# PCL库知识

## 基本知识

PCLVisualizer 是PCL的一个功能健全的类，尽管它比CloudViewer要复杂，但其功能也更强大，提供 显示 法向量，画形状以及多视角功能

如果要更新已有点云数据，updatePointCloud(),addPointCloud(),removePointCloud()

viewer->setPointCloudRenderingProperties

(pcl::visualization::PCL\_VISUALIZER\_POINT\_SIZE, 3, "sample cloud");

表示点的大小

X (red), Y (green) and Z (blue)为真实世界坐标系，viewer->addCoordinateSystem (1.0);为默认，如果没有也是一样

viewer->addPointCloudNormals<pcl::PointXYZRGB, pcl::Normal> (cloud, normals, 10,

0.05, "normals");法向量，后面两个参数分别是每十个法向量显示一个，以及法线长度

viewer->resetCameraViewpoint("sample cloud");//使显示点云中间，非常有用！

根据扫描设备可将点云分有序点云和无序点云 有序点云: 在行列方向上有恒定的密度进行排列 无序点云: 不定的密度在空间的任一位置存在（这样的例子很多）

## 关键点(keypoint)

### 定义

通过给定标准获取的具有 稳定性 区别性的 点集.其数量比原始点云的数据量减小很多.与局部特征描述子结合一起,组成关键点描述子.

### PCL中的keypoints模块

pcl\_keypoints模块利用7个类实现了利用几种关键点检测算法相关的数据结构与检测方法,其依赖于common search kdtree octree range\_image features filters模块

## 深度图像

### 定义

深度图像(Depth Images)也被称为距离影像(Range Images),图像的像素值是 从 图像采集器 到场景中个点的距离(深度).一般描述为range images

深度图像是物体的三维表示形式,一般通过立体照相机或者ToF照相机获取,如果具备照相机的内标定参数,就可以将深度图像转换为点云

### PCL中的pcl\_range\_image模块

pcl\_range\_image模块包含两个表达深度图像和对操作深度图像类.其依赖于pcl::common模块.

#### Class pcl::RangeImage类

该类继承于PointCloud,主要功能是实现从一个特定视点得到一个三维场景的深度图像.

Void createFromPointCloud

## 点云配准

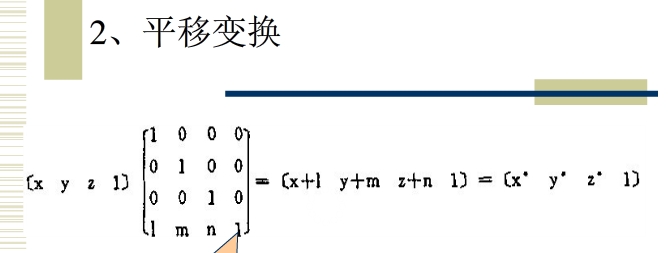
### 刚体变换矩阵

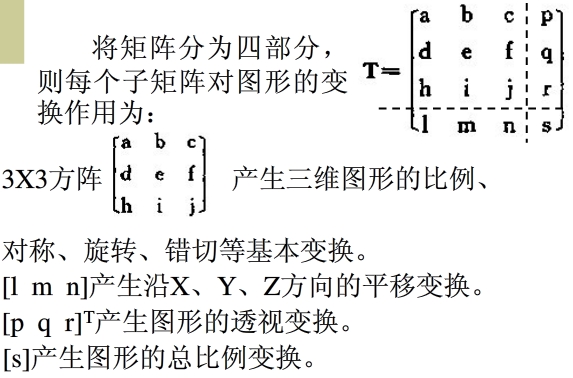
[三维空间](http://baike.baidu.com/view/175999.htm)中， 把一个[几何](http://baike.baidu.com/view/15136.htm" \t "_blank)物体作[旋转](http://baike.baidu.com/view/131763.htm)， [平移](http://baike.baidu.com/view/131749.htm)的[运动](http://baike.baidu.com/view/50853.htm)，称之为[刚体](http://baike.baidu.com/view/68357.htm)[变换](http://baike.baidu.com/view/422035.htm)。

[刚体](http://baike.baidu.com/view/68357.htm)[运动](http://baike.baidu.com/view/50853.htm)也可以理解为保持[长度](http://baike.baidu.com/view/87781.htm)，[角度](http://baike.baidu.com/view/524959.htm)，[面积](http://baike.baidu.com/view/898.htm)等不变的[仿射变换](http://baike.baidu.com/view/954621.htm)， 即保持[内积](http://baike.baidu.com/view/22008.htm" \t "_blank)和[度量](http://baike.baidu.com/view/707359.htm)不变。

从[坐标变换](http://baike.baidu.com/view/290245.htm" \t "_blank)上看， [旋转](http://baike.baidu.com/view/131763.htm)对应[行列式](http://baike.baidu.com/view/111348.htm)为1的[正交矩阵](http://baike.baidu.com/view/21085.htm" \t "_blank)。

此外，[刚体](http://baike.baidu.com/view/68357.htm" \t "_blank)[变换](http://baike.baidu.com/view/422035.htm)下， 具有[物理意义](http://baike.baidu.com/view/1973663.htm" \t "_blank)的量，如[梯度](http://baike.baidu.com/view/454441.htm)，[散度](http://baike.baidu.com/view/454440.htm)和[旋度](http://baike.baidu.com/view/445017.htm)都保持不变。





### 点云配准步骤

首先冲两个数据集在中按照同样的关键点选取标准,提取关键点.

对选择的所有关键点分别计算其特征描述子

结合特征描述子在两个数据集中的坐标的位置,以二者之间特征和位置的相似度为基础,来估算它们的对应关系,初步估计对应点对.

假定数据是有噪声的,除去对配准有影响的错误的对应点对.

利用剩余的正确对应关系来估算刚体变换,完成配准.

整个配准过程中最重要的步骤是关键点的提取和关键点的特征描述,以确保对应估计的准确性和效率,是后续配准过程的基础.

# MFC PCL

error C2661: 'pcl::PointCloud<PointT>::operator new' : no overloaded function takes 3 arguments   
It can be solved by comment out followings in the XXXDlg.cpp   
//#ifdef \_DEBUG   
//#define new DEBUG\_NEW   
//#endif

如std::max()这种的要改成(std::max)()

# QT PCL

## 开发环境搭建(vs2010)

[原创]搭建vs2010下PCL开发环境

2012-02-27 17:14:10|  分类： [learning](http://blog.163.com/gz_ricky/blog/#m=0&t=1&c=fks_084065087087086067085094086095085094083067080095087066092) |  标签：[lnk2019](http://blog.163.com/gz_ricky/blog/" \l "m=0&t=3&c=lnk2019)  [pcl](http://blog.163.com/gz_ricky/blog/" \l "m=0&t=3&c=pcl)  [vtk](http://blog.163.com/gz_ricky/blog/" \l "m=0&t=3&c=vtk)  [vs2010](http://blog.163.com/gz_ricky/blog/#m=0&t=3&c=vs2010)  [三维云点](http://blog.163.com/gz_ricky/blog/#m=0&t=3&c=三维云点)  |字号 订阅

走了真多弯路，哎~记录一下吧！

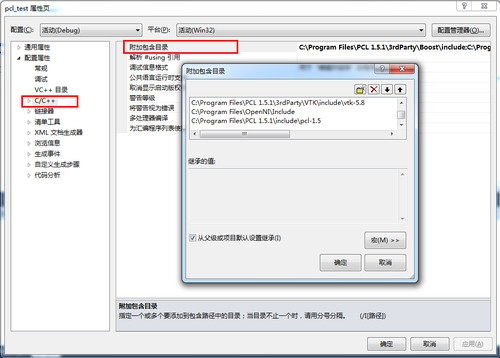
1.首先在<http://www.pointclouds.org/downloads/windows.html>下载windows的安装包。我用的是预编译版本的多合一包，需要自己编译的也可以。注意下载时要看清楚vs2010的版本是32位还是64位，不然一些库会找不到对应入口，导致编译不通过的。

tips：可以在开始菜单下的Microsoft Visual Studio 2010文件夹下，查看Microsoft Windows SDK Tools，如果看到程序后面（如IL反汇编程序）有(x64)字样，则vs是64位的，没有就放心安装32位版本吧。

如果是用cmake生成pcl解决方案，要注意在cmake的时候确定各个依赖库的路径，可以在对应项上把鼠标停留一下，看看有什么提示信息，特别是VTK的目录。有些时候能够继续生成，但是部分模块并没有设置好参数，会导致编译出来的PCL运行不良好。

2.安装完毕后就测试一下PCL的效果吧，要注意项目对应的包含目录、库目录、库文件（LIB）和运行环境（DLL）。

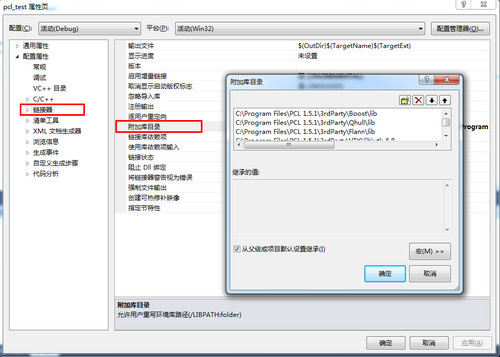
用<http://www.pointclouds.org/documentation/tutorials/pcl_visualizer.php#pcl-visualizer>上面的例程试一试，新建工程，粘贴代码……



在包含目录下添加：

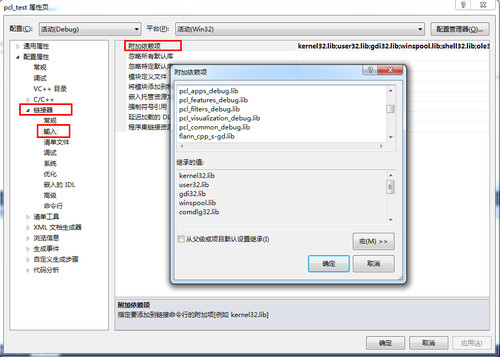
C:\Program Files\PCL 1.5.1\3rdParty\Boost\include;  
C:\Program Files\PCL 1.5.1\3rdParty\Eigen\include;  
C:\Program Files\PCL 1.5.1\3rdParty\Flann\include;  
C:\Program Files\PCL 1.5.1\3rdParty\Qhull\include;  
C:\Program Files\PCL 1.5.1\3rdParty\VTK\include\vtk-5.8;  
C:\Program Files\OpenNI\Include;  
C:\Program Files\PCL 1.5.1\include\pcl-1.5;

（对应自己机子上PCL的位置，如果为了CMAKE方便很多会放在根目录下。即C:\PCL）



在库目录下添加：

C:\Program Files\PCL 1.5.1\3rdParty\Boost\lib;  
C:\Program Files\PCL 1.5.1\3rdParty\Qhull\lib;  
C:\Program Files\PCL 1.5.1\3rdParty\Flann\lib;  
C:\Program Files\PCL 1.5.1\3rdParty\VTK\lib\vtk-5.8;  
C:\Program Files\PCL 1.5.1\lib;  
C:\Program Files\OpenNI\Lib;



在附加依赖项上添加：

 opengl32.lib  
pcl\_kdtree\_debug.lib  
pcl\_io\_debug.lib  
pcl\_search\_debug.lib  
pcl\_segmentation\_debug.lib  
pcl\_range\_image\_border\_extractor\_debug.lib  
pcl\_apps\_debug.lib  
pcl\_features\_debug.lib  
pcl\_filters\_debug.lib  
pcl\_visualization\_debug.lib  
pcl\_common\_debug.lib  
flann\_cpp\_s-gd.lib  
libboost\_system-vc100-mt-gd-1\_47.lib  
libboost\_filesystem-vc100-mt-gd-1\_47.lib  
libboost\_thread-vc100-mt-gd-1\_47.lib  
libboost\_date\_time-vc100-mt-gd-1\_47.lib  
libboost\_iostreams-vc100-mt-gd-1\_47.lib  
openNI.lib  
vtkalglib-gd.lib  
vtkCharts-gd.lib  
vtkCommon-gd.lib  
vtkDICOMParser-gd.lib  
vtkexoIIc-gd.lib  
vtkexpat-gd.lib  
vtkFiltering-gd.lib  
vtkfreetype-gd.lib  
vtkftgl-gd.lib  
vtkGenericFiltering-gd.lib  
vtkGeovis-gd.lib  
vtkGraphics-gd.lib  
vtkhdf5-gd.lib  
vtkHybrid-gd.lib  
vtkImaging-gd.lib  
vtkInfovis-gd.lib  
vtkIO-gd.lib  
vtkjpeg-gd.lib  
vtklibxml2-gd.lib  
vtkmetaio-gd.lib  
vtkNetCDF-gd.lib  
vtkNetCDF\_cxx-gd.lib  
vtkpng-gd.lib  
vtkproj4-gd.lib  
vtkRendering-gd.lib  
vtksqlite-gd.lib  
vtksys-gd.lib  
vtktiff-gd.lib  
vtkverdict-gd.lib  
vtkViews-gd.lib  
vtkVolumeRendering-gd.lib  
vtkWidgets-gd.lib  
vtkzlib-gd.lib

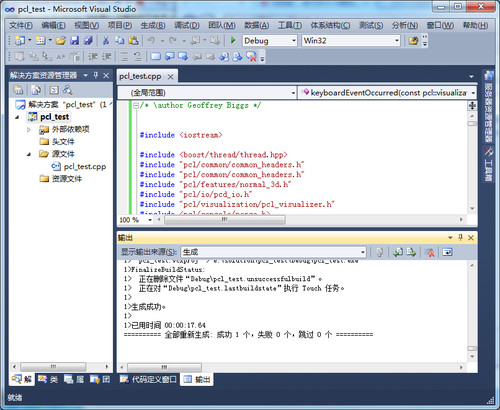
这里要注意PCL引用到的其他库文件一定要加进去，VTK的库很多，而且依赖关系我看得很乱，所以为了方便就全加进去了。

如果没有加完整，很多信息提示错误的，之前忘了加一个什么库，导致很多error LNK2019，还让我重装了很多次……

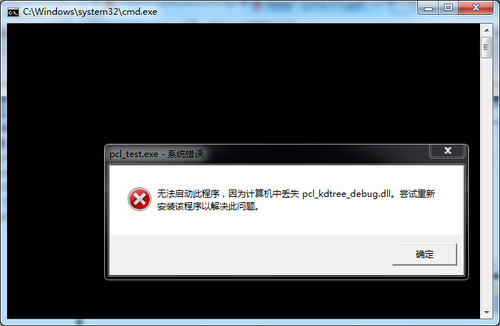
注意库要对应debug和release版，debug后面一般都有-gd或者\_debug的，看看就能分出来了。对应错了还会有error LNK：

error LNK2038: mismatch detected for '\_ITERATOR\_DEBUG\_LEVEL': value '2' doesn't match value '0' in  XXXX.......

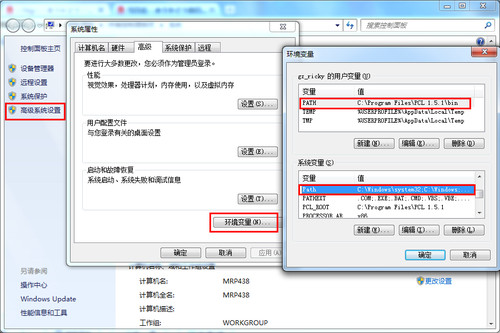
这时应该可以看到成功生成的信息了：



别着急运行，会发现还没做完的。

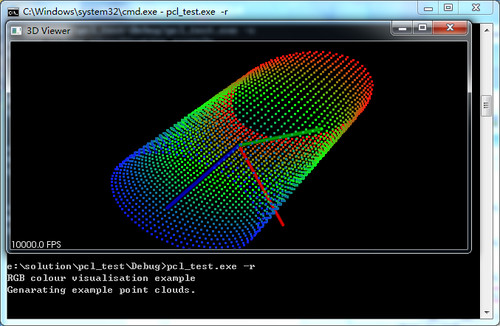


 我们还要把对应的dll加到环境中，可以直接把用到的那些dll放到应用程序搜索到的位置，如cpp对应的目录。（=\_=！好笨的方法，不过我就是这样做的……）或者是把dll的目录设置到环境中去，譬如



这样就真的能够运行了~泪奔啊，就这么几个步骤弄了半个星期，我这个SB！

运行结果



默泪~转载请注明[http://blog.163.com/gz\_ricky/blog/static/182049118201212742637279](http://blog.163.com/gz_ricky/blog/static/182049118201212742637279/)

不行的话注意一下64位的

配环境变量的话 dll不用考到项目目录了

3、配置环境变量

添加环境变量PCL\_ROOT

D:\Program Files\PCL\

系统路径path后添加：

D:\Program Files\OpenNI\Bin;%PCL\_ROOT%\bin;%PCL\_ROOT%\3rdParty\FLANN\bin;%PCL\_ROOT%\3rdParty\Qhull\bin;

总体步骤:

先装qt x64(对应版本),再装vs插件,然后按vs 的配置配置qt即可,vs插件用qt-projectsetting选择版本,option 到bin下找qmake.exe（手动）

QVTKWidget:要加QVTK.lib!!!!

若cmd退出，则是因为有中文路径！

Ctrl + R 可以预览界面！！

Pcl1.5.0没有QVTKWidget.h用的是1.6.0的，也就是交叉使用，搞清楚！

## Pcl嵌入QMainWindow中

### 整个嵌入

QVTKWidget widget;

boost::shared\_ptr<pcl::visualization::PCLVisualizer> viewer (new pcl::visualization::PCLVisualizer ("3D Viewer"));

viewer->setBackgroundColor (0, 0, 0);

// viewer->addPointCloud<pcl::PointXYZ> (cloud\_xyz, "sample cloud");

viewer->setPointCloudRenderingProperties

(pcl::visualization::PCL\_VISUALIZER\_POINT\_SIZE, 3, "sample cloud");

viewer->addCoordinateSystem (1.0);

viewer->initCameraParameters ();

vtkSmartPointer<vtkRenderWindow> renderWindow = viewer->getRenderWindow();

widget.SetRenderWindow (renderWindow);

//this->setCentralWidget(widget.topLevelWidget());

this->setCentralWidget(widget.window());

while (!viewer->wasStopped ())

{

viewer->spinOnce (100);

boost::this\_thread::sleep (boost::posix\_time::microseconds (1000));

}

### 局部嵌入

Ui上搞一个Widget，提升为QVTKWidget，填好头文件然后直接ui.qvtkwidget->setRenderingWindow

## QT PCL VS2010

### QT基本知识常用

#### 打开保存文件对话框

打开文件对话框

#include<QFileDialog>

QString pcdPath = QFileDialog::getOpenFileName(this, tr("打开数据文件"), ".", tr("PCD Files(\*.pcd)"));

if(pcdPath.length() == 0)

{

QMessageBox::information(NULL, tr("Path"), tr("未选择任何文件"));

}

打开多个文件,多选文件

QFileDialog \*fd=new QFileDialog(this);

//必须设定,不然不能多选

fd->setFileMode(QFileDialog::ExistingFiles);

if(fd->exec()==QFileDialog::Accepted)

{

QStringList files=fd->selectedFiles();

//迭代器的用法

for(QStringList::Iterator it =files.begin();it!=files.end();it++)

QMessageBox::information(NULL,tr("Path"),\*it);

}

#### 类型转换

int string QString转换

QString s2q(const string &s)

{

return QString(QString::fromLocal8Bit(s.c\_str()));

}

string q2s(const QString &s)

{

return string((const char \*)s.toLocal8Bit());

}

QString::number(int or float)

#### 全局函数调用对话框

**最近做VC时，需要用到多线程。在网上基于对话框的MFC例程中，网上所有例程均将线程函数定义为全局函数，而我在全局函数中需要用到对话框Dlg类成员函数，此时使用This指针。**

**1）先申请个全局void类型指针： void \*pdlg;**

**2）在dlg InitDialog中，将该void指针初始化为this： pdlg= this;**

**3）在全局函数中加入以下程序**

**CTestDlg \*dlg; //将void类型强制转换成dlg类型**

**dlg = (CTestDlg \*)pdlg;**

**这样在全局函数中就可以使用dlg->指针来使用其变量及函数了！**

#### 状态栏

ui.statusBar->showMessage(QString::number(cloud->at(0).x)+" "+QString::number(cloud->at(0).y)+" "+QString::number(cloud->at(0).z));

#### 中文乱码问题

也许是QTextCodec::setCodecForTr(QTextCodec::codecForLocale());

也许是QTextCodec::setCodecForTr(QTextCodec::codecForName("UTF-8"));自己看着办

### 点云相关

访问并改变数据cloud->points[i].z=4;

简化定义:typedef pcl::PointCloud<pcl::PointXYZ> xlfdPointCloud;

点云个数:cloud->points.size();

Pcl快捷键 g r u….

坐标轴与颜色的关系:红x,绿y,蓝z

Int -> string

char buf[10];

sprintf(buf, "%d", 100);

string b = buf;

添加进点云之后再viewer->resetCamera();就可以自动以适当视角展示点云

### 多线程

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include <QMessageBox>

#include <QThread>

class MyThread : public QThread

{

public:

virtual void *run*();

};

void MyThread::*run*()

{

for( int count = 0; count < 20; count++ )

{

sleep( 1 );

qDebug( "Ping!" );

}

}

MainWindow::MainWindow(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::MainWindow)

{

ui->setupUi(this);

MyThread \*th=new MyThread();

th->start();

或者

MyThread th;

Th.start();

}

经典应用

#include "mainwindow.h"

#include <QPushButton>

#include <QVBoxLayout>

#include "myobject.h"

MainWindow::MainWindow(QWidget \*parent) : QMainWindow(parent) {

my = new MyObject;

firstButton = new QPushButton(tr("first"), 0);

connect(firstButton, SIGNAL(clicked()), my, SLOT(first()), Qt::QueuedConnection);

secondButton = new QPushButton(tr("second"), 0);

connect(secondButton, SIGNAL(clicked()), my, SLOT(second()), Qt::QueuedConnection);

threeButton = new QPushButton(tr("three"), 0);

connect(threeButton, SIGNAL(clicked()), my, SLOT(three()), Qt::QueuedConnection);

selfButton = new QPushButton(tr("self"), 0);

connect(selfButton, SIGNAL(clicked()), this, SLOT(onSelfPushed()));

exitButton = new QPushButton(tr("exit"), 0);

connect(exitButton, SIGNAL(clicked()), this, SLOT(onExitPushed()));

QVBoxLayout \*layout = new QVBoxLayout;

layout->addWidget(firstButton);

layout->addWidget(secondButton);

layout->addWidget(threeButton);

layout->addWidget(selfButton);

layout->addWidget(exitButton);

QWidget \*p = new QWidget;

p->setLayout(layout);

QThread \*thread = new QThread;

my->moveToThread(thread);

thread->start();

connect(thread, SIGNAL(started()), my, SLOT(first()));

setCentralWidget(p);

}

MainWindow::~MainWindow() {

}

void MainWindow::onFirstPushed() {

my->first();

}

void MainWindow::onSecondPushed() {

my->second();

}

void MainWindow::onThreePushed() {

my->three();

}

void MainWindow::onSelfPushed() {

qDebug() << QThread::currentThreadId();

}

void MainWindow::onExitPushed() {

close();

}

### 必须注意的东西

新建线程初始化内不能用pclvisualizer！必须在run内初始！

## QTCreator PCL环境搭建(速度比vs快的多!)

### 安装

vs2010—编译器,必备

pcl1.6.0all in one x64,编译好的vs2010 包

qt4.8.0x64---也就是pcl官网上下下来的

Qtcreator---配置编译工具选vs2010 中是三角叹号的编译器,其它的不行,本电脑amd64

中文路径,否则出错

### 配置

.pro配置

# PCL

INCLUDEPATH += "C:/Program Files/PCL 1.6.0/include/pcl-1.6/"

INCLUDEPATH += "C:/Program Files/PCL 1.6.0/3rdParty/Eigen/include/"

INCLUDEPATH += "C:/Program Files/PCL 1.6.0/3rdParty/Boost/include/"

INCLUDEPATH += "C:/Program Files/PCL 1.6.0/3rdParty/FLANN/include/"

INCLUDEPATH += "C:/Program Files/PCL 1.6.0/3rdParty/Qhull/include/"

INCLUDEPATH += "C:/Program Files/PCL 1.6.0/3rdParty/VTK/include/vtk-5.8/"

INCLUDEPATH += "C:/Program Files/OpenNI/Include"

LIBS += -L"C:/Program Files/PCL 1.6.0/3rdParty/Boost/lib/"

LIBS += -L"C:/Program Files/PCL 1.6.0/3rdParty/Flann/lib/"

LIBS += -L"C:/Program Files/PCL 1.6.0/3rdParty/Qhull/lib/"

LIBS += -L"C:/Program Files/PCL 1.6.0/3rdParty/VTK/lib/vtk-5.8/"

LIBS += -L"C:/Program Files/PCL 1.6.0/lib/"

LIBS += -L"C:/Program Files/OpenNI/Lib64/"

#解决 无法解析的外部符号 vtkcommon.lib vtkRendering的问题！

LIBS += opengl32.lib

LIBS += user32.lib

LIBS += Advapi32.lib

LIBS += gdi32.lib

LIBS += pcl\_kdtree\_release.lib

LIBS += pcl\_surface\_release.lib

LIBS += pcl\_io\_release.lib

LIBS += pcl\_io\_ply\_release.lib

LIBS += pcl\_search\_release.lib

LIBS += pcl\_segmentation\_release.lib

LIBS += pcl\_apps\_release.lib

LIBS += pcl\_features\_release.lib

LIBS += pcl\_filters\_release.lib

LIBS += pcl\_visualization\_release.lib

LIBS += pcl\_common\_release.lib

LIBS += flann\_cpp\_s.lib

LIBS += libboost\_system-vc100-mt-1\_47.lib

LIBS += libboost\_filesystem-vc100-mt-1\_47.lib

LIBS += libboost\_thread-vc100-mt-1\_47.lib

LIBS += libboost\_date\_time-vc100-mt-1\_47.lib

LIBS += libboost\_iostreams-vc100-mt-1\_47.lib

LIBS += openNI.lib

LIBS += vtkalglib.lib

LIBS += vtkCharts.lib

LIBS += vtkCommon.lib

LIBS += vtkDICOMParser.lib

LIBS += vtkexoIIc.lib

LIBS += vtkexpat.lib

LIBS += vtkFiltering.lib

LIBS += vtkfreetype.lib

LIBS += vtkftgl.lib

LIBS += vtkGenericFiltering.lib

LIBS += vtkGeovis.lib

LIBS += vtkGraphics.lib

LIBS += vtkhdf5.lib

LIBS += vtkHybrid.lib

LIBS += vtkImaging.lib

LIBS += vtkInfovis.lib

LIBS += vtkIO.lib

LIBS += vtkjpeg.lib

LIBS += vtklibxml2.lib

LIBS += vtkmetaio.lib

LIBS += vtkNetCDF.lib

LIBS += vtkNetCDF\_cxx.lib

LIBS += vtkpng.lib

LIBS += vtkproj4.lib

LIBS += vtkRendering.lib

LIBS += vtksqlite.lib

LIBS += vtksys.lib

LIBS += vtktiff.lib

LIBS += vtkverdict.lib

LIBS += vtkViews.lib

LIBS += vtkVolumeRendering.lib

LIBS += vtkWidgets.lib

LIBS += vtkzlib.lib

LIBS += QVTK.lib

### 错误

correspondence 类似的错误:

一定要把pcl相关.h放在Q开头的前边,不然怎么都错