技术解析：

1. 边缘检测：

这一步我们采用canny算子边缘检测的方法：可以直接调用opencv中的Canny函数

1. 设定canny函数参数

函数原型：

void cvCanny

(

const CvArr\* image

CvArr\* edges

double threshold1

double threshold2

int aperture\_size=3

);

·image 输入图像，这个必须是单通道的，即灰度图

·edges 输出的边缘图像 ，也是单通道的，但是是黑白的

·threshold1 第一个阈值

·threshold2 第二个阈值

低于阈值1的像素点会被认为不是边缘；

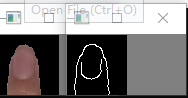
高于阈值2的像素点会被认为是边缘；

在阈值1和阈值2之间的像素点,若与第2步得到的边缘像素点相邻，则被认为是边缘，否则被认为不是边缘。

·aperture\_size Sobel 算子内核大小

默认为3

1. 调用Canny函数
2. 得到处理结果：



1. 形态学处理

这一步我们观察提取出来的指甲轮廓形状，选择合适的形态学滤波方法来更好地提取指甲部分。

我们熟悉的形态学滤波方法有 腐蚀，膨胀，均值滤波。这些单独的操作明显无法达到效果

于是这一步我们采用更高级的形态学变换：闭操作。在opencv中可以直接调用morphologyEx函数。

1. 设定morphologyEx函数参数

函数原型：

void morphologyEx

(

InputArray src

OutputArray dst

int op

InputArray kernel

Point anchor=Point(-1,-1)

intiterations=1

int borderType=BORDER\_CONSTANT

const Scalar& borderValue=morphologyDefaultBorderValue()

)

·第一个参数，InputArray类型的src，输入图像。

·第二个参数，OutputArray类型的dst，输出图像。

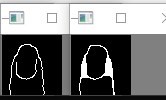
·第三个参数，int类型的op，表示形态学运算的类型，这里选用：

MORPH\_CLOSE – 闭运算（Closing operation）

·第四个参数：InputArray类型的kernel，形态学运算的内核。我们一般使用函数 getStructuringElement配合这个参数的使用。getStructuringElement函数会返回指定形状和尺寸的结构元素（内核矩阵）。

·其他参数都可以省略，设为默认。

1. 调用morphologyEx函数
2. 结果图：



1. 绘制轮廓

这一步我们用到两个函数：findContours和drawContours

1. 调用findContours函数找出所有轮廓：

函数原型：void findContours//提取轮廓，用于提取图像的轮廓

(

InputOutputArray image,

OutputArrayOfArrays contours,

OutputArray hierarchy,

int mode,

int method,

Point offset = Point()

)

·第一个参数：image，要处理的图像。

·第二个参数：contours，是一个向量，里面每个元素保存了一组由连续的Point点构成的点的集合的向量，每一组Point点集就是一个轮廓。

·第三个参数：hierarchy，可选的输出向量，包含图像的拓扑信息。

·第四个参数：int型的mode，定义轮廓的检索模式：

这里选用：CV\_RETR\_TREE， 检测所有轮廓，所有轮廓建立一个等级树结构。外层轮廓包含内层轮廓，内层轮廓还可以继续包含内嵌轮廓。

·第五个参数：int型的method，定义轮廓的近似方法：

这里选用:CV\_CHAIN\_APPROX\_SIMPLE 仅保存轮廓的拐点信息，把所有轮廓拐点处的点保存入contours向量内，拐点与拐点之间直线段上的信息点不予保留

·第六个参数：Point偏移量。

这里偏移量为0即可

1. 调用drawContours函数绘制所有轮廓：

函数原型：

void drawContours//绘制轮廓，用于绘制找到的图像轮廓

(

 InputOutputArray image,

 InputArrayOfArrays contours,

 int contourIdx,

 const Scalar& color,

 int thickness = 1,

 int lineType = 8,

 InputArray hierarchy = noArray(),

 int maxLevel = INT\_MAX,

 Point offset = Point()

)

·第一个参数：image是要绘制轮廓的图像

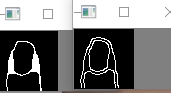
·第二个参数：contours，所有输入的轮廓

·第三个参数：contourIdx，指定要绘制轮廓的编号，如果是负数，则绘制所有的轮廓

·第四个参宿：color，绘制轮廓所用的颜色

·其他参数均可以省略，设为默认值。

1. 结果图：

绘制全部轮廓：

在drawContours的结尾加上参数：CV\_FILLED，选择目标轮廓编号，即可填充轮廓。

