CHAPTER 2 基本功能

Wednesday, October 13, 2021 1:24 PM

1.转换灰度图&&高斯模糊

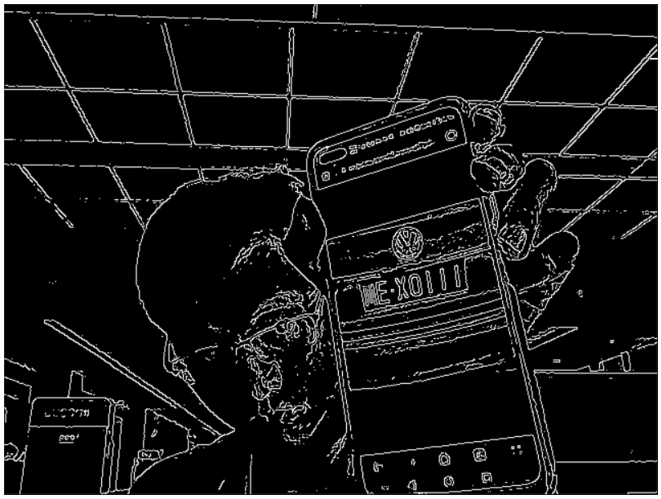
```
#include"headler.h"


void main()
{
	string path = "F:\\C++_Study_project\\opencv_project\\C++_opencvCourse\\C++_opencvCourse\\resource
	\\test.png";
	Mat img = imread(path);

	Mat imgGray,imgBlur, imgBlur2;
	cvtColor(img, imgGray, COLOR_BGR2GRAY); //转换图像为灰度图（转变为其他图也是这个）
	GaussianBlur(img, imgBlur, Size(7, 7), 50, 50);//高斯模糊 size为内核大小，越大越模糊
	GaussianBlur(img, imgBlur2, Size(7, 7), 3, 0);

	imshow("image", imgBlur2);
	imshow("img", img);
	imshow("image1", imgBlur);
	waitKey(0);

}
```

2.边缘检测

边缘检测使用canny函数，在边缘检测前需要先将图像模糊化

```
#include"headler.h"


void main()
{
	string path = "F:\\C++_Study_project\\opencv_project\\C++_opencvCourse\\C++_opencvCourse\\resource
	\\test.png";
	Mat img = imread(path);

	Mat imgGray,imgBlur, imgCanny;
	cvtColor(img, imgGray, COLOR_BGR2GRAY); //转换图像为灰度图（转变为其他图也是这个）
	GaussianBlur(img, imgBlur, Size(7, 7), 50, 50);//高斯模糊 size为内核大小，越大越模糊

	Canny(imgBlur, imgCanny, 50, 50);//值越大滤掉的越多，前者是下限后者是上限

	imshow("image", imgCanny);
	imshow("img", img);
	imshow("image1", imgBlur);
	waitKey(0);

}
```

DLC:边缘检测摄像头实时监控

```
#include"headler.h"

void main()
{
	VideoCapture cap(0);
	Mat img;

	while (true)
	{
		cap.read(img);
		Mat imgGray,imgBlur,imgCanny;
		cvtColor(img, imgGray, COLOR_BGR2GRAY);
		GaussianBlur(imgGray, imgBlur, Size(7, 7), 0, 4);
		Canny(imgGray, imgCanny, 50, 50);

		imshow("imgcanny", imgCanny);
		waitKey(1);
	}
}
```

3.膨胀腐蚀

dilate：膨胀

erode：腐蚀



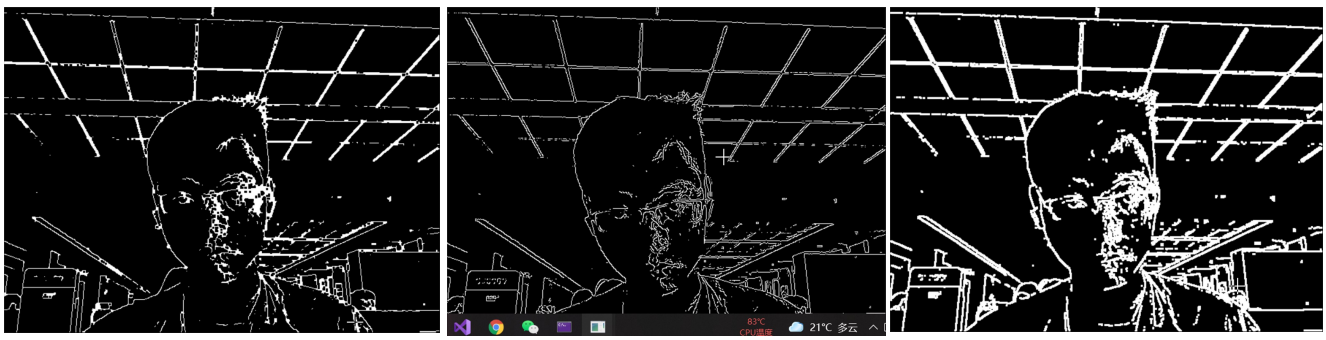
膨胀　　膨胀后腐蚀

MORPH_ERODE	腐蚀
MORPH_DILATE	膨胀
MORPH_OPEN	开运算
MORPH_CLOSE	闭运算
MORPH_GRADIENT	形态学梯度
MORPH_TOPHAT	顶帽运算
MORPH_BLACKHAT	黑帽运算

- 开运算与闭运算
  开运算与闭运算都是在腐蚀膨胀的基础上进行的；准确地说，**开运算就是先进行腐蚀再进行膨胀，可以消除图像中较小的亮点**（在前景检测中开运算能够很好的消除二值前景图像中的噪声点）；**闭运算就是先进行膨胀再进行腐蚀，可以消除图像中小型的黑洞。**
- 形态学梯度
  形态学梯度与开闭运算一样，是在腐蚀膨胀的基础上进行的运算；不同的是形态学梯度图是**膨胀图与腐蚀图做差分得到的图像**，通过膨胀图减去腐蚀图可以得到图像的边缘轮廓区域的像素。
- 顶帽与黑帽
  顶帽：原图像与开运算图之差，用于**分离局部较亮区域的像素。**
  黑帽：闭运算与原图像之差，用于**分离局部较暗区域的像素。**

```
#include"headler.h"


void main()
{
	string path = "F:\\C++_Study_project\\opencv_project\\C++_opencvCourse\\C++_opencvCourse\\resource\\test.png";
	Mat img = imread(path);

	Mat imgGray,imgBlur, imgCanny, imgDia, imgErode;
	cvtColor(img, imgGray, COLOR_BGR2GRAY); //转换图像为灰度图（转变为其他图也是这个）
	GaussianBlur(img, imgBlur, Size(7, 7), 50, 50);//高斯模糊 size为内核大小，越大越模糊

	Canny(imgBlur, imgCanny, 50, 50);//值越大滤掉的越多，可以简单理解为只有像素>所设值才是边缘?
	Mat Kernel1 = getStructuringElement(MORPH_DILATE,Size(3,3));//定义内核，将在这里定义类型或者形状,可以创建一个膨胀内核，尺寸越大膨胀越多
	Mat Kernel2 = getStructuringElement(MORPH_ERODE, Size(3, 3));
	//最好只用奇数
	dilate(imgCanny, imgDia,Kernel1);//输入输出内核（膨胀）

	erode(imgDia, imgErode, Kernel2);//腐蚀

	imshow("image", imgCanny);
	imshow("img", img);
	imshow("image1", imgBlur);
	imshow("image Dilation", imgDia);
	imshow("image Erodation", imgErode);
	waitKey(0);

}
```

DLC:膨胀腐蚀摄像头实时监控



```
#include"headler.h"


void main()
{
	VideoCapture cap(0);
	Mat img;

	while (true)
	{
```

```
            cap.read(img);
            Mat imgGray, imgBlur, imgCanny,imgDia,imgErode;
            cvtColor(img, imgGray, COLOR_BGR2GRAY);
            GaussianBlur(imgGray, imgBlur, Size(7, 7), 0, 4);
            Canny(imgGray, imgCanny, 50, 50);

            Mat kernel = getStructuringElement(MORPH_RECT, Size(3, 3));
            dilate(imgCanny, imgDia, kernel);
            erode(imgDia, imgErode, kernel);

            imshow("imgCanny", imgCanny);
            imshow("imgDia", imgDia);
            imshow("imErode", imgErode);
            waitKey(1);
        }
}
```