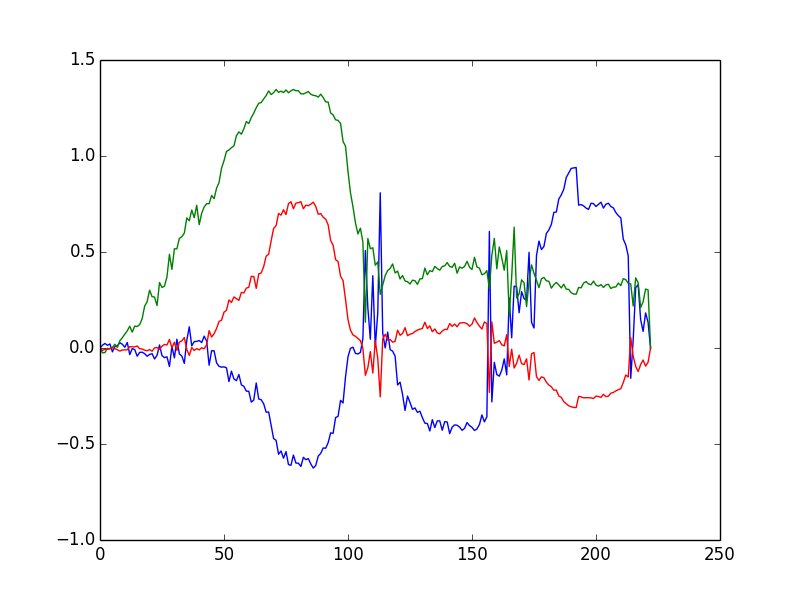


图 3 M2计算得到的欧拉角E2，XYZ三轴分别对应BGR三色

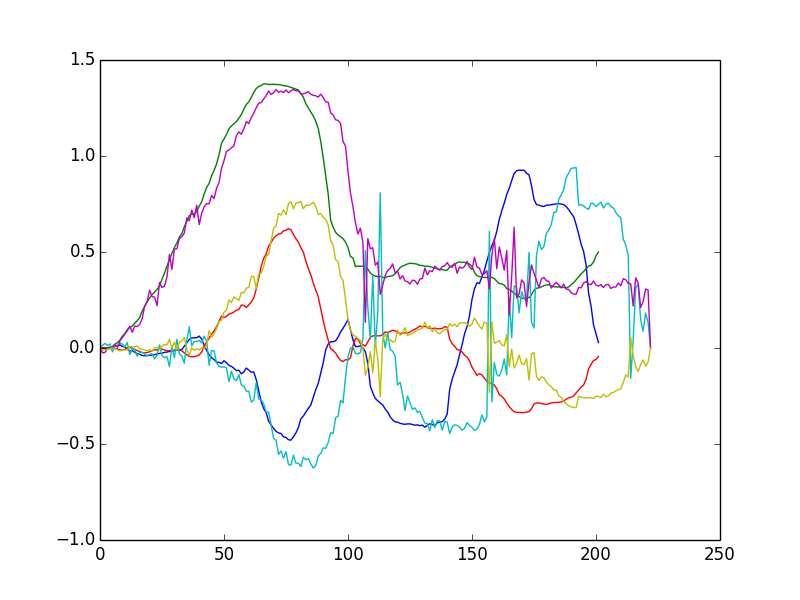


图 4 E1 (BGR序) 与 E2 (CMY序)绘制在同一幅图

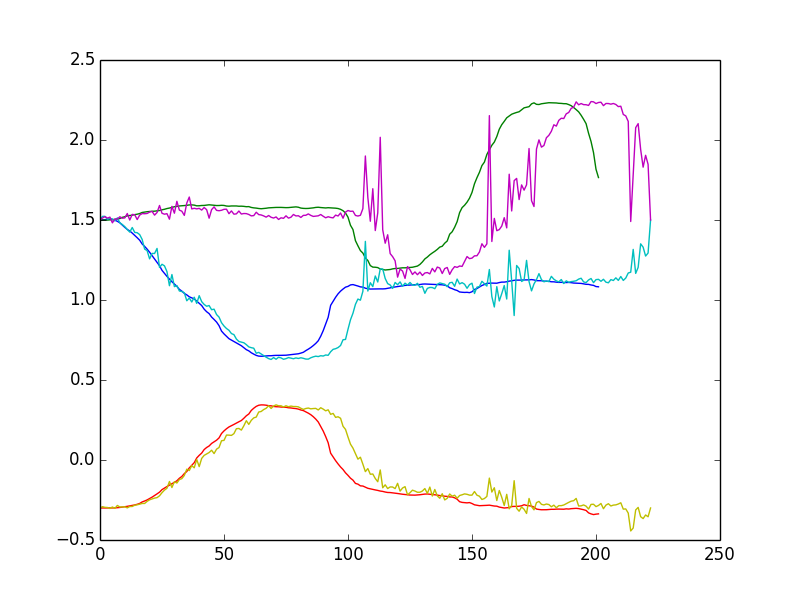


图 5 M1平移曲线T1 (BGR序)与M2的平移曲线T2 (CMY序)绘制在同一幅图

测试序号2：

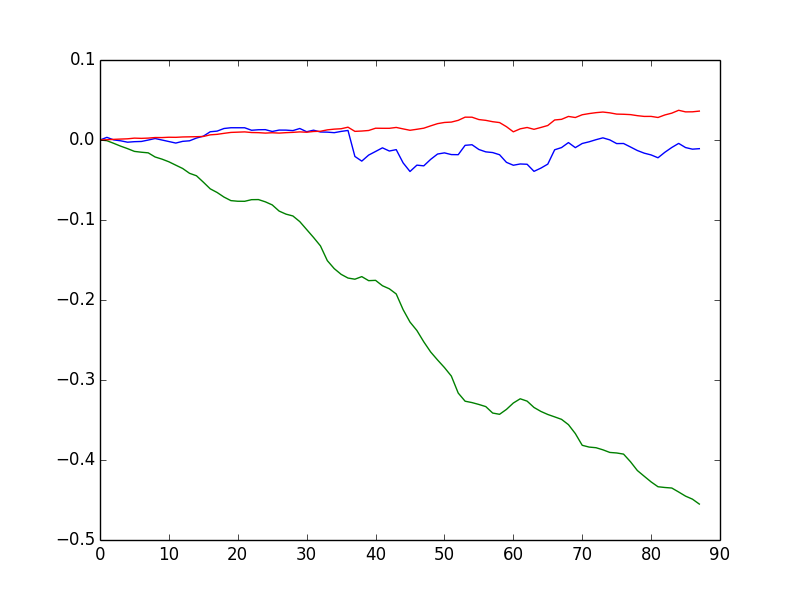


图 6 M1计算得到的欧拉角E1，XYZ三轴分别对应BGR三色

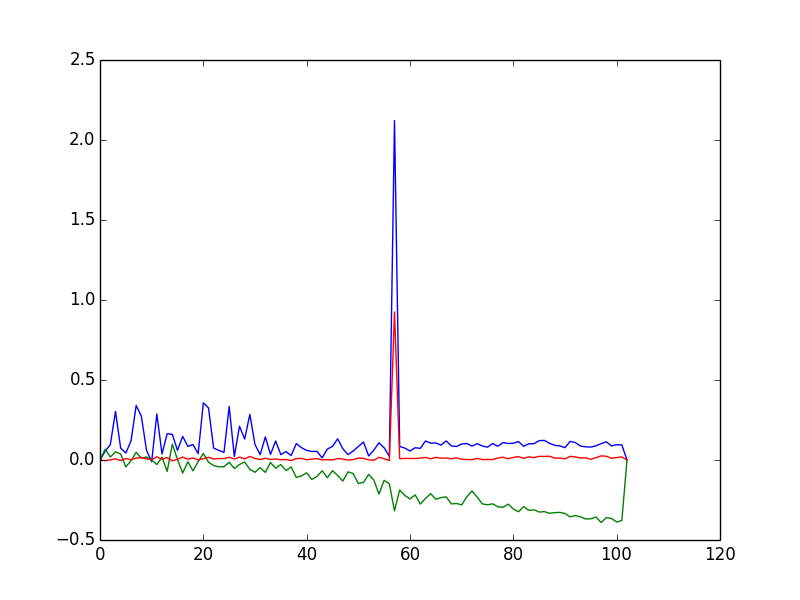


图 7 M2计算得到的欧拉角E2，XYZ三轴分别对应BGR三色

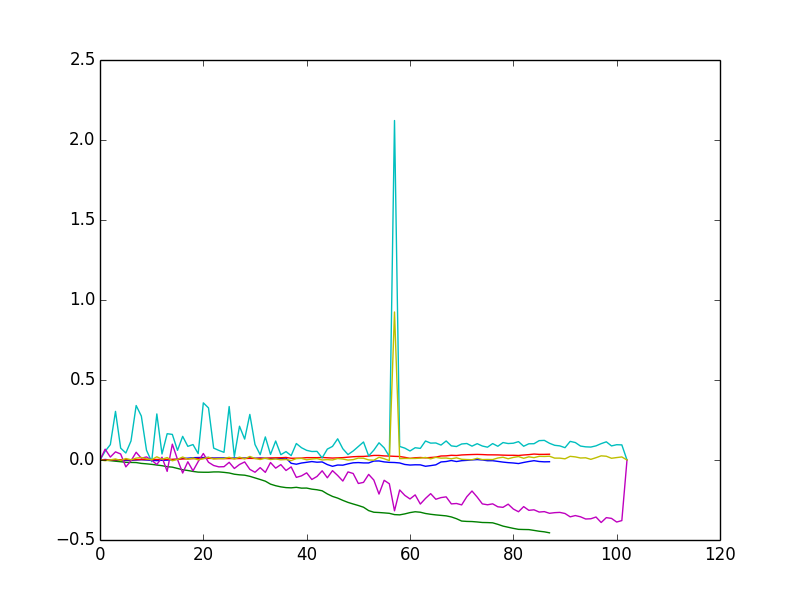


图 8 E1 (BGR序) 与 E2 (CMY序)绘制在同一幅图

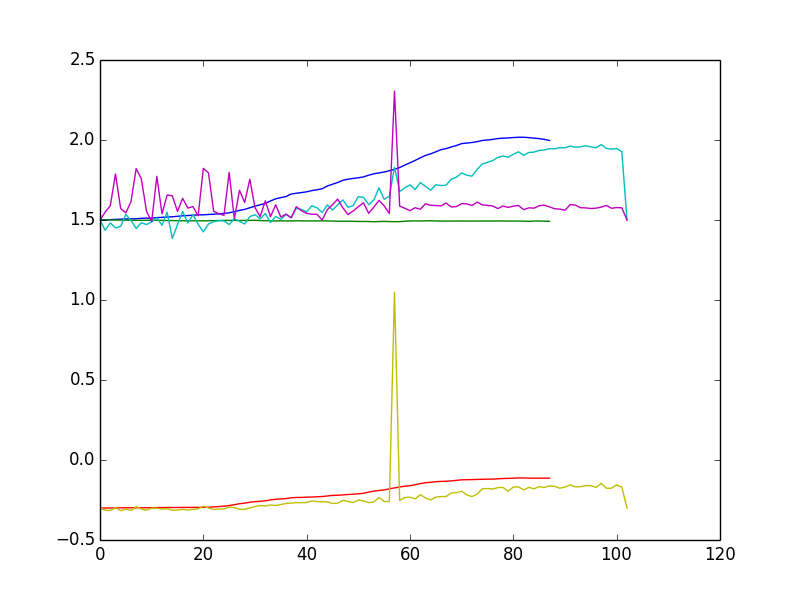


图 9 M1平移曲线T1 (BGR序)与M2的平移曲线T2 (CMY序)绘制在同一幅图

测试序号7：

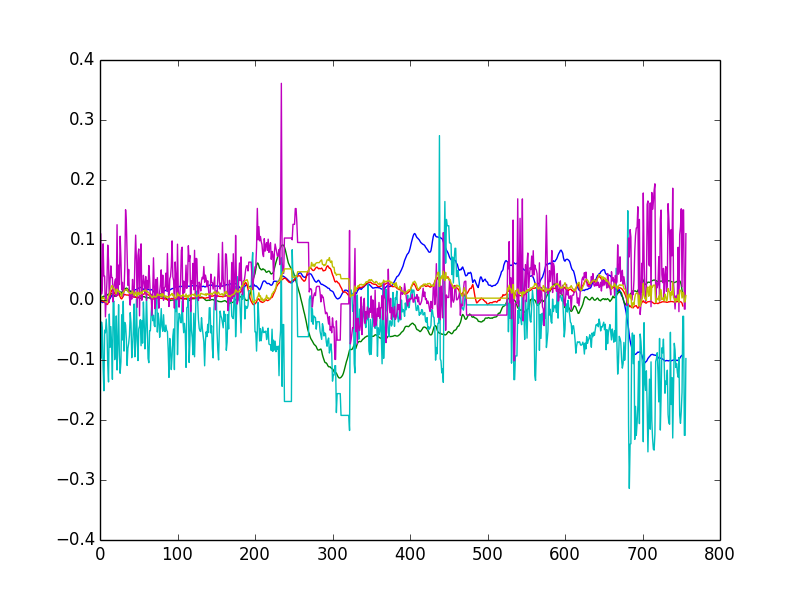


图 10 E1 (BGR序) 与 E2 (CMY序)绘制在同一幅图，结果差

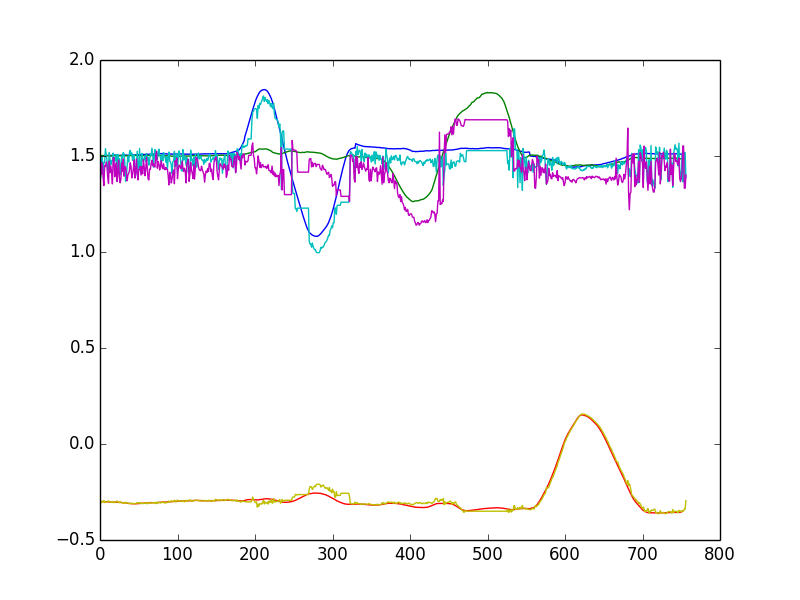


图 11 M1平移曲线T1 (BGR序)与M2的平移曲线T2 (CMY序)绘制在同一幅图，略优于图5、9

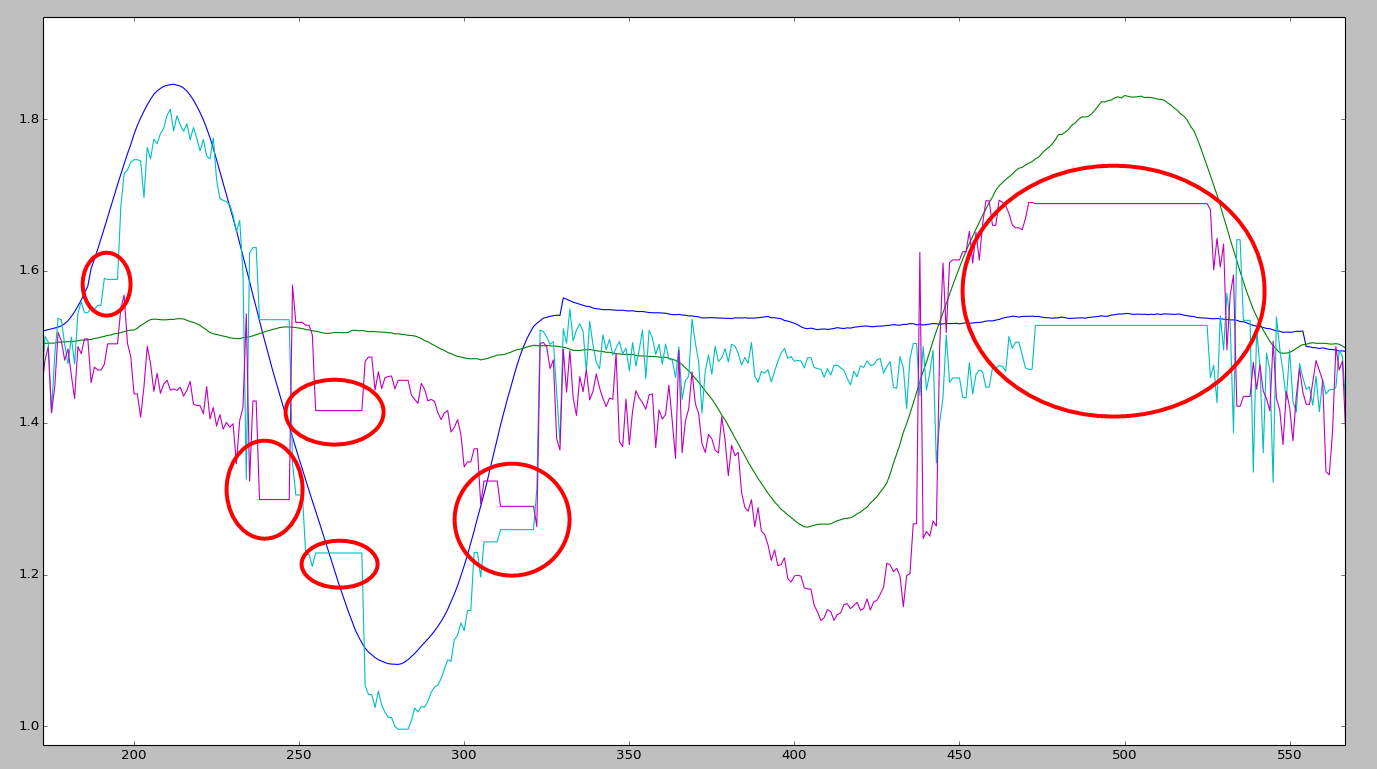


图 12 对图11进行局部放大，观察到“直角尖峰”现象

测试序号8：

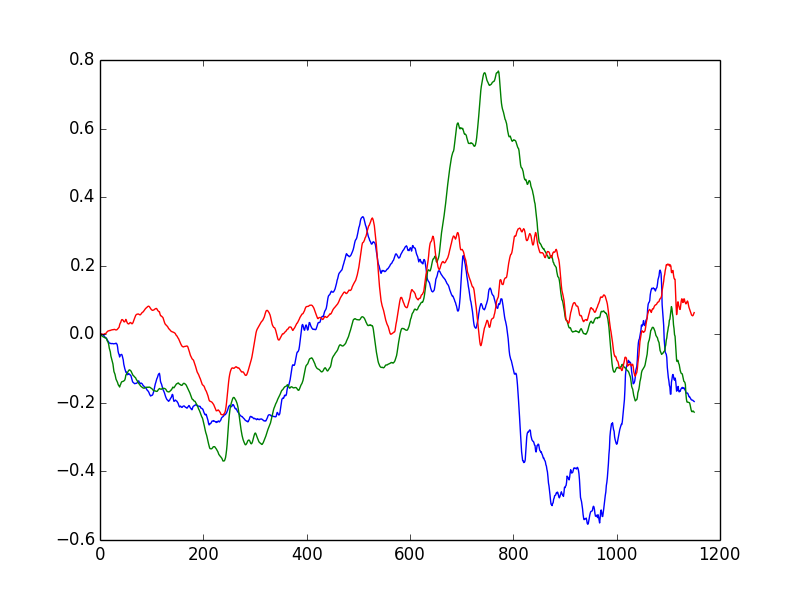


图 13 M1计算得到的欧拉角E1，XYZ三轴分别对应BGR三色

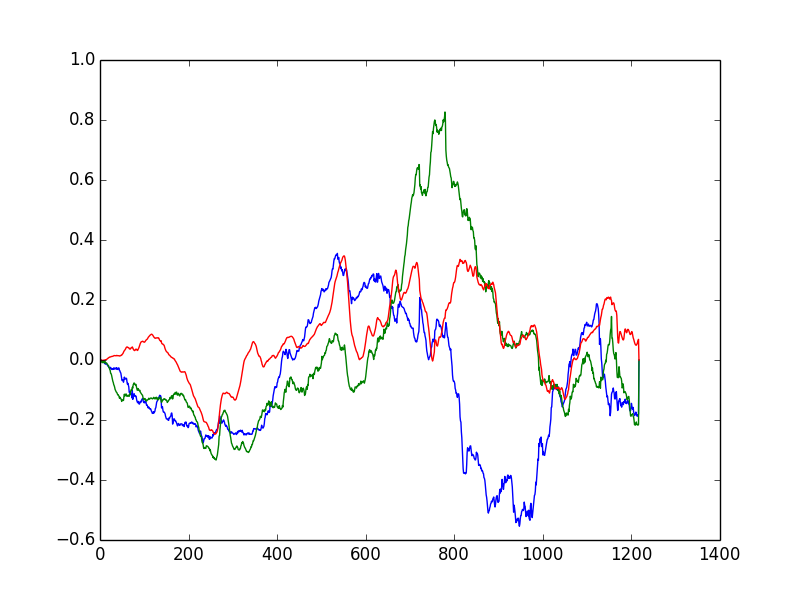


图 14 M2计算得到的欧拉角E2，XYZ三轴分别对应BGR三色

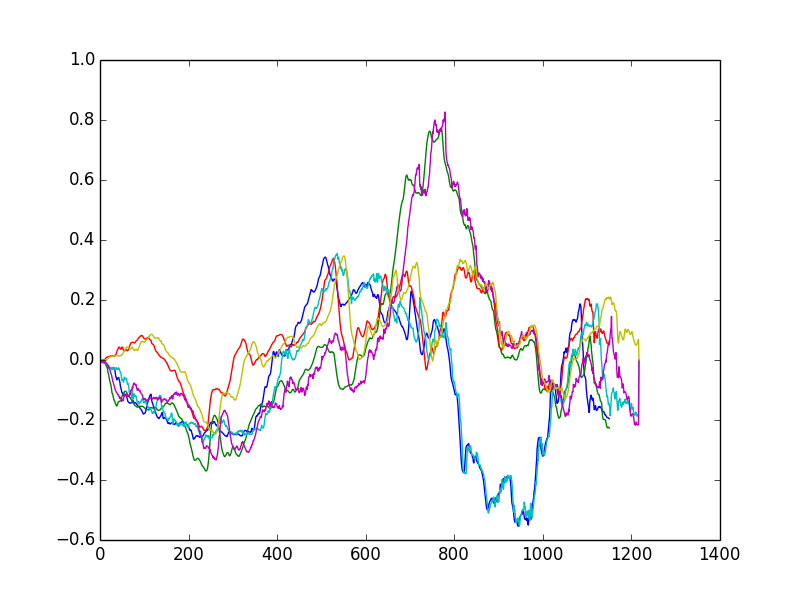


图 15 M1平移曲线T1 (BGR序)与M2的平移曲线T2 (CMY序)绘制在同一幅图

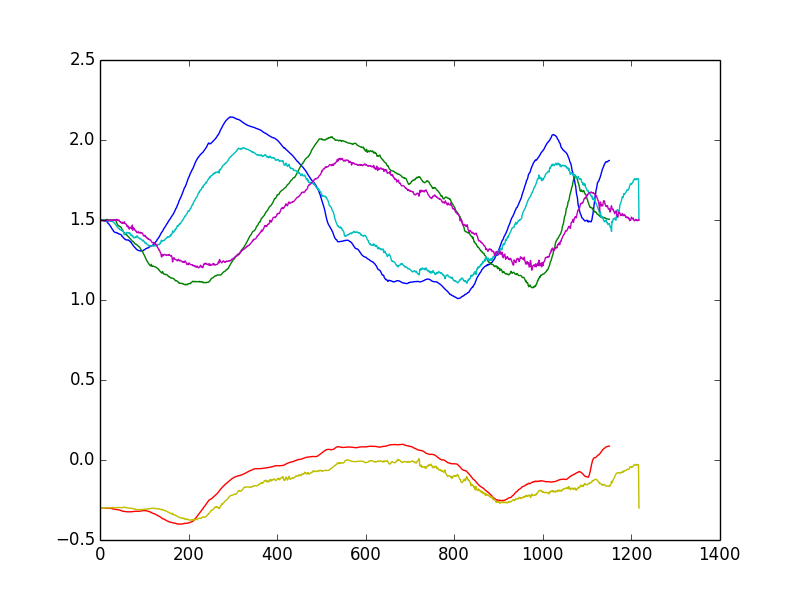


图 16 M1平移曲线T1 (BGR序)与M2的平移曲线T2 (CMY序)绘制在同一幅图