VTK学习（十五）封闭性检测

原创 yuyangyg 2017-11-14 20:53:01 1193 收藏 版权

分类专栏：VTK

```
#include "vtkAutoInit.h"
VTK_MODULE_INIT(vtkRenderingOpenGL2); // VTK was built with vtkRenderingOpenGL2
VTK_MODULE_INIT(vtkInteractionStyle);

#include <vtkSmartPointer.h>
#include <vtkSelectionNode.h>
#include <vtkInformation.h>
#include <vtkUnstructuredGrid.h>
#include <vtkPolyData.h>
#include <vtkPolyDataNormals.h>
#include <vtkPointData.h>
#include <vtkSTLPolyDataReader.h>
#include <vtkRenderWindow.h>
#include <vtkRenderWindowInteractor.h>
#include <vtkRenderer.h>
#include <vtkSelection.h>
#include <vtkSelectionNode.h>
#include <vtkSphereSource.h>
#include <vtkPolyDataMapper.h>
#include <vtkActor.h>
#include <vtkCamera.h>
#include <vtkProperty.h>
#include <vtkIdTypeArray.h>
#include <vtkExtractSelection.h>
#include <vtkDataSetSurfaceFilter.h>
#include <vtkFeatureEdges.h>
#include <vtkFillholesFilter.h>

void GenerateData(vtkSmartPointer<vtkPolyData> input)
{
    vtkSmartPointer<vtkSphereSource> sphereSource =
        vtkSmartPointer<vtkSphereSource>::New();
    sphereSource->Update();

    idx->SetNumberOfComponents(1);
    idx->InsertNextValue(10);
    idx->InsertNextValue(20);

    vtkSmartPointer<vtkSelectionNode> selectionNode =
        vtkSmartPointer<vtkSelectionNode>::New();
    selectionNode->SetFieldType(vtkSelectionNode::CELL);
    selectionNode->SetContentType(vtkSelectionNode::INDICES);
    selectionNode->SetSelectionList(idx);
    selectionNode->GetProperties()->Set(vtkSelectionNode::INVERSE(), 1);

    vtkSmartPointer<vtkSelection> selection =
        vtkSmartPointer<vtkSelection>::New();
    selection->AddNode(selectionNode);

    vtkSmartPointer<vtkExtractSelection> extractSelection =
        vtkSmartPointer<vtkExtractSelection>::New();
    extractSelection->SetInputData(0, sphereSource->GetOutput());
    extractSelection->SetInputData(1, selection);
    extractSelection->Update();

    vtkSmartPointer<vtkDataSetSurfaceFilter> surfaceFilter =
        vtkSmartPointer<vtkDataSetSurfaceFilter>::New();
    surfaceFilter->SetInputConnection(extractSelection->GetOutputPort());
    surfaceFilter->Update();

    input->ShallowCopy(surfaceFilter->GetOutput());
}

int main(int argc, char *argv[])
{
    vtkSmartPointer<vtkPolyData> input =
        vtkSmartPointer<vtkPolyData>::New();
    GenerateData(input);

    vtkSmartPointer<vtkFeatureEdges> featureEdges =
        vtkSmartPointer<vtkFeatureEdges>::New();
    featureEdges->SetInputData(input);
    featureEdges->BoundaryEdgesOn();
    featureEdges->FeatureEdgesOff();
    featureEdges->ManifoldEdgesOff();
    featureEdges->NonManifoldEdgesOff();
    featureEdges->Update();

    int numberOfOpenEdges = featureEdges->GetOutput()->GetNumberOfCells();
    if (numberOfOpenEdges)
    {
        std::cout << "这网格模型不是封闭的..." << std::endl;
    }
    else
    {
        std::cout << "这网格模型是封闭的..." << std::endl;
        return EXIT_SUCCESS;
    }

    vtkSmartPointer<vtkFillholesFilter> fillholesFilter =
        vtkSmartPointer<vtkFillholesFilter>::New();
    fillholesFilter->SetInputConnection(input);
    fillholesFilter->Update();

    vtkSmartPointer<vtkPolyDataNormals> normals =
        vtkSmartPointer<vtkPolyDataNormals>::New();
    normals->SetInputConnection(fillholesFilter->GetOutputPort());
    normals->ConsistencyOn();
    normals->SplittingOff();
    normals->Update();

    //
    double leftViewport[4] = {0.0, 0.0, 0.5, 1.0};
    double rightViewport[4] = {0.5, 0.0, 1.0, 1.0};

    vtkSmartPointer<vtkPolyDataMapper> originalMapper =
        vtkSmartPointer<vtkPolyDataMapper>::New();
    originalMapper->SetInputData(input);

    vtkSmartPointer<vtkProperty> backfaceProp =
        vtkSmartPointer<vtkProperty>::New();
    backfaceProp->SetDiffuseColor(0.89, 0.81, 0.34);

    vtkSmartPointer<vtkActor> originalActor =
        vtkSmartPointer<vtkActor>::New();
    originalActor->SetMapper(originalMapper);
    originalActor->SetBackfaceProperty(backfaceProp);
    originalActor->GetProperty()->SetDiffuseColor(1.0, 0.3882, 0.2784);

    vtkSmartPointer<vtkPolyDataMapper> edgeMapper =
        vtkSmartPointer<vtkPolyDataMapper>::New();
    edgeMapper->SetInputData(featureEdges->GetOutput());
    vtkSmartPointer<vtkActor> edgeActor =
        vtkSmartPointer<vtkActor>::New();
    edgeActor->SetMapper(edgeMapper);
    edgeActor->GetProperty()->SetEdgeColor(0.0, 0.0, 1.0);
    edgeActor->GetProperty()->SetEdgeVisibility(1);
    edgeActor->GetProperty()->SetLineWidth(5);

    vtkSmartPointer<vtkPolyDataMapper> filledMapper =
        vtkSmartPointer<vtkPolyDataMapper>::New();
    filledMapper->SetInputConnection(normals->GetOutputPort());

    vtkSmartPointer<vtkActor> filledActor =
        vtkSmartPointer<vtkActor>::New();
    filledActor->SetMapper(filledMapper);
    filledActor->GetProperty()->SetDiffuseColor(1.0, 0.3882, 0.2784);

    vtkSmartPointer<vtkRenderer> leftRenderer =
        vtkSmartPointer<vtkRenderer>::New();
    leftRenderer->SetViewport(leftViewport);
    leftRenderer->AddActor(originalActor);
    leftRenderer->AddActor(edgeActor);
    leftRenderer->SetBackground(1.0, 1.0, 1.0);

    vtkSmartPointer<vtkRenderer> rightRenderer =
        vtkSmartPointer<vtkRenderer>::New();
    rightRenderer->SetViewport(rightViewport);
    rightRenderer->AddActor(filledActor);
    rightRenderer->SetBackground(1.0, 1.0, 1.0);

    vtkSmartPointer<vtkRenderWindow> renderWindow =
        vtkSmartPointer<vtkRenderWindow>::New();
    renderWindow->AddRenderer(leftRenderer);
    renderWindow->AddRenderer(rightRenderer);
    renderWindow->SetSize(640, 320);
    renderWindow->Render();
    renderWindow->SetWindowName("PolyDataClosed");

    vtkSmartPointer<vtkRenderWindowInteractor> renderWindowInteractor =
        vtkSmartPointer<vtkRenderWindowInteractor>::New();
    renderWindowInteractor->SetRenderWindow(renderWindow);

    leftRenderer->GetActiveCamera()->SetPosition(0, -1, 0);
    leftRenderer->GetActiveCamera()->SetFocalPoint(0, 0, 0);
    leftRenderer->GetActiveCamera()->SetViewUp(0, 0, 1);
    leftRenderer->GetActiveCamera()->Azimuth(30);
    leftRenderer->GetActiveCamera()->Elevation(30);
    leftRenderer->ResetCamera();
    rightRenderer->GetActiveCamera[leftRenderer->GetActiveCamera()];

    renderWindow->Render();
    renderWindowInteractor->Start();

    return EXIT_SUCCESS;
}
```

如果一个边只被一个多边形包含，那么这条边就是边界边。是否存在边界边是检测一个网格模型是否封闭的重要特征。

vtkFeatureEdges是一个非常重要的类，该类能够提取多边形网格模型中四种类型的边

1.边界边，就是只被一个多边形包含的边

2.非流形边，被三个或者三个以上的多边形包含的边即为四种类型的边

3.特征边，当两点同一条边的两个三角形法向量的夹角大于该阈值时，即为特征边。

4.流行边，只为两个多边形包含的边

```
    vtkSmartPointer<vtkFeatureEdges> featureEdges =
        vtkSmartPointer<vtkFeatureEdges>::New();
    featureEdges->SetInputData(input);
    featureEdges->BoundaryEdgesOn();
    featureEdges->FeatureEdgesOff();
    featureEdges->ManifoldEdgesOff();
    featureEdges->NonManifoldEdgesOff();
    featureEdges->Update();
```

可以通过边界边的数目来确定网格是否封闭

```
    int numberOfOpenEdges = featureEdges->GetOutput()->GetNumberOfCells();
```

可以用vtkFillHolesFilter来进行漏洞补，首先检测出网格中所有的边界边，然后找出这些边界边的每一个边的四周，最后用这些四周进行三角化，实现填补目的。

```
    vtkSmartPointer<vtkPolyDataNormals> normals =
        vtkSmartPointer<vtkPolyDataNormals>::New();
    normals->SetInputConnection(fillHolesFilter->GetOutputPort());
    normals->ConsistencyOn();
    normals->SplittingOff();
    normals->Update();
```

VTK入门（三）—图像加工处理（一）
acheo501913的博客 1.7k+
不知不觉离上篇博客已经过去了一个月了，决定把vtk图像处理的方式方法总结下来，编写vtk程序就和平时做事情一样，...
VTK学习（五）
qq_35424261的博客 198
1 vtkPolyData 8 vtkImageData vtkDataObject->vtkDataSet->vtkRectilinearGrid
1vtl...
发表评论 内容不能为空 发表
评论

相关推荐

VTK学习笔记2.1-3
陶 1423
整理于 https://www.course163.org/course/BIT-1001877001（Python语言计算三维可视化）VTK可视化实例 知...
图形处理之拓扑检测
时代小猫仔的专栏 1487
本文介绍VTK中拓扑检测，基本概念 介绍拓扑检测之前，介绍一些基本概念。如果一个网格模型中，一条...
VTK学习（五）
1 vtkPolyData 8 vtkImageData vtkDataObject->vtkDataSet->vtkRectilinearGrid
VTK学习-VTK管线
zy23178780的博客 608
VTK学习-VTK管线 写在前面：希望通过这篇博客，对中小学生（VTK初学者）中国最的原理。
VTK必要头文件
Sua83的博客 518
#include <vtkAutoInit.h> VTK_MODULE_INIT(vtkRenderingOpenGL2) VTK_MODULE_INIT(vtkInteractionStyle);...
VTK修炼之路84：图形基本操作进阶_表面重建技术（三角剖分）
江表知识博客 6628
1.表面重建（表面三角剖分）表面重建是指根据点云数据的空间坐标，将点云表达为三角形面片，...
VTK 3.0升级到VTK7.0.0要做的事情
OneSeal的专栏 8313
1 背景 之前使用VTK 3.0编译时到现在了一些bug，最近官网VTK-7.0.0已经发布，期待着自某的版本中会解决...
VTK学习-vtkCamera
zy23178780的博客 935
VTK学习-vtkCamera
VTK 读取STL文件并且显示
xiaoh的专栏 7474
#include "stdafx.h" #include "vtkCamera.h" #include "vtkGenericRenderWindowInteractor.h" #include "vtkInteract...
VTK学习（七）坐标系统和空间变换
yuyangyg的博客 2280
#include #include #include #include #include #include #include #include #include #include #include #include VTK_MODUL...
初次学习VTK遇到的一些坑（适用于初学者）
greg的博客 144
bug1：所有图形对象的实现类都是从vtkProp3DFollowerRayCastCompositeFunctionEvxbVolumeRayCastMapper都派生自VTK的...
VTK加载OBJ、STL格式的模型并显示图片功能
HXxiyingj的博客 485
#include <QCoreApplication> #include <vtkPolyDataReader.h> #include <vtkOBJReader.h> #include "vtk...
VTK修炼之路14：图像处理_创建
江表知识博客 3571
1 可以在图像数据的每一个体素（像素）上执行任何操作。图像主要由两部分组成：图像主要由两部分组成：图像主要由两部分组成：图像主要由两部分组成。...
VTK编译时出错，无法解析外部符号
berlioxian的博客 4286
今天配置VTK，通过了不少麻烦才在一一解决了，到中一个是在编译下面两个时出现的： error LNK2001: unresolved ...
VTK交互系统 3 自定义交互器样式
Dezeming的博客 3379
我现阶段主要从事图像大变形一个交互器样式比较好，而不是直接对上一大堆程序，可是一开始有不能或不知道了...
©2020 CSDN 皮肤主题：编辑工具室 设计师:CSDN官方博客 版权投诉