# 目录

[目录 1](#_Toc3403199)

[陶瓷图像的分割 2](#_Toc3403200)

[1. 方法概述 2](#_Toc3403201)

[2. 工程包含文件 2](#_Toc3403202)

[3. 本程序运行环境和所需库函数 3](#_Toc3403203)

[4. 算法流程图 4](#_Toc3403204)

[5. 代码详解 4](#_Toc3403205)

[6. 软件介绍 10](#_Toc3403206)

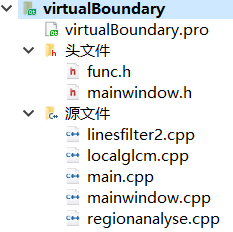
# 陶瓷图像的分割

## 方法概述

本分割方法主要针对具有虚拟边界的材料图像，进行虚拟边界检测并依据边界分割图像。方法主要分为两个部分：第一部分主要是利用膨胀、glcm方法填充图像中较细小的间隙后，利用霍夫变换检测虚拟边界，并根据边界对图像进行划分。第二部分主要是对第一部分划分得到的区域中的微观结构进一步聚类，得到不同类别的微观结构。

## 2. 工程包含文件

**virtualBoundary（项目）**



1. virtualBoundary.pro是整个工程的属性配置文件，需要注意配置其中opencv库的路径。
2. mainwindow.h , mainwindow.cpp是ui界面相关的源文件，包括消息响应、消息控件、按钮等，以及整个算法的主程序部分。
3. func.h是算法中用到的各种方法的声明。Linesfilter2.cpp主要包含对检测出的直线进行筛选的方法。Localglcm.cpp主要包含GLCM(灰度共生矩阵)的实现方法。Regionanalyse.cpp主要包含对微观结构进行聚类分析的方法。

## 3. 本程序运行环境和所需库函数

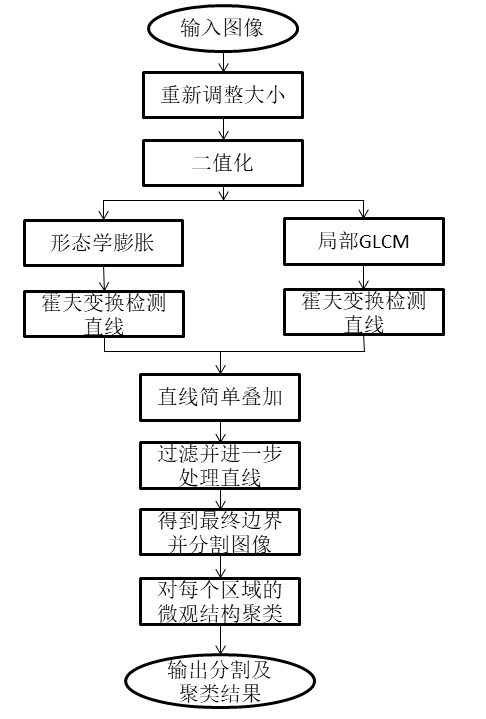
操作系统：Windows系统

运行语言：程序在QT下使用C++开发(QT版本5.8.0 MSVC2015 32bit)

程序开发使用的函数库：C++下的OpenCV2.4.9（需保证是2.4.9，更高的版本在霍夫变换上可能有差异，会导致结果有问题）, python运行环境及机器学习相关库(numpy,scikit-learn)

可执行程序运行所需环境：需安装python运行环境及机器学习相关库(numpy,scikit-learn)

## 算法流程图



## 代码详解

**5.1 主程序函数（mainwindow.cpp）**

**5.1.1主要函数**

1. void MainWindow::on\_inputImage\_clicked()

功能：按钮“打开图片”功能的实现。选择一张图片并调整显示窗口大小，将图片显示在操作窗口的旁边。

作者：赖传滨

改写：无

输入：无

返回：无。显示选择的图片。

1. void MainWindow::on\_detectBoundary\_clicked()

功能：“边界检测”功能的实现。读入输入的参数，对图片调整大小后二值化，然后分别进行膨胀操作以及局部GLCM特征提取。之后，将两个结果分别进行霍夫变换检测直线，然后将检测的直线合并在一起，对所有直线进行过滤及延长等操作，得到最终的边界。

作者：赖传滨

改写：无

输入：无，主要是自动读取由界面输入的参数 ：

l1:膨胀元素的大小

sigma:直线经过白条部分占自身的比例阈值

返回：无。显示检测到的虚拟边界。

1. void MainWindow::on\_pushButton\_cla\_clicked()

功能：对分割图片后指定的区域内的微观结构进行特征提取并聚类。

作者：赖传滨

改写：无

输入：无。主要是自动读入由界面输入的参数：

区域编号，dbscan参数eps和minpts

返回：无。指定区域的微观结构聚类结果。

**5.2 边界检测相关函数（func.h, localglcm.cpp, linesfilter2.cpp）**

**5.2.1结构体**

typedef struct ConnectedComp

{

int area; //面积

std::vector<cv::Point> pointSet; //区域内的点集

int num; //编号

}

ConnectedComp;

这个结构体主要代表的是区域块的信息

**5.2.2 GLCM相关函数（localglcm.cpp）**

1. void localGLCM(Mat image, Mat &result, float threshold, int radius)

功能：局部GLCM功能实现。将一幅图划分成n\*n的小区域，对每个区域提取4个方向的GLCM，并通过GLCM计算区域在4个方向的“对比度”，然后计算这4个对比度的方差作为特征。若该特征值大于阈值，则该区域涂黑，否则该区域涂白。输出该变换后的结果。

作者：赖传滨

改写：无

输入：image：待变换图像

result: 变换后的图像

threshold：特征阈值

radius: 划分区域的半径

返回：变换后的图像。

1. int calGLCM(CvMat\* bWavelet,int angleDirection,double\* featureVector)

功能：计算一个区域指定方向的GLCM，以及从GLCM中提取对比度特征。

作者：赖传滨

改写：无

输入：bWavelet：一个区域

angleDirection: 指定方向

featureVector: 输出的特征向量

返回：从GLCM中提取的四种特征（本方法只用了对比度这一种特征）

**5.2.3 直线过滤相关函数（linesfilter2.cpp）**

1. void linesFilter2(Mat src\_binary, vector<Vec4i>Lines, Mat &line\_img, vector<ConnectedComp> &ccompSet,int sigma\_1,double sigma\_2,int sigma\_3,double sigma\_4) {

功能：过滤通过霍夫变换检测到的直线，并对过滤后的直线进行延长，删去相交产生的线头等处理。

作者：赖传滨

改写：无

输入：src\_binary：二值化后的原图像

Lines：霍夫变换检测出的直线

line\_img：画出最后的分割线

ccompSet：根据分割线分割得到的各区域

sigma\_1：线段的距离阈值（距离小于阈值的认为是同一组）

sigma\_2：线穿过白条占线本身长度的比例的阈值，若该比例过小，则删去该线

sigma\_3：同一组线内线的数量的阈值，若该组内的线数量不足，则删去该组所有线

sigma\_4：线穿过白条占线本身长度的比例的阈值，若该比例过小，则删去该线。该阈值用于延长线时使用。

返回：分割后的各区域。

1. double computeRate(Mat src\_binary, cv::Point pt\_start, cv::Point pt\_end)

功能：计算一条线经过的白条的部分占线长的比例

作者：赖传滨

改写：无

输入：src\_binary：原图的二值图

pt\_start：线段的一个端点

pt\_end: 线段的另一个端点

返回：返回线经过的白条的部分占线长的比例

1. double computeJuLi(cv::Point itr\_start,cv::Point itr\_end,cv::Point itrZhun\_start,double itrZhunXie)

功能：计算两线段（线段a,线段b）的距离(一个线段的中点到另一个线段的距离)

作者：赖传滨

改写：无

输入：itr\_start：线段a的一个端点

itr\_end：线段a的另一个端点

itrZhun\_start: 线段b的一个端点

itrZhunXie: 线段b的另一个端点

返回：两线段（线段a,线段b）的距离

1. bool neighborXielv(Vec4i LineFinalTemp,Mat src\_binary)

功能：将某条线和其周围线对比斜率，如果相近则返回true否则返回false

作者：赖传滨

改写：无

输入：LineFinalTemp：待比较的一条线

src\_binary：原图的二值图

返回：某条线是否和其周围线的斜率相近

1. int\* deleteXianDuan(vector<Vec4i>& LinesFinal,int rows,int cols)

功能：对过滤后的直线，若有直线相交的情况，删去较短的线头

作者：赖传滨

改写：无

输入：LinesFinal：过滤后的直线

Rows，cols：原图的行数和列数

返回： flag，指明删去线头后交点的情况，0代表两端都没有交点；1代表start点是交点；2代表end点是交点

1. void connectXianDuan1(vector<Vec4i>& LinesFinal,int rows,int cols,int\* flag,Mat& tempImg,vector<Vec4i>& LinesFinalFinal,Mat src\_binary,double rateThre)

功能：对过滤后的直线进行延长。第一次延长，先延长两端都没有交点的直线

作者：赖传滨

改写：无

输入：LinesFinal：过滤后的直线

Rows，cols：原图的行数和列数

Flag: 指明直线是否有交点，0代表两端都没有交点；1代表start点是交点；2代表end点是交点

tempImg：直线图

LinesFinalFinal：存储延长后的直线

src\_binary：原图的二值图

rateThre：线穿过白条占线本身长度的比例的阈值，若该比例过小，则删去该线。该阈值用于延长线时使用。（值为sigma\_4）

返回：返回延长后的直线

1. void connectXianDuan2(vector<Vec4i>& LinesFinal,int rows,int cols,int\* flag,Mat& tempImg,vector<Vec4i>& LinesFinalFinal,Mat src\_binary,double rateThre)

功能：对第一次延长直线的结果再进行延长，这次延长两端有交点的直线

作者：赖传滨

改写：无

输入：LinesFinal：过滤后的直线

Rows，cols：原图的行数和列数

Flag: 指明直线是否有交点，0代表两端都没有交点；1代表start点是交点；2代表end点是交点

tempImg：直线图

LinesFinalFinal：存储延长后的直线

src\_binary：原图的二值图

rateThre：线穿过白条占线本身长度的比例的阈值，若该比例过小，则删去该线。该阈值用于延长线时使用。（值为sigma\_4）

返回：返回延长后的直线

1. int newfloodfill(Mat &\_image, Mat &\_mask,Point seedPoint,int diff, int range\_area, vector<ConnectedComp> &ccompSet)

功能：种子填充，用于提取分割后的区域

作者：赖传滨

改写：无

输入：\_image：原图

\_mask：掩码，指示像素是否被访问过

seedPoint: 种子点

diff：像素差值的阈值

range\_area：区域面积阈值

ccompSet：存储分割后的区域

返回：返回区域的面积以及分割后的区域ccompSet

**5.3 聚类相关函数与脚本（regionanalyse.cpp, dbscan.py）**

**5.3.1 regionanalyse.cpp 函数说明**

1. Size regionAnalyser(Mat& src,vector<ConnectedComp> &ccompSet,Mat& src\_small,std::vector<std::vector<std::vector<cv::Point>>>& contoursRes)

说明: 对所有分割后的区域进行处理，提取各区域微观结构的特征，并将特征值写入bars\_feature.txt

作者：赖传滨

改写：无

输入：

src: 原图

ccompSet: 区域集

src\_small: 缩小后的原图

contoursRes：存储每个微观结构的轮廓

输出：缩放后图像的大小（无用）

1. void computeWhitebars(std::vector<std::vector<cv::Point>>& contours,int flag,vector<barFeatures>& barsFeatures,Mat& region)

说明: 计算每个微观结构的特征，主要有长短轴之比以及长轴方向

作者：赖传滨

改写：无

输入：

contours: 某个区域所有微观结构的轮廓

flag: 记录是第几个区域

barsFeatures: 存储特征

region：某个区域的图像

输出：无

1. void singleRegionAnalyser(int nReg,double eps,int minPts,std::vector<std::vector<std::vector<cv::Point>>>& contoursRes,Size regSize,cv::Mat& result)

说明: 将dbscan参数写入“results\\parameters.txt”，并对指定的区域进行微观结构聚类分析

作者：赖传滨

改写：无

输入：

nReg: 区域编号

eps,minpts: dbscan参数

contoursRes: 所有微观结构的轮廓

regSize：无用

result:分类结果图，不同类别微观元素用不同颜色画出

输出：分类结果图

**5.3.2 脚本解释**

C++代码中是用system("python dbscan.py")来调用脚本的

1. dbscan.py

说明: 读取results\\parameters.txt中有关dbscan参数的设置, bars\_feature.txt文件中有微观结构的特征值，对特征值归一化和聚类，输出聚类结果

作者：赖传滨

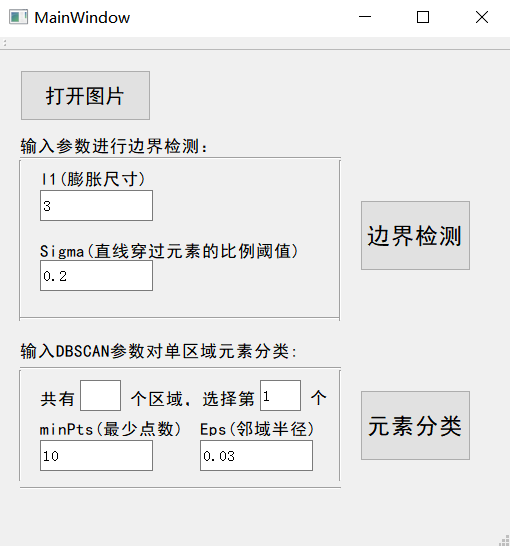
改写：无

输出：results\\label%d.txt文件（第%d个区域内每个微观结构的类别标签）

Results\\nLabels%d.txt文件（第%d个区域内微观结构的总类数）

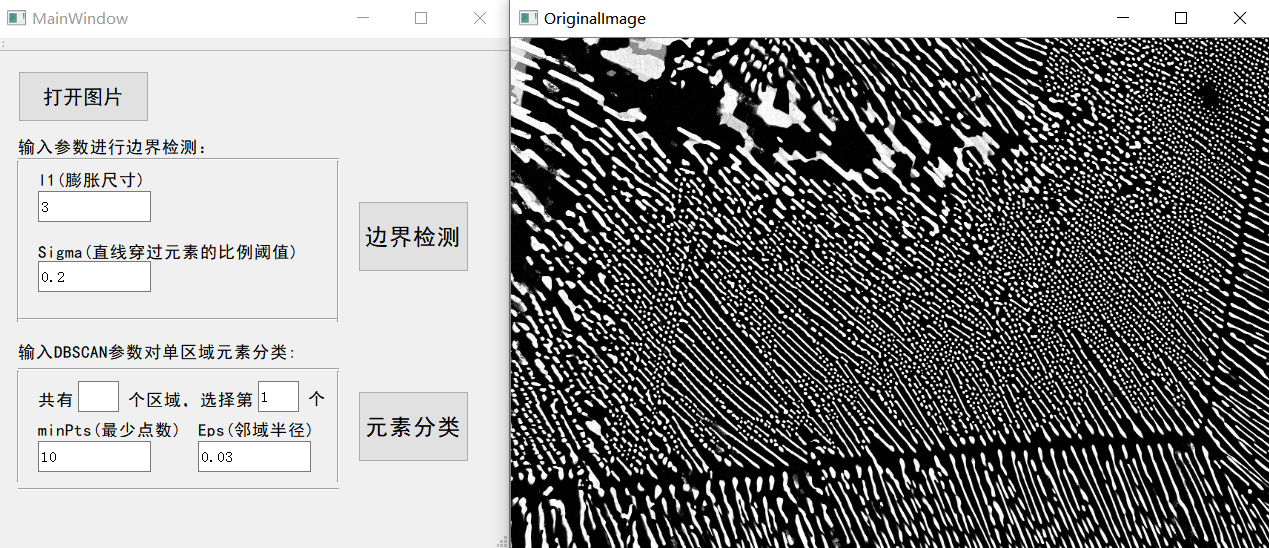
## 软件介绍

双击virtualBoundary.exe运行程序：

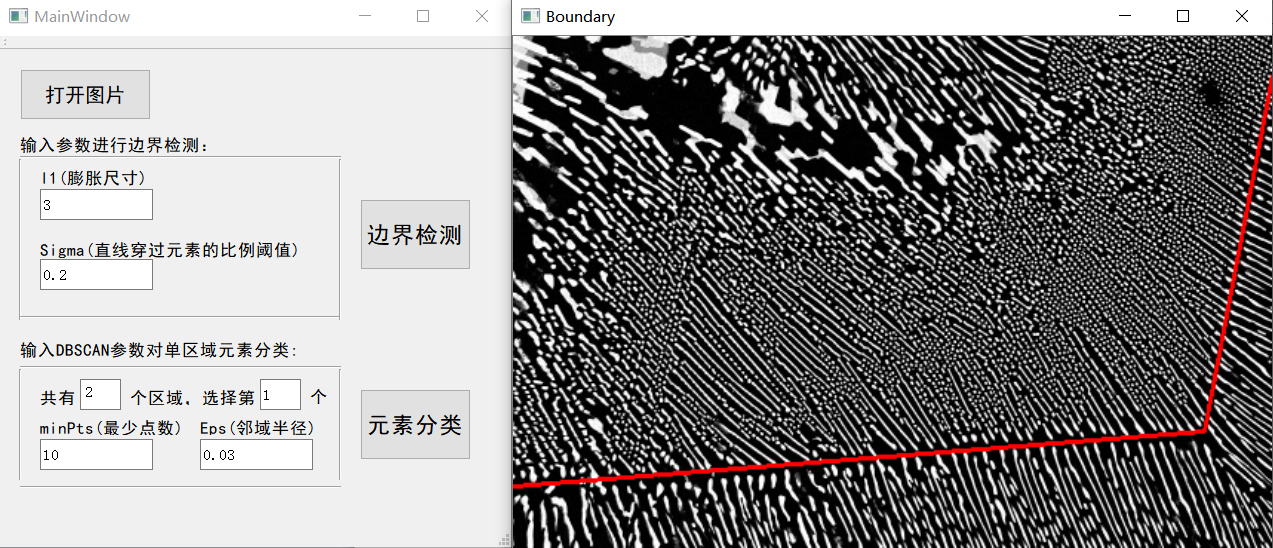


1. **软件各个功能**
2. **打开图片**：选择需要分割的图片
3. **输入参数进行边界检测:** 此部分有2个参数需要设定，l1和sigma，这两个参数都有默认值。
4. **输入DBSCAN参数对单区域元素分类:**此部分需设定要分析第几个区域，以及dbscan的两个参数eps和minpts
5. **边界检测:** 点击此按钮会执行边界检测程序并显示检测结果
6. **元素分类:** 点击此按钮，会执行聚类分析程序并显示分析结果
7. **软件执行流程**

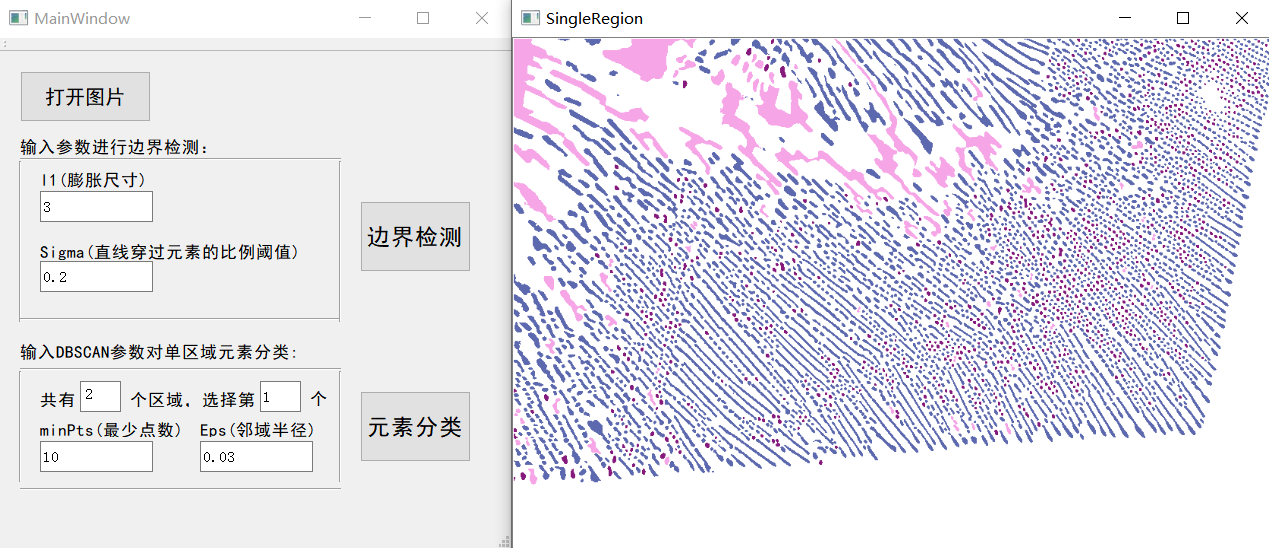
先点击**打开图片**按钮，选择需要分割的图片，选择的图片会显示在窗口右侧。然后设置参数，参数都有默认值。需注意，本程序在聚类过程中调用了python脚本，所以需要安装有python运行环境以及相关库(numpy,scikit-learn)。



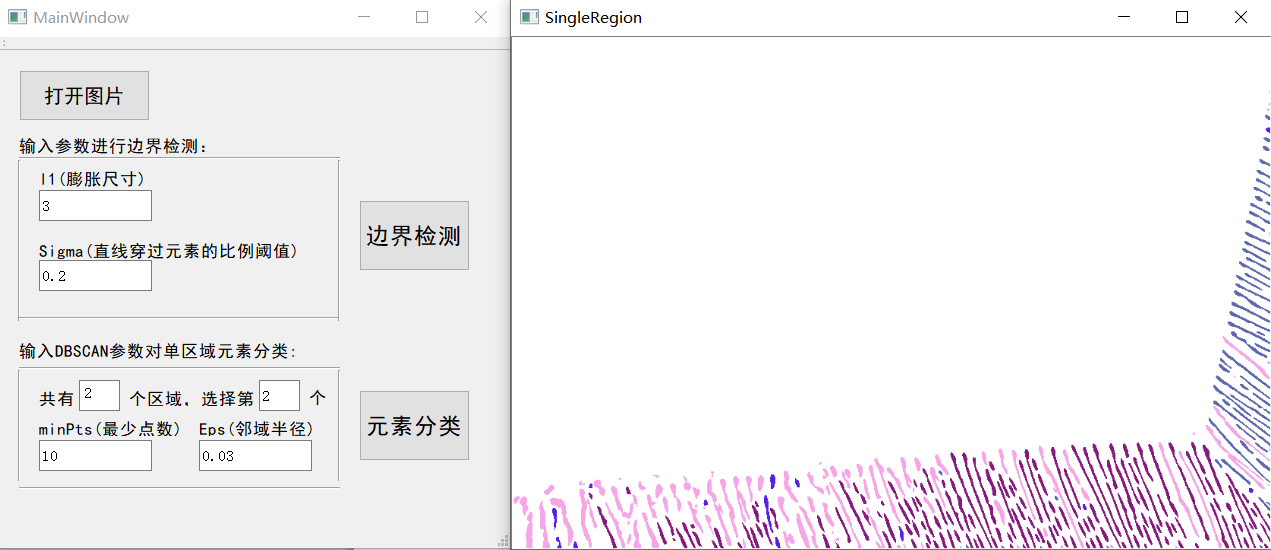
点击“边界检测”，程序检测图中的虚拟边界并显示结果：



检测虚拟边界后，程序会显示共有几个区域，然后输入需要进行聚类分析的区域编号，点击“元素分类”对该区域内的微观结构进行聚类，比如此例中共有2个区域，可分别对这两个区域进行聚类：



区域1聚类结果



区域2聚类结果

边界检测结果以及聚类结果的好坏取决于参数的设置，边界结果以及聚类结果都会保存在results目录下，边界检查结果的文件名为“finallines.png”。聚类结果的文件名则为“classRes-0.png”、“classRes-1.png”以此类推，一张图代表一个区域的聚类结果。

另外还会生成一些txt格式的中间结果，这些txt文件主要是微观结构的特征以及python脚本输出的标签。

