#include "get\_date\_roi.h"

#include "opencv2/opencv.hpp"

#include<iostream>

using namespace std;

using namespace cv;

get\_date\_roi::get\_date\_roi()

{

}

get\_date\_roi::~get\_date\_roi()

{

}

//将图片旋转正

Mat get\_date\_roi::trans2horizon(Mat img)

{

//Read photo

//Mat srcImg = imread("C:\\Users\\nanyi\\Desktop\\类别1\\20191122A\\2020\_6\_30\_11\_39\_33\_415.bmp");

//imshow("src", srcImg);

Mat srcImg = img;

Mat dstImg = srcImg.clone();

//Split color channel

//提取红色通道

Mat BGRs[3];

split(srcImg, BGRs);

srcImg = BGRs[2];

//二值化操作

threshold(srcImg, srcImg, 30, 255, THRESH\_BINARY);

//Morphological opening-and-closing operation

Mat element\_open = getStructuringElement(MORPH\_RECT, Size(5, 5)); //定义结构元素

morphologyEx(srcImg, srcImg, MORPH\_OPEN, element\_open);

Mat element\_close = getStructuringElement(MORPH\_RECT, Size(30, 30)); //定义结构元素

morphologyEx(srcImg, srcImg, MORPH\_CLOSE, element\_close);

//imshow("erode", srcImg);

//imshow("dilate", srcImg);

//提取轮廓

vector<vector<Point>> contours;

vector<Vec4i> hierarcy;

findContours(srcImg, contours, hierarcy, CV\_RETR\_TREE, CV\_CHAIN\_APPROX\_NONE);

vector<Rect> boundRect(contours.size());

vector<RotatedRect> box(contours.size());

Point2f rect[4];

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

{

//根据面积过滤

box[i] = minAreaRect(Mat(contours[i]));

//boundRect[i] = boundingRect(Mat(contours[i]));

if (box[i].size.width < 100 || box[i].size.height < 100)//筛选

continue;

//rectangle(dstImg, Point(boundRect[i].x, boundRect[i].y), Point(boundRect[i].x + boundRect[i].width, boundRect[i].y + boundRect[i].height), Scalar(0, 255, 0), 2, 8);

//circle(dstImg, Point(box[i].center.x, box[i].center.y), 5, Scalar(0, 255, 0), -1, 8);

box[i].points(rect);

//画直线

/\*for (int j = 0; j < 4; j++)

{

line(dstImg, rect[j], rect[(j + 1) % 4], Scalar(0, 0, 255), 2, 8);

}\*/

float angle;

cout << "angle=" << box[i].angle << endl;

angle = box[i].angle;

char width[20], height[20];

//Trans operate

if (0 < abs(angle) && abs(angle) <= 45)

angle = angle;//负数，顺时针旋转

else if (45 < abs(angle) && abs(angle) < 90)

angle = 90 - abs(angle);//正数，逆时针旋转

Point2f center = box[i].center; //定义旋转中心坐标

double angle0 = angle;

double scale = 1;

Mat roateM = getRotationMatrix2D(center, angle0, scale); //获得旋转矩阵,顺时针为负，逆时针为正

warpAffine(dstImg, dstImg, roateM, dstImg.size()); //仿射变换，旋转

}

//namedWindow("dst", 0);

//imshow("dst", dstImg);

return dstImg;

}

Mat get\_date\_roi::get\_date(Mat img)

{

//String path = file\_path;

//String full\_path = file\_path + file\_name;//我想要一个可以直接所有cpp都能读的path

Mat srcImg = img;

//Read template

//Mat temp = imread("C:\\Users\\nanyi\\Desktop\\类别1\\20191122A\\1\_1\_2020\_7\_2\_16\_40\_7\_718.jpg");

Mat temp = imread(R"(C:\Users\pc\Desktop\img\mb.jpg)");

/\*if (img.empty() || temp.empty())

{

cout << "文件错误" << endl;

}\*/

Mat result;

matchTemplate(srcImg, temp, result, TM\_CCOEFF\_NORMED);

double maxVal, minVal;

Point minLoc, maxLoc;

minMaxLoc(result, &minVal, &maxVal, &minLoc, &maxLoc);

rectangle(srcImg, cv::Rect(maxLoc.x, maxLoc.y, temp.cols, temp.rows), Scalar(0, 0, 255), 2);

//rectangle(srcImg, rect, Scalar(0, 0, 255), 2); //多一根包装

//rectangle(srcImg, cv::Rect(maxLoc.x - 40, maxLoc.y + temp.rows + 10, 440, 100), Scalar(0, 0, 255), 2);

//rectangle(srcImg, cv::Rect(maxLoc.x - 350, maxLoc.y + temp.rows + 10, 750, 100), Scalar(0, 0, 255), 2);

Mat ROI = srcImg(Rect(maxLoc.x - 350, maxLoc.y + temp.rows + 10, 750, 100));

//imwrite("date.jpg", ROI);

//namedWindow("原图", 0);

//imshow("原图", img);

//waitKey(0);

return ROI;

}

void get\_date\_roi::do\_get\_date\_roi(Mat src, Mat & dst) {

Mat img1 = trans2horizon(src);

dst = get\_date(img1);

}