

# 计算机视觉

南区计软326

深圳大学 计算机与软件学院





# 实验内容

序号	实验主题	实验内容	实验要求	实验时数	每组人数	实验 类型
1	图像处理应用 实验	1. 熟悉图像的表示及基本元素、通道操作; 2. 掌握基本图像增强方法; 3. 掌握OpenCV计算机视觉库;	必做	6	1	讲授 + 实验
2	图像特征提取 及综合应用实 践	<ol> <li>熟悉图像处理基本操作;</li> <li>掌握图像边缘检测原理;</li> <li>掌握图像基本特征抽取以及在实际问题中的应用;</li> </ol>	必做	6	1	讲授 + 实验
3	计算机视觉系 统实践	1. 熟悉计算机视觉分类任务; 2. 掌握数据集的准备及模型训练过程; 3. 培养应用计算机视觉解决问题的能力;	必做	6	1	讲授 + 实验

# 涉及学科

- ●数字图像处理
- ●计算机视觉
- ●模式识别
- ●程序设计





# 实验考察形式

- 三次实验报告(在Blackboard发布与提交)
- 平时编程练习及表现
- 实验汇报

# 实验考察内容

- 计算机视觉与模式识别基础知识掌握能力
- 算法设计能力
- 编程及系统架构能力
- 论文写作及表达能力





# 上节课上机练习内容:

- 1. OpenCV环境配置;
- 2. 实现对图像的读取;
- 3. 把一副彩色图像的三个通道变成3个单通道图像存储到硬盘 上并显示;
- 4. 计算一幅单通道图像的直方图;
- 5. 编程实现对一幅单通道图像的边缘检测。
- 6. 对一幅灰度图像进行直方图均衡化
- 7.对图像进行二值化操作;
- 8. 图像形态学操作;
- 9. 获取图像轮廓;
- 10. 实现目标的计数;
- 11. 目标的旋转;









### 实验二 图像处理综合-路沿检测

### 实验目的:

- 1. 熟悉图像处理基本操作;
- 2. 掌握图像边缘检测原理;
- 3. 掌握图像基本特征抽取以及在实际问题中的应用;

### 实验要求:

- 1. 实验提交文件为实验报告和相关程序代码,以压缩包的形式提交,实验报告命名规则为"计算机视觉-学号-姓名-实验报告2. doc",其他文件打包成压缩文件,命名为"计算机视觉-学号-姓名-实验报告2-其他. zip";
- 2. 所有素材和参考材料需列明出处,实验报告中的图片和程序代码建议标注个人水印或标识信息: 姓名,班级,学号信息;

### 二、实验内容:

针对给定的视频,利用图像处理基本方法实现道路路沿的检测;

提示:可利用Hough变换进行线检测,融合路沿的结构信息实现路沿边界定位(图中红色的点位置)。

### 三、实验时间

实验报告统一命名为: 计算机视觉-学号-姓名-实验报告2. doc 如有提交其他文件或程序, 统一命名为: 计算机视觉-学号-姓名-实验报告2-其他. zip

实验报告提交时间: 2024年11月18日下午5:00 (注意按时提交,不要延期)



# ● 实验二

#### •一、实验目的与要求

#### 实验目的:

- 1. 熟悉图像处理基本操作:
- 2. 掌握图像边缘检测原理;
- 3. 掌握图像基本特征抽取以及在实际问题中的应用:

#### 实验要求:

- 实验提交文件为实验报告和相关程序代码,以压缩包的形式提交,实验报告命名规则为"计算机视觉-学号-姓名-实验报告 2.doc",其他文件打包成压缩文件,命名为"计算机视觉-学号-姓名-实验报告 2-其他zip";
- 所有素材和参考材料需列明出处,实验报告中的图片和程序代码建议标注个人水印或标识信息:姓名,班级,学号信息;

#### •二、实验内容与方法

实验内容: 针对给定的视频,利用图像处理基本方法实现道路路沿的检测: 提示: 可利用 Hough 变换进行线检测,融合路沿的结构信息实现路沿边界定位(图中 红色的点位置)。



#### •三、实验步骤与过程

是示、是文件を附近開始了底标注提示信息力 根据实验内容自己到地填写。内容被回插入实验提出后人小要偶想这中,并且多求加上相关的说明。每页图片建议不超过6件,排取的类型也加入评分标准。(以后所有的实验报告都是这个要求、不再单复证明)。

#### • 四、实验结论或体会















# ● 模板匹配





Figure 13-9. cv::matchTemplate() sweeps a template image patch across another image looking for matches





已知目标图像,在新图像中找到该目标

CV\_EXPORTS\_W void matchTemplate( InputArray image, InputArray templ, OutputArray result, int method );

image: 待匹配的源图像

templ: 模板图像

result: 保存结果的矩阵, 我们可以通过minMaxLoc() 确定结果矩阵的最大值和最小

值的位置.

method: 模板匹配的算法

有以下六种:

enum { TM\_SQDIFF=0, TM\_SQDIFF\_NORMED=1, TM\_CCORR=2, TM\_CCORR\_NORMED=3, TM\_CCOEFF=4, TM\_CCOEFF\_NORMED=5 };



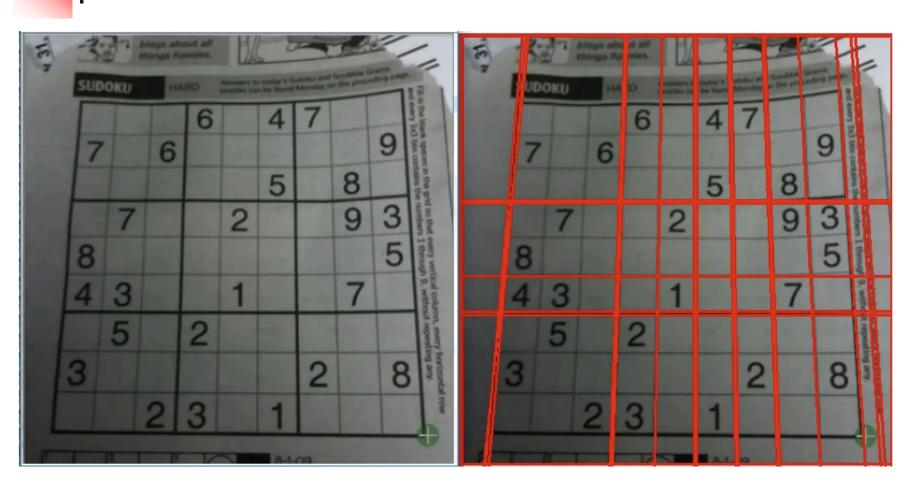
# ●模板匹配

TM_SQDIFF	$R(x,y) = \sum_{x',y'} (T(x',y') - I(x+x',y+y'))^2$		
TM_SQDIFF_NORMED	$R(x,y) = rac{\sum_{x',y'} (T(x',y') - I(x+x',y+y'))^2}{\sqrt{\sum_{x',y'} T(x',y')^2 \cdot \sum_{x',y'} I(x+x',y+y')^2}}$		
TM_CCORR	$R(x,y) = \sum_{x',y'} (T(x',y') \cdot I(x+x',y+y'))$		
TM_CCORR_NORMED	$R(x,y) = rac{\sum_{x',y'} (T(x',y') \cdot I(x+x',y+y'))}{\sqrt{\sum_{x',y'} T(x',y')^2 \cdot \sum_{x',y'} I(x+x',y+y')^2}}$		
TM_CCOEFF	$R(x,y) = \sum_{x',y'} (T'(x',y') \cdot I'(x+x',y+y'))$ where $T'(x',y') = T(x',y') - 1/(w \cdot h) \cdot \sum_{x'',y''} T(x'',y'')$ $I'(x+x',y+y') = I(x+x',y+y') - 1/(w \cdot h) \cdot \sum_{x'',y''} I(x+x'',y+y'')$		
TM_CCOEFF_NORMED	$R(x,y) = rac{\sum_{x',y'} (T'(x',y') \cdot I'(x+x',y+y'))}{\sqrt{\sum_{x',y'} T'(x',y')^2 \cdot \sum_{x',y'} I'(x+x',y+y')^2}}$		

TM\_SQDIFF, TM\_SQDIFF\_NORMED匹配数值越低表示匹配效果越好,其它四种反之。
TM\_SQDIFF\_NORMED, TM\_CCORR\_NORMED, TM\_CCOEFF\_NORMED是标准化的匹配,得到的最大值,最小值范围在0~1之间,其它则需要自己对结果矩阵归一化。











void HoughLines(InputArray image, OutputArray lines, double rho, double theta, int threshold, double srn=0, double stn=0);

### 参数详解:

image: 边缘检测的输出图像. 它应该是个灰度图 (但事实上是个二值化图)

lines: 储存着检测到的直线的参数对 的容器

rho: 参数极径 以像素值为单位的分辨率. 我们使用 1 像素.

theta: 参数极角 以弧度为单位的分辨率. 我们使用 1度 (即CV\_PI/180)

theta: 要"检测"一条直线所需最少的的曲线交点

srn and stn:参数默认为 0.





void HoughLinesP(InputArray image, OutputArray lines, double rho, double theta, int threshold, double minLineLength=0, double maxLineGap=0);

### 参数详解:

image: 边缘检测的输出图像. 它应该是个灰度图 (实际上是个二值化图)

lines: 储存着检测到的直线的参数对 的容器,也就是线段两个端点的坐标

rho: 参数极径 以像素值为单位的分辨率. 我们使用 1 像素.

theta: 参数极角 以弧度为单位的分辨率. 我们使用 1度 (即CV\_PI/180)

threshold: 要"检测"一条直线所需最少的的曲线交点

minLinLength: 能组成一条直线的最少点的数量. 点数量不足的直线将被抛弃.线段的最小长度

maxLineGap: 线段上最近两点之间的阈值



# ● OpenCV中的基本视频操作

# **VideoCapture**类

open (const <u>String</u> &filename) //打开视频 <u>isOpened</u> () const //检测打开是否成功

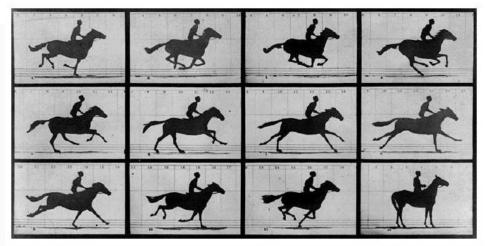
get (int propId) const //视频属性

operator>> (Mat &image) //取帧 read (OutputArray image)

grab () //缓存后取帧 retrieve (OutputArray image, int flag=0);

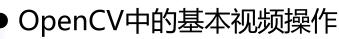
## release () //释放文件

- CAP PROP POS MSEC Current position of the video file in milliseconds or video capture timestamp.
- CAP\_PROP\_POS\_FRAMES 0-based index of the frame to be decoded/captured next.
- CAP\_PROP\_POS\_AVI\_RATIO Relative position of the video file: 0 start of the film, 1 end of the film.
- CAP\_PROP\_FRAME\_WIDTH Width of the frames in the video stream.
- CAP\_PROP\_FRAME\_HEIGHT Height of the frames in the video stream.
- CAP\_PROP\_FPS Frame rate.
- CAP\_PROP\_FOURCC 4-character code of codec.
- CAP\_PROP\_FRAME\_COUNT Number of frames in the video file.
- CAP\_PROP\_FORMAT Format of the Mat objects returned by retrieve() .
- CAP\_PROP\_MODE Backend-specific value indicating the current capture mode.









深圳大学 SHENZHEN UNIVERSITY

- ① 变量声明: VideoCapture capture
- ② 打开视频: capture.open(string filename)
- ③ 读取视频帧: capture>>fame, 其中fame是Mat变量
- ④ 判断视频是否已打开: capture.isOpened()
- ⑤ 判断帧是否为空: if(frame.empty())

```
string filename = "face_test.avi";//<u>打开的视频文件</u>
VideoCapture capture:
capture.open(filename);
double rate = capture.get(CV CAP PROP FPS);// 获取视频文化
int delay = cvRound(1000.000 / rate);
if (!capture isOpened())//判断是否打开视频文件
 printf("离开\n");
  exit(0);
else
  while (true)
    Mat frame:
    capture >> frame;//读出每一帧的图像
    if (frame.empty()) break;
   imshow("处理前视频", frame);
    //processiamge(frame);
   imshow("处理后视频", frame);
   waitKey(delay);
  视频读取关键代码
```

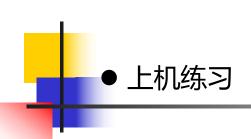


- ① 摄像头变量声明: VideoCapture capture(0)
- ② 摄像头是否成功打开: capture.isOpened()
- ③ 摄像头图像读取: capture.read(frame), 其中fame是Mat变量

```
VideoCapture capture(0);
if(!capture.isOpened())//检查摄像头是否成功打开
  printf("视频没有读取成功\n");
  exit(0);
//这里必须两个一起设置才能生效, 宽和高
capture.set(CV_CAP_PROP_FRAME_WIDTH,1920);//1080);
capture.set(CV_CAP_PROP_FRAME_HEIGHT,1080);//1080);
//capture.set(CV CAP PROP SHARPNESS,100);//1080);
Mat frame://当前视频帧
int i=0:
while(1)
  if(!capture.read(frame))
    break;
  imshow("video",frame);
  waitKey(30);
  if(i++>100)
    break:
```

摄像头读取关键代码



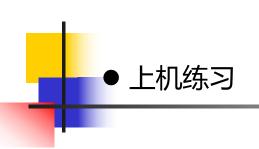


1.在图像中实现目标匹配; (先获取该目标的一个图像, 然后利用这个已知的图像, 在测试图像上找到该目标)

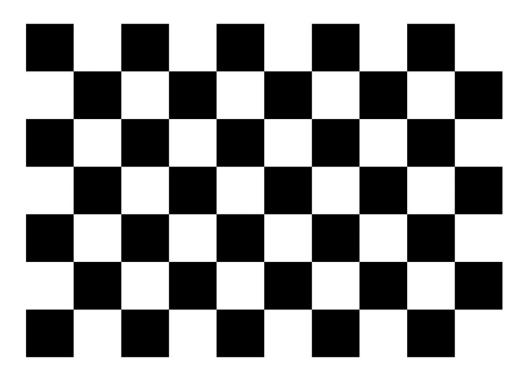






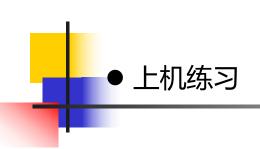


2.在图像上通过霍夫变换寻找直线;

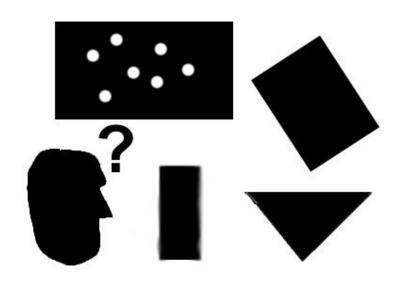


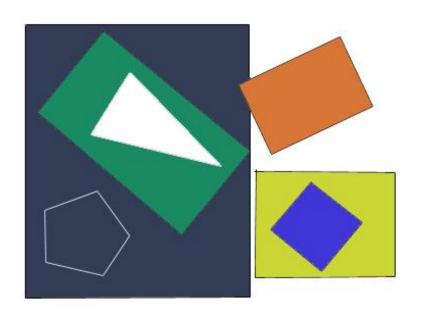
This is a 9x6 OpenCV chessbook



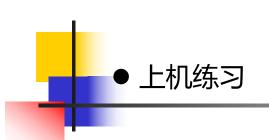


**3.**基于霍夫变换,实现三角形检测,如果是四边形呢?多边形呢?

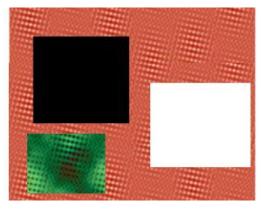


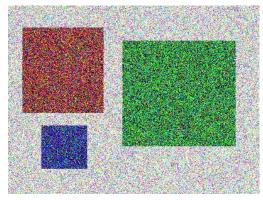


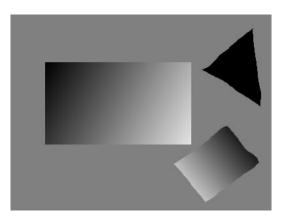


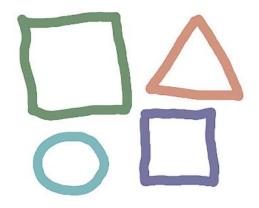


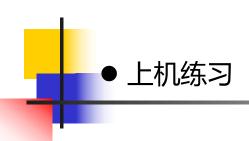
**4.**基于霍夫变换,实现三角形或矩形检测,如果遇到噪声? 边缘发生弯曲?













# 5.练习基于OpenCV的视频读取。





