**opencv定位二维码，大家伙一起来改进啊**

（https://blog.csdn.net/guanyonglai/article/details/59644437）

阅读数：2200

    虽然挺简单一个程序，，这几天断断续续一直在搞这个，有点无语了，还好遇到的几个问题再opencv论坛提问都被解决了，尤其要感谢smallyouyou童鞋，好像是台湾那边的，太暖的太热心了，，中间遇到几个小问题

1、轮廓的层级关系傻傻弄不清楚

2、Point point1=(centerx,centery);  应该改成  Point point1 = Point(centerx,centery); 这个错误虽然有点，，，但是我真的搞了一整天，，好想哭，最后还是悠悠同学帮我改的，我就说返回的时候怎么只有x坐标值，y坐标值为什么一直都是0,

3、一个轮廓的父级轮廓该怎么表示，还是没太清楚，这个的好好看看，据说这里有解释，http://docs.opencv.org/3.1.0/d9/d8b/tutorial\_py\_contours\_hierarchy.html

好啦，直接上代码吧

**[cpp]** [view plain](https://blog.csdn.net/guanyonglai/article/details/59644437) [copy](https://blog.csdn.net/guanyonglai/article/details/59644437)

[print?](https://blog.csdn.net/guanyonglai/article/details/59644437)

1. #include "opencv2/highgui/highgui.hpp"
2. #include "opencv2/imgproc/imgproc.hpp"
3. #include <iostream>
4. #include <stdio.h>
5. #include <stdlib.h>
6. #include <math.h>

9. using namespace cv;
10. using namespace std;

13. Mat src; Mat src\_gray;

16. RNG rng(12345);
17. //Scalar colorful = CV\_RGB(rng.uniform(0,255),rng.uniform(0,255),rng.uniform(0,255));

20. Point Center\_cal(vector<vector<Point> > contours,int i)//找到所提取轮廓的中心点
21. {
22. int centerx=0,centery=0,n=contours[i].size();
23. //在提取的小正方形的边界上每隔周长个像素提取一个点的坐标，求所提取四个点的平均坐标（即为小正方形的大致中心）
24. centerx = (contours[i][n/4].x + contours[i][n\*2/4].x + contours[i][3\*n/4].x + contours[i][n-1].x)/4;
25. centery = (contours[i][n/4].y + contours[i][n\*2/4].y + contours[i][3\*n/4].y + contours[i][n-1].y)/4;
26. Point point1=Point(centerx,centery);
27. return point1;
28. }





35. int main( int argc, char\*\* argv[] )
36. {

39. src = imread( "03025.jpg", 1 );
40. Mat src\_all=src.clone();



45. cvtColor( src, src\_gray, CV\_BGR2GRAY );
46. //  src\_gray = Scalar::all(255) - src\_gray;
47. blur( src\_gray, src\_gray, Size(3,3) );
48. equalizeHist( src\_gray, src\_gray );
49. imshow("滤波后",src\_gray);

52. Scalar color = Scalar(1,1,255 );
53. Mat threshold\_output;
54. vector<vector<Point> > contours,contours2;
55. vector<Vec4i> hierarchy;
56. Mat drawing = Mat::zeros( src.size(), CV\_8UC3 );
57. Mat drawing2 = Mat::zeros( src.size(), CV\_8UC3 );
58. threshold( src\_gray, threshold\_output, 112, 255, THRESH\_BINARY );
59. //Canny(src\_gray,threshold\_output,136,196,3);
60. //imshow("预处理后：",threshold\_output);

63. findContours( threshold\_output, contours, hierarchy,  CV\_RETR\_TREE, CHAIN\_APPROX\_NONE, Point(0, 0) );
64. //CHAIN\_APPROX\_NONE全体,CV\_CHAIN\_APPROX\_SIMPLE,,,RETR\_TREE    RETR\_EXTERNAL    RETR\_LIST   RETR\_CCOMP

67. int c=0,ic=0,k=0,area=0;
68. //
69. //程序的核心筛选
70. //程序的核心筛选
71. int parentIdx=-1;
72. for( int i = 0; i< contours.size(); i++ )
73. {
74. if (hierarchy[i][2] != -1 && ic==0)
75. {
76. parentIdx = i;
77. ic++;
78. }
79. else if (hierarchy[i][2] != -1)
80. {
81. ic++;
82. }
83. else if(hierarchy[i][2] == -1)
84. {
85. ic = 0;
86. parentIdx = -1;
87. }

90. if ( ic >= 2)
91. {
92. contours2.push\_back(contours[parentIdx]);
93. drawContours( drawing, contours, parentIdx,  CV\_RGB(rng.uniform(0,255),rng.uniform(0,255),rng.uniform(0,255)) , 1, 8);
94. ic = 0;
95. parentIdx = -1;
96. area = contourArea(contours[i]);//得出一个二维码定位角的面积，以便计算其边长（area\_side）（数据覆盖无所谓，三个定位角中任意一个数据都可以）
97. }
98. //cout<<"i= "<<i<<" hierarchy[i][2]= "<<hierarchy[i][2]<<" parentIdx= "<<parentIdx<<" ic= "<<ic<<endl;

101. }

104. for(int i=0; i<contours2.size(); i++)
105. drawContours( drawing2, contours2, i,  CV\_RGB(rng.uniform(100,255),rng.uniform(100,255),rng.uniform(100,255)) , -1, 4, hierarchy[k][2], 0, Point() );

108. Point point[3];
109. for(int i=0; i<contours2.size(); i++)
110. {
111. point[i] = Center\_cal( contours2, i );
112. }
113. <span style="white-space:pre;"> </span>
114. <span style="white-space:pre;"> </span>area = contourArea(contours2[1]);//为什么这一句和前面一句计算的面积不一样呢
115. int area\_side = cvRound (sqrt (double(area)));
116. for(int i=0; i<contours2.size(); i++)
117. {
118. line(drawing2,point[i%contours2.size()],point[(i+1)%contours2.size()],color,area\_side/2,8);
119. }

122. imshow( "提取后", drawing2 );
123. printf("%d\n", contours.size());
124. //imshow( "Contours", drawing );

127. <span style="white-space:pre;"> </span>//接下来要框出这整个二维码
128. <span style="white-space:pre;"> </span>Mat gray\_all,threshold\_output\_all;
129. <span style="white-space:pre;"> </span>vector<vector<Point> > contours\_all;
130. vector<Vec4i> hierarchy\_all;
131. <span style="white-space:pre;"> </span>cvtColor( drawing2, gray\_all, CV\_BGR2GRAY );

134. <span style="white-space:pre;"> </span>threshold( gray\_all, threshold\_output\_all, 45, 255, THRESH\_BINARY );
135. <span style="white-space:pre;"> </span>findContours( threshold\_output\_all, contours\_all, hierarchy\_all,  RETR\_EXTERNAL, CHAIN\_APPROX\_NONE, Point(0, 0) );//RETR\_EXTERNAL表示只寻找最外层轮廓

138. <span style="white-space:pre;"> </span>//求最小包围矩形，斜的也可以哦
139. <span style="white-space:pre;"> </span> RotatedRect rectPoint = minAreaRect(contours\_all[0]);
140. <span style="white-space:pre;"> </span> Point2f fourPoint2f[4];

143. <span style="white-space:pre;"> </span> //将rectPoint变量中存储的坐标值放到 fourPoint的数组中
144. <span style="white-space:pre;"> </span> rectPoint.points(fourPoint2f);

147. <span style="white-space:pre;"> </span>  for (int i = 0; i < 4; i++)
148. {
149. line(src\_all, fourPoint2f[i%4], fourPoint2f[(i + 1)%4]
150. , Scalar(20,21,237), 3);
151. }

154. <span style="white-space:pre;"> </span> imshow( "二维码", src\_all );

157. waitKey(0);
158. return(0);
159. }

#include "opencv2/highgui/highgui.hpp"

#include "opencv2/imgproc/imgproc.hpp"

#include <iostream>

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

using namespace cv;

using namespace std;

Mat src; Mat src\_gray;

RNG rng(12345);

//Scalar colorful = CV\_RGB(rng.uniform(0,255),rng.uniform(0,255),rng.uniform(0,255));

Point Center\_cal(vector<vector<Point> > contours,int i)//找到所提取轮廓的中心点

   {

          int centerx=0,centery=0,n=contours[i].size();

          //在提取的小正方形的边界上每隔周长个像素提取一个点的坐标，求所提取四个点的平均坐标（即为小正方形的大致中心）

          centerx = (contours[i][n/4].x + contours[i][n\*2/4].x + contours[i][3\*n/4].x + contours[i][n-1].x)/4;

          centery = (contours[i][n/4].y + contours[i][n\*2/4].y + contours[i][3\*n/4].y + contours[i][n-1].y)/4;

          Point point1=Point(centerx,centery);

          return point1;

   }

int main( int argc, char\*\* argv[] )

{

  src = imread( "03025.jpg", 1 );

  Mat src\_all=src.clone();

  cvtColor( src, src\_gray, CV\_BGR2GRAY );

//  src\_gray = Scalar::all(255) - src\_gray;

  blur( src\_gray, src\_gray, Size(3,3) );

  equalizeHist( src\_gray, src\_gray );

  imshow("滤波后",src\_gray);

  Scalar color = Scalar(1,1,255 );

  Mat threshold\_output;

  vector<vector<Point> > contours,contours2;

  vector<Vec4i> hierarchy;

  Mat drawing = Mat::zeros( src.size(), CV\_8UC3 );

  Mat drawing2 = Mat::zeros( src.size(), CV\_8UC3 );

  threshold( src\_gray, threshold\_output, 112, 255, THRESH\_BINARY );

  //Canny(src\_gray,threshold\_output,136,196,3);

   //imshow("预处理后：",threshold\_output);

  findContours( threshold\_output, contours, hierarchy,  CV\_RETR\_TREE, CHAIN\_APPROX\_NONE, Point(0, 0) );

  //CHAIN\_APPROX\_NONE全体,CV\_CHAIN\_APPROX\_SIMPLE,,,RETR\_TREE    RETR\_EXTERNAL    RETR\_LIST   RETR\_CCOMP

    int c=0,ic=0,k=0,area=0;

//

    //程序的核心筛选

    //程序的核心筛选

    int parentIdx=-1;

    for( int i = 0; i< contours.size(); i++ )

    {

        if (hierarchy[i][2] != -1 && ic==0)

        {

            parentIdx = i;

            ic++;

        }

        else if (hierarchy[i][2] != -1)

        {

            ic++;

        }

        else if(hierarchy[i][2] == -1)

        {

            ic = 0;

            parentIdx = -1;

        }

        if ( ic >= 2)

        {

            contours2.push\_back(contours[parentIdx]);

            drawContours( drawing, contours, parentIdx,  CV\_RGB(rng.uniform(0,255),rng.uniform(0,255),rng.uniform(0,255)) , 1, 8);

            ic = 0;

            parentIdx = -1;

            area = contourArea(contours[i]);//得出一个二维码定位角的面积，以便计算其边长（area\_side）（数据覆盖无所谓，三个定位角中任意一个数据都可以）

        }

        //cout<<"i= "<<i<<" hierarchy[i][2]= "<<hierarchy[i][2]<<" parentIdx= "<<parentIdx<<" ic= "<<ic<<endl;

    }

    for(int i=0; i<contours2.size(); i++)

        drawContours( drawing2, contours2, i,  CV\_RGB(rng.uniform(100,255),rng.uniform(100,255),rng.uniform(100,255)) , -1, 4, hierarchy[k][2], 0, Point() );

    Point point[3];

    for(int i=0; i<contours2.size(); i++)

    {

        point[i] = Center\_cal( contours2, i );

    }

area = contourArea(contours2[1]);//为什么这一句和前面一句计算的面积不一样呢

    int area\_side = cvRound (sqrt (double(area)));

    for(int i=0; i<contours2.size(); i++)

    {

        line(drawing2,point[i%contours2.size()],point[(i+1)%contours2.size()],color,area\_side/2,8);

    }

    imshow( "提取后", drawing2 );

    printf("%d\n", contours.size());

    //imshow( "Contours", drawing );

//接下来要框出这整个二维码

Mat gray\_all,threshold\_output\_all;

vector<vector<Point> > contours\_all;

    vector<Vec4i> hierarchy\_all;

cvtColor( drawing2, gray\_all, CV\_BGR2GRAY );

threshold( gray\_all, threshold\_output\_all, 45, 255, THRESH\_BINARY );

findContours( threshold\_output\_all, contours\_all, hierarchy\_all,  RETR\_EXTERNAL, CHAIN\_APPROX\_NONE, Point(0, 0) );//RETR\_EXTERNAL表示只寻找最外层轮廓

//求最小包围矩形，斜的也可以哦

RotatedRect rectPoint = minAreaRect(contours\_all[0]);

Point2f fourPoint2f[4];

//将rectPoint变量中存储的坐标值放到 fourPoint的数组中

rectPoint.points(fourPoint2f);

 for (int i = 0; i < 4; i++)

        {

            line(src\_all, fourPoint2f[i%4], fourPoint2f[(i + 1)%4]

                , Scalar(20,21,237), 3);

        }

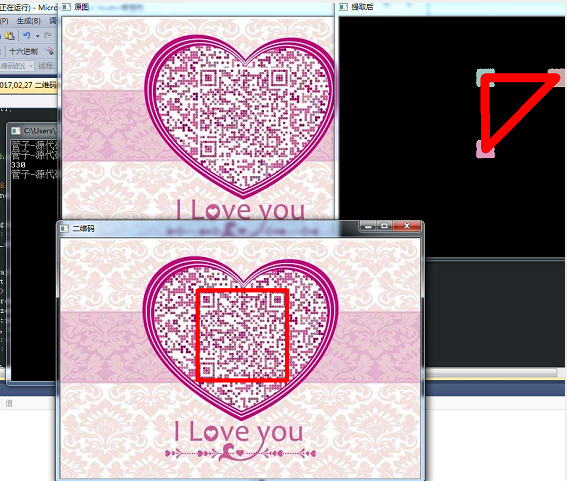
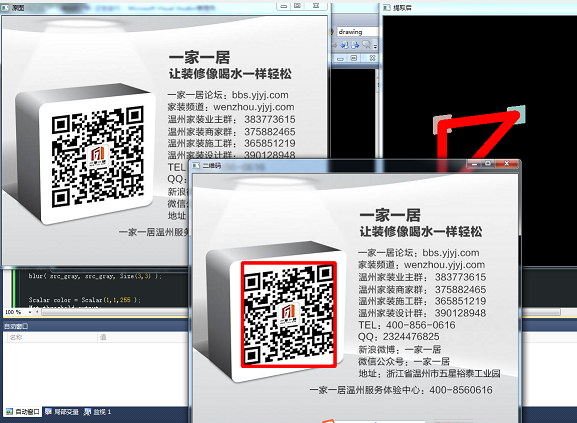
imshow( "二维码", src\_all );

    waitKey(0);

    return(0);

}

点评一下需要改进的吧，就是求最小包围矩形的时候，不能斜着画矩形，这个我还在看资料，改进应该不难的



个人分类： [学习](https://blog.csdn.net/guanyonglai/article/category/6066870)