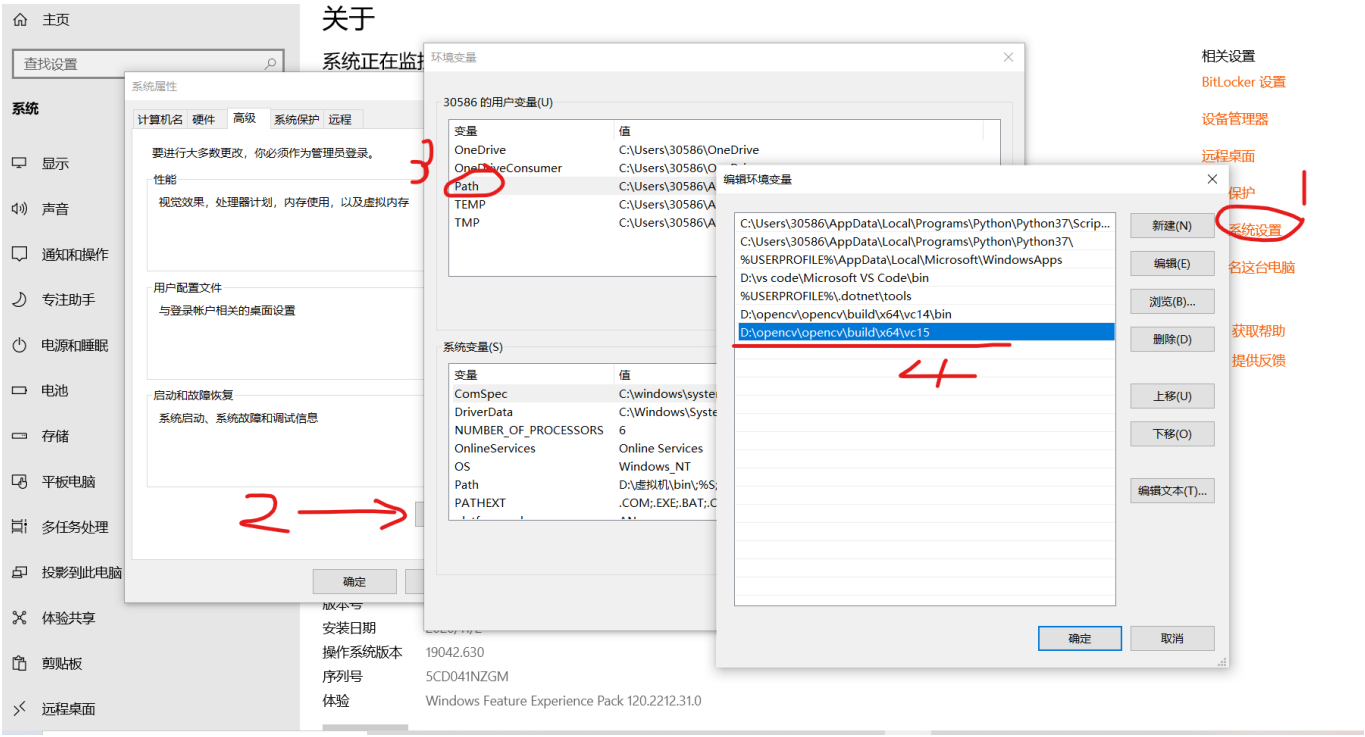
第一次作业

opencv安装以及配置

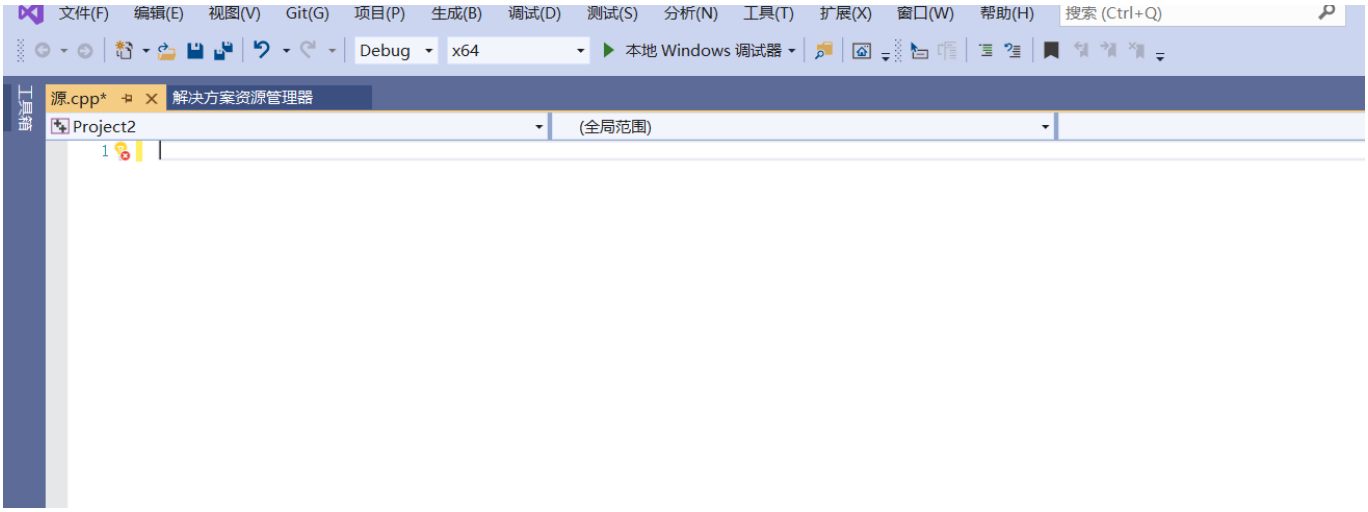
软件准备

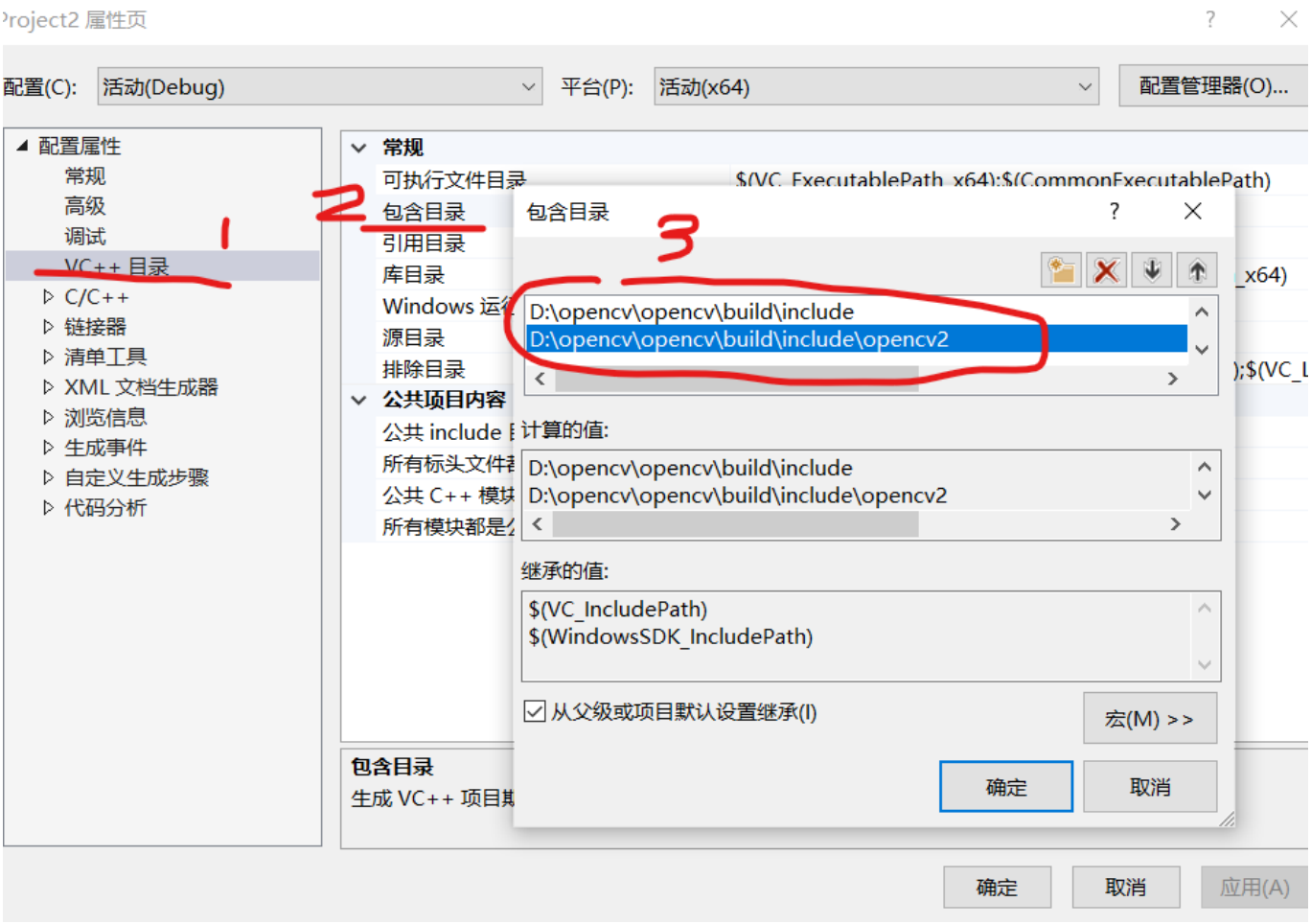
1. Visual Studio 2019
2. opencv

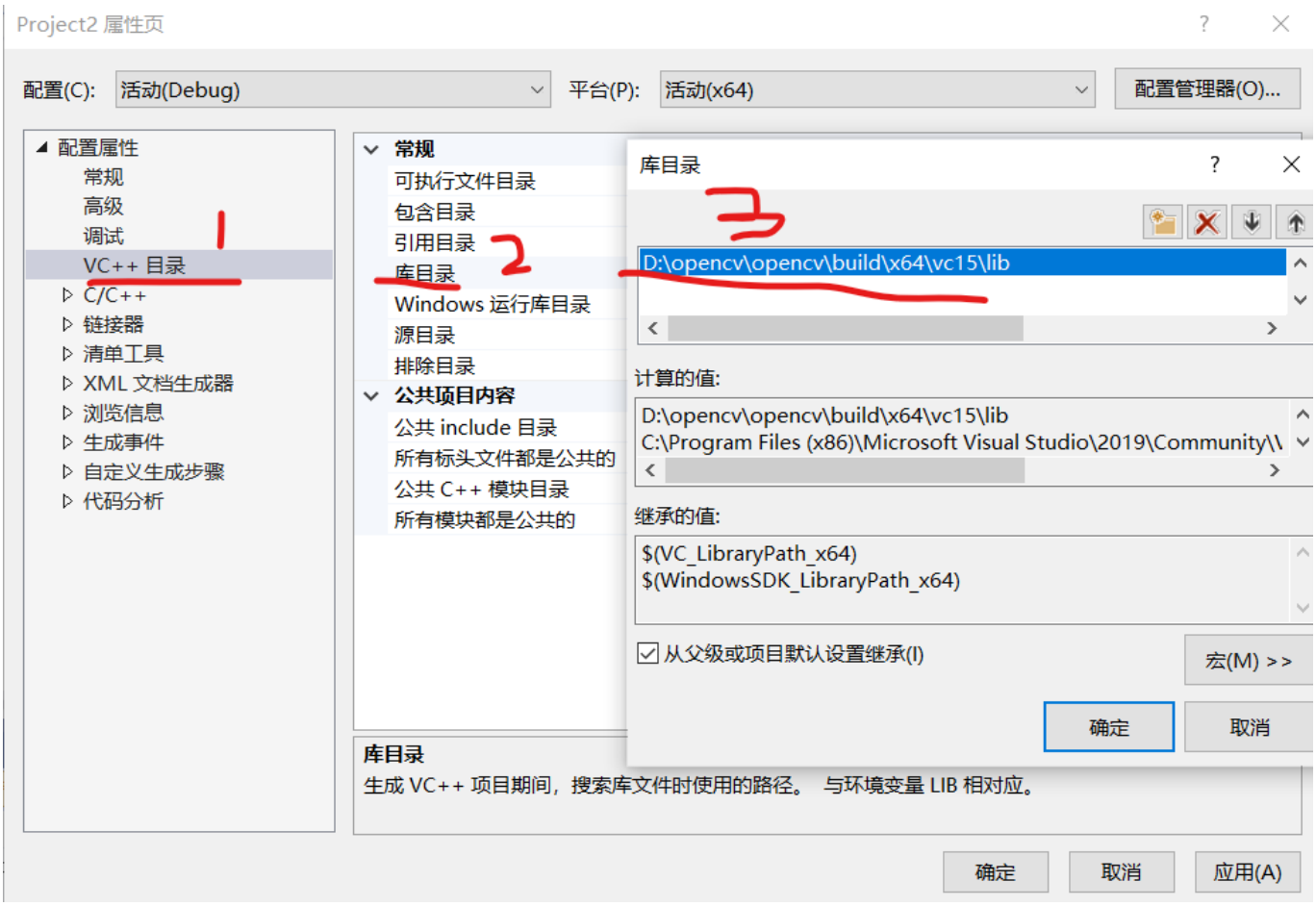
环境配置 环境变量配置

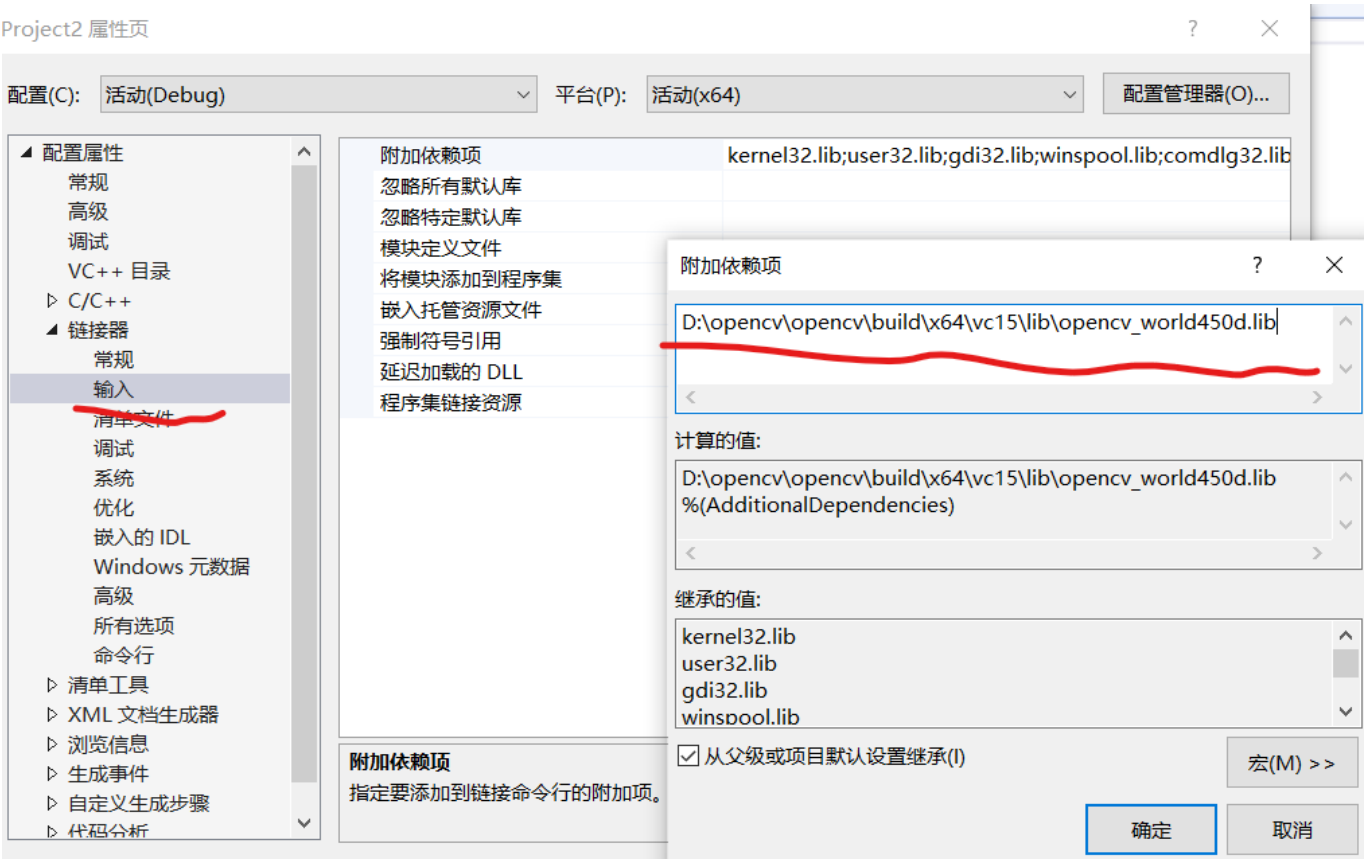


在vs2019创建项目进行操作,并配置相关参数




Project2 属性页
配置(C): 活动(Debug)
平台(P): 活动(x64)
配置管理器(O)...
配置属性
常规
高级
调试
VC++ 目录
C/C++
链接器
清单工具
XML 文档生成器
浏览信息
生成事件
自定义生成步骤
代码分析
常规
可执行文件目录
包含目录
引用目录
库目录
Windows 运行库目录
源目录
排除目录
公共项目内容
公共 include 目录
所有标头文件都是公共的
公共 C++ 模块目录
所有模块都是公共的
包含目录
D:\opencv\opencv\build\include
D:\opencv\opencv\build\include\opencv2
计算的值:
D:\opencv\opencv\build\include
D:\opencv\opencv\build\include\opencv2
继承的值:
$(VC_IncludePath)
$(WindowsSDK_IncludePath)
从父级或项目默认设置继承(I)
宏(M) >>
确定
取消
包含目录
生成 VC++ 项目期间，搜索包含文件时使用的路径。 与环境变量 INCLUDE 相对应。
确定
取消
应用(A)
1
2
3


Project2 属性页
配置(C): 活动(Debug)
平台(P): 活动(x64)
配置管理器(O)...
配置属性
常规
高级
调试
VC++ 目录
C/C++
链接器
清单工具
XML 文档生成器
浏览信息
生成事件
自定义生成步骤
代码分析
常规
可执行文件目录
包含目录
引用目录
库目录
Windows 运行库目录
源目录
排除目录
公共项目内容
公共 include 目录
所有标头文件都是公共的
公共 C++ 模块目录
所有模块都是公共的
库目录
D:\opencv\opencv\build\x64\vc15\lib
计算的值:
D:\opencv\opencv\build\x64\vc15\lib
C:\Program Files (x86)\Microsoft Visual Studio\2019\Community\\
继承的值:
$(VC_LibraryPath_x64)
$(WindowsSDK_LibraryPath_x64)
从父级或项目默认设置继承(I)
宏(M) >>
确定
取消
库目录
生成 VC++ 项目期间，搜索库文件时使用的路径。 与环境变量 LIB 相对应。
确定
取消
应用(A)
1
2
3



opencv实验

opencv 图像处理

实验一图像显示

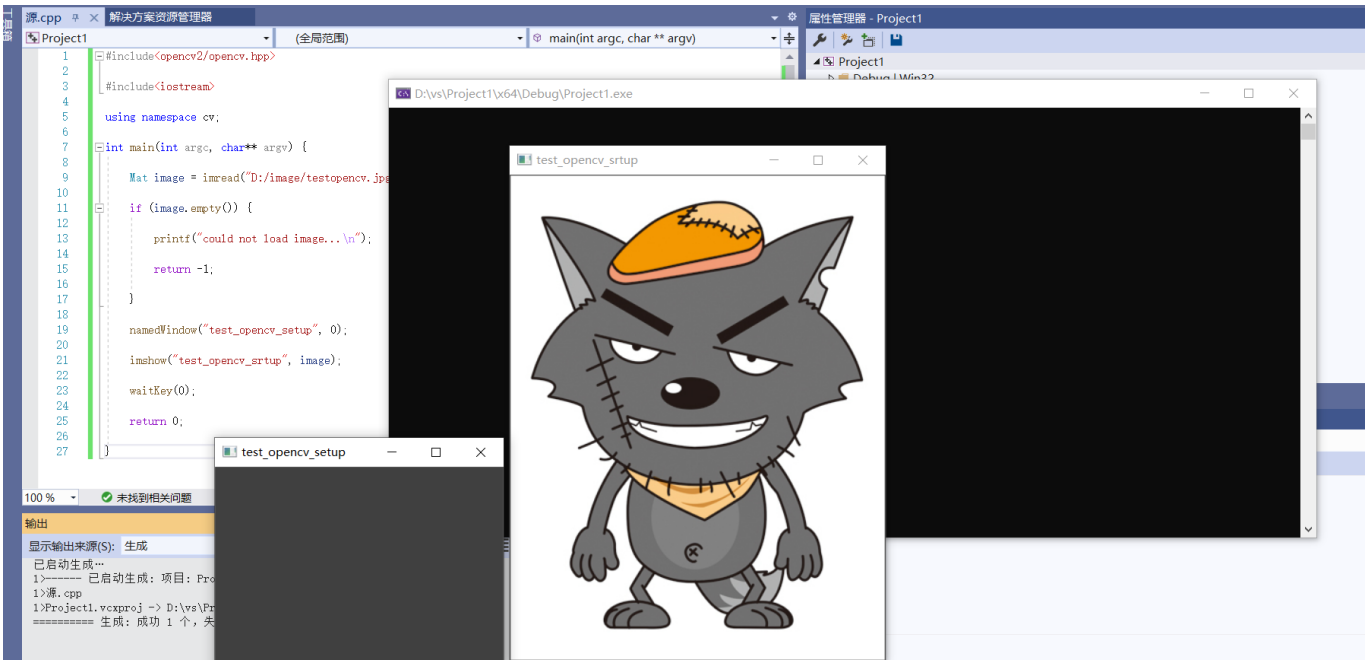
```
#include<opencv2/opencv.hpp>
#include<iostream>
using namespace cv;
int main(int argc, char** argv) {
    Mat image = imread("D:/image/testopencv.jpg");
    if (image.empty()) {
        printf("could not load image...\n");
        return -1;
    }
    namedWindow("test_opencv_setup", 0);
    imshow("test_opencv_srtup", image);
```

```
    waitKey(0);

    return 0;

}
```

实验运行结果

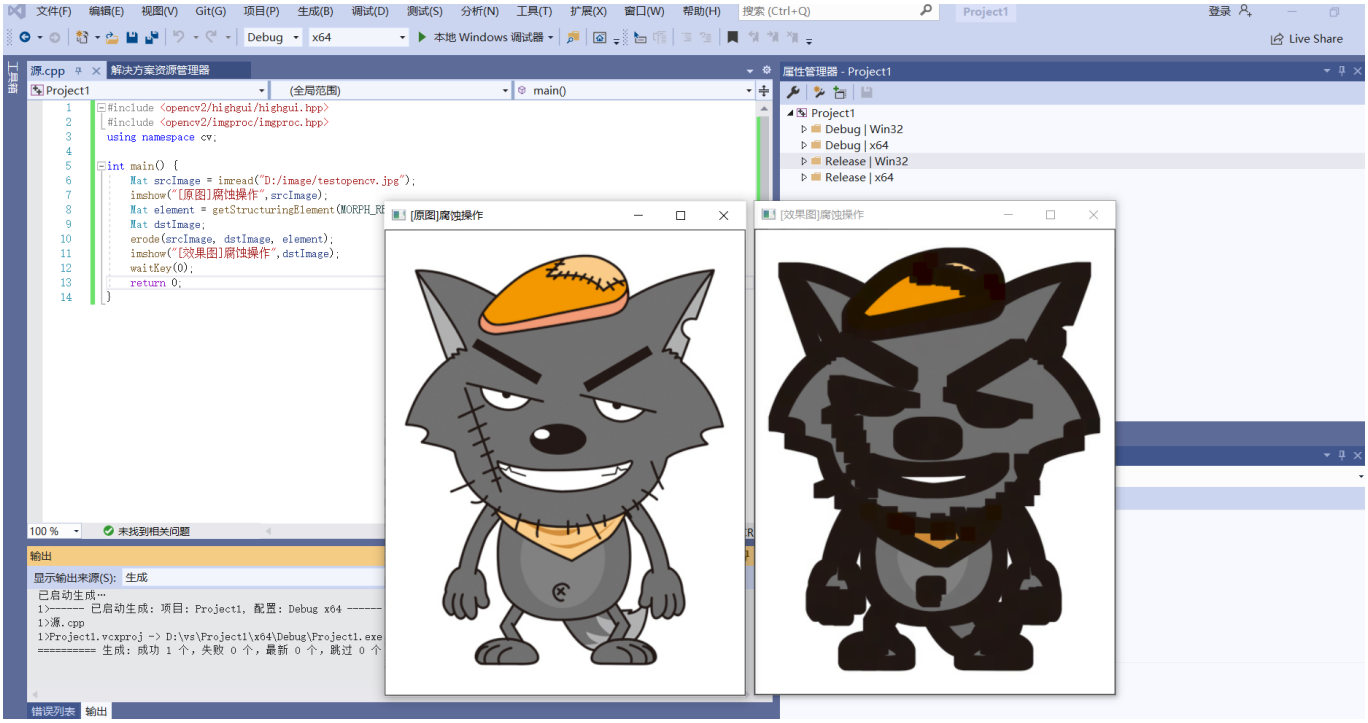


实验二图像腐蚀

```
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/imgproc/imgproc.hpp>
using namespace cv;

int main() {
    Mat srcImage = imread("D:/image/testopencv.jpg");
    imshow("[原图]腐蚀操作",srcImage);
    Mat element = getStructuringElement(MORPH_RECT, Size(15, 15));
    Mat dstImage;
    erode(srcImage, dstImage, element);
    imshow("[效果图]腐蚀操作",dstImage);
    waitKey(0);
    return 0;
}
```

实验结果

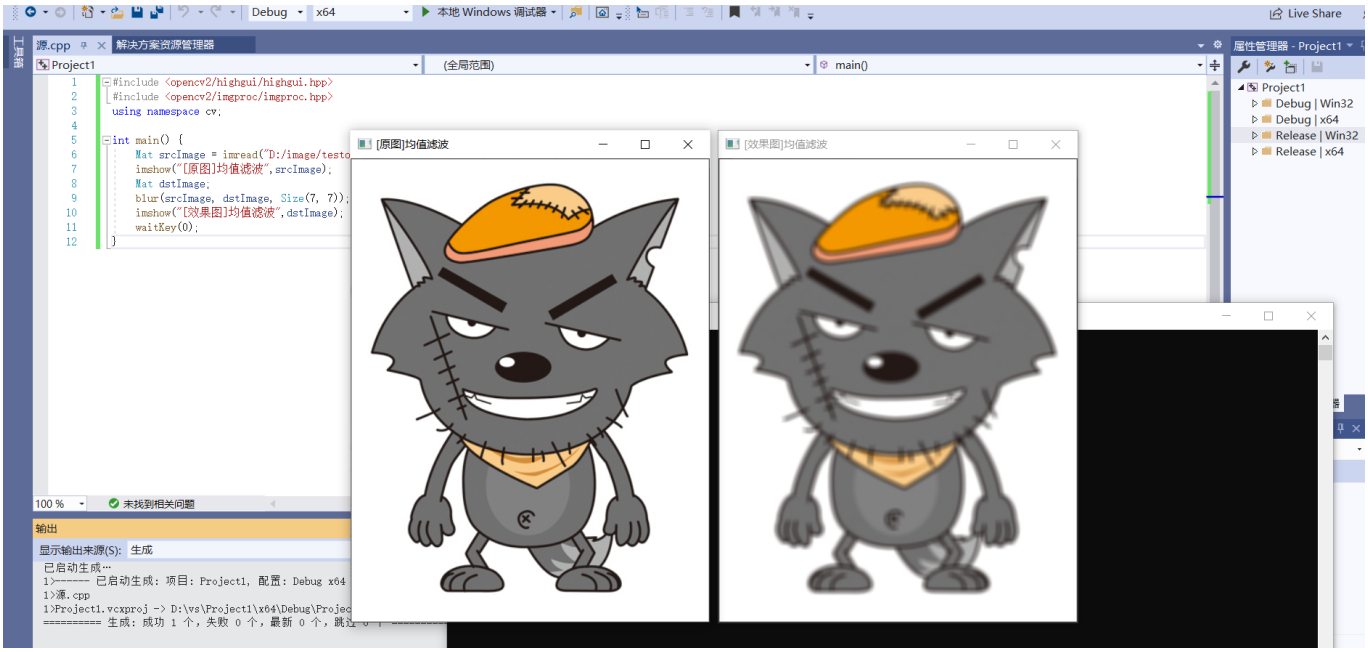


实验三图像模糊处理

```
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/imgproc/imgproc.hpp>
using namespace cv;

int main() {
    Mat srcImage = imread("D:/image/testopencv.jpg");
    imshow("[原图]均值滤波",srcImage);
    Mat dstImage;
    blur(srcImage, dstImage, Size(7, 7));
    imshow("[效果图]均值滤波",dstImage);
    waitKey(0);
}
```

实验结果



实验四carry边缘检测

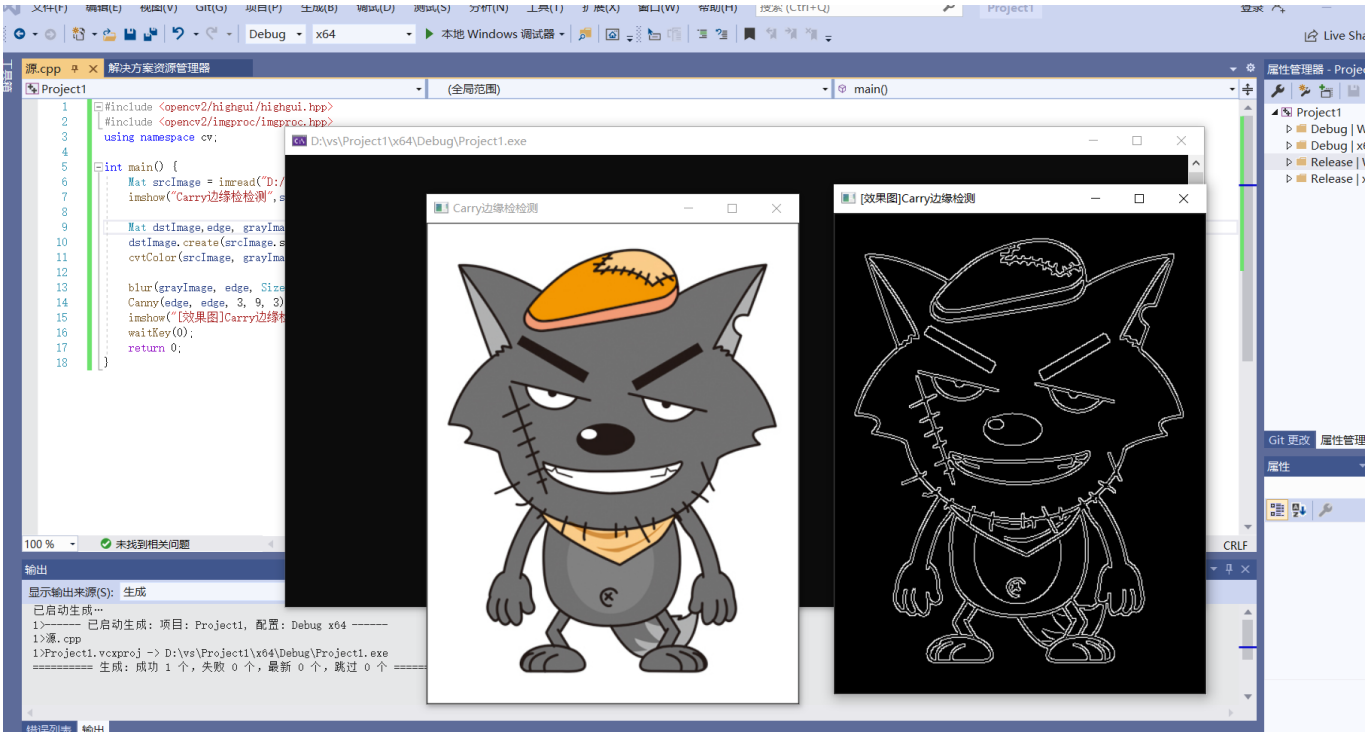
```
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/imgproc/imgproc.hpp>
using namespace cv;

int main() {
    Mat srcImage = imread("D:/image/testopencv.jpg");
    imshow("Carry边缘检检测",srcImage);

    Mat dstImage,edge, grayImage;
    dstImage.create(srcImage.size(), srcImage.type());
    cvtColor(srcImage, grayImage, COLOR_BGR2GRAY);

    blur(grayImage, edge, Size(3, 3));
    Canny(edge, edge, 3, 9, 3);
    imshow("[效果图]Carry边缘检测",edge);
    waitKey(0);
    return 0;
}
```

实验结果



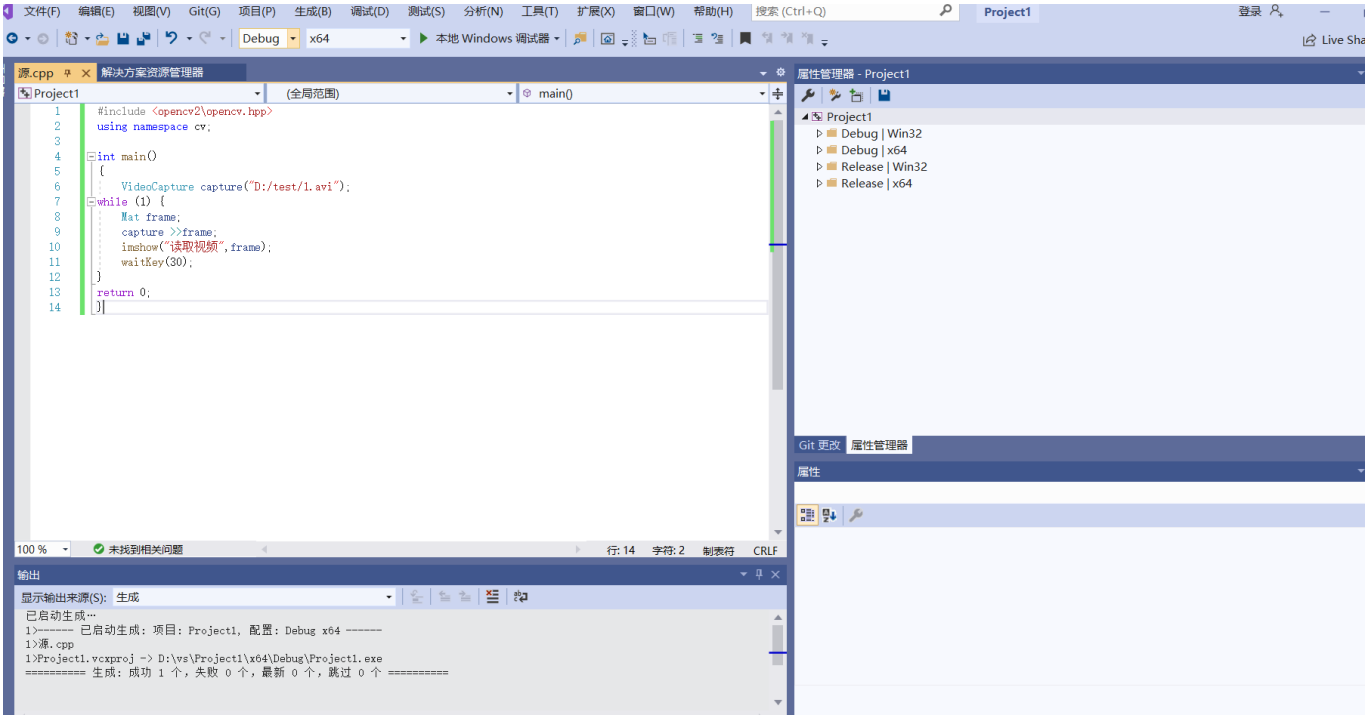
opencv 视频操作

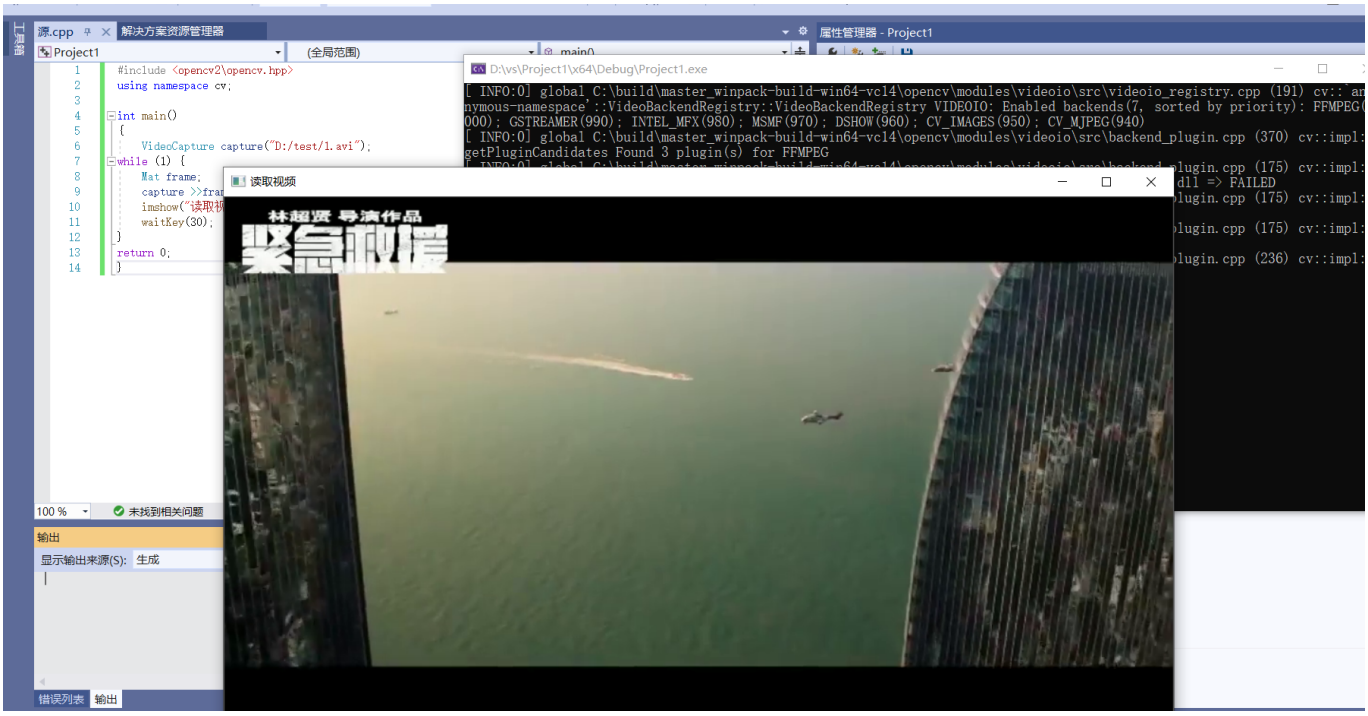
实验五读取播放视频

```
#include <opencv2\opencv.hpp>
using namespace cv;

int main()
{
    VideoCapture capture("D:/test/1.avi");
while (1) {
    Mat frame;
    capture >>frame;
    imshow("读取视频",frame);
    waitKey(30);
}
return 0;
}
```

下载视频并且存放到代码所指定的文件夹





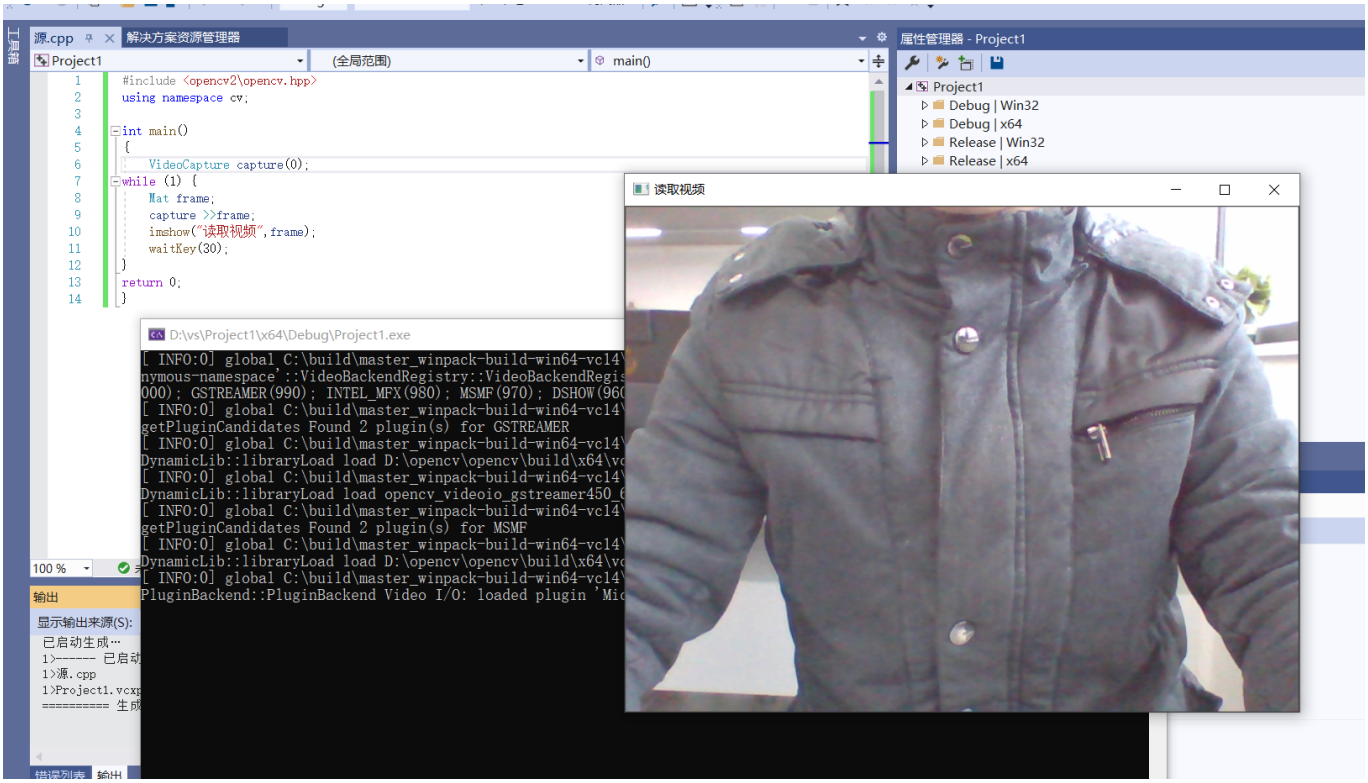
实验六调用摄像头采集图像

```
#include <opencv2\opencv.hpp>
using namespace cv;

int main()
{
    VideoCapture capture(0);
while (1) {
    Mat frame;
    capture >>frame;
    imshow("读取视频",frame);
```

```
    waitKey(30);
}
return 0;
}
```

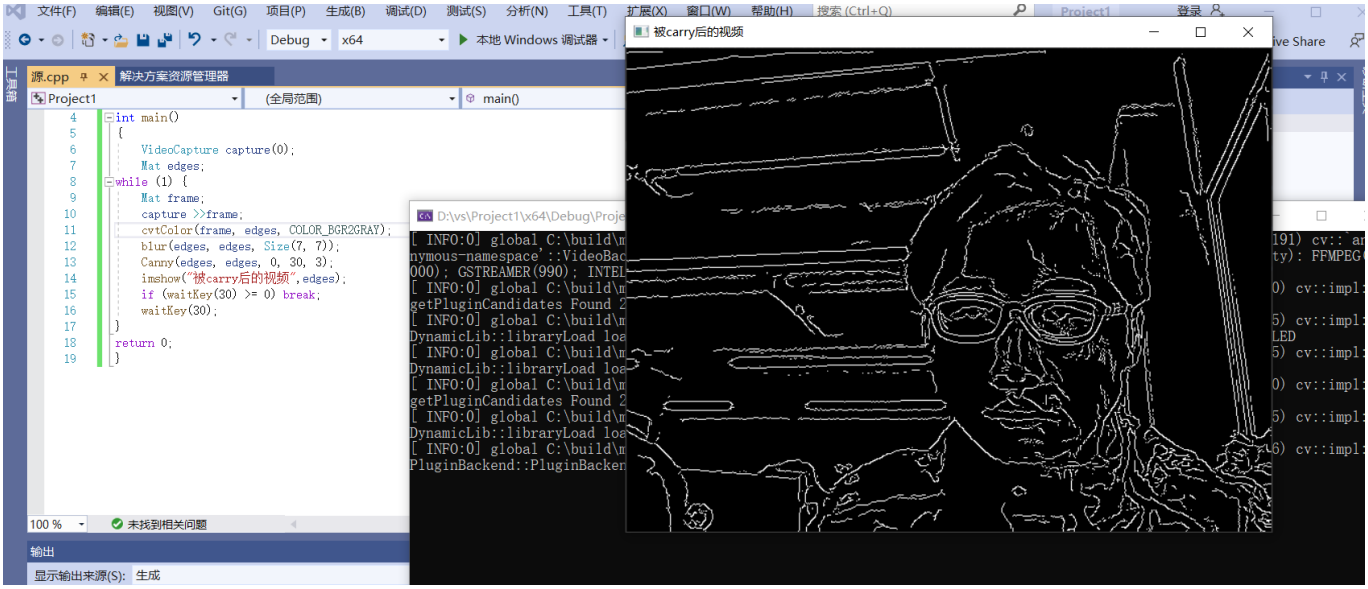
实验结果



实验七carry边缘检测

```
int main()
{
    VideoCapture capture(0);
    Mat edges;
while (1) {
    Mat frame;
    capture >>frame;
    cvtColor(frame, edges, COLOR_BGR2GRAY);
    blur(edges, edges, Size(7, 7));
    Canny(edges, edges, 0, 30, 3);
    imshow("被carry后的视频",edges);
    if (waitKey(30) >= 0) break;
    waitKey(30);
}
return 0;
}
```

实验结果


被carry后的视频
源.cpp
解决方案资源管理器
Project1
(全局范围)
main()
int main()
{
VideoCapture capture(0);
Mat edges;
while (1) {
Mat frame;
capture >>frame;
cvtColor(frame, edges, COLOR_BGR2GRAY);
blur(edges, edges, Size(7, 7));
Canny(edges, edges, 0, 30, 3);
imshow("被carry后的视频",edges);
if (waitKey(30) >= 0) break;
}
return 0;
}
100 %
未找到相关问题
输出
显示输出来源(S): 生成