狼牙战队 2024赛季算法组招新考核 任务

> 2023.9

说明

1. 对于大一的同学，我们希望你完成基础部分任务；对于大二及以上的同学，我们希望你完成进阶部分任务。
2. 在做这些任务的时候你可能会遇到很多困难，**在规定的时间内可能无法全部完成，这很正常。希望你不要轻易放弃，完成尽量多的任务。**可以跳着做。对于尝试但未完成的任务，同样提交已完成的部分和报告；对于没来得及做的任务，也可以记录你的思考。
3. 在**9.30晚11:59前**提交你的报告和工程源码，打包成压缩文件发送到张昭骏的**邮箱**1357299736@qq.com，先将文件放入一个文件夹，再将文件夹整体打包。此外，**文件夹内需要包含一个整体说明文件**，命名为“完成概况”，文件内说明你选择的题组、具体完成了哪些任务、你使用的开发环境（系统、IDE等）。

   邮件标题格式：狼牙战队24赛季算法组招新考核-专业年级-姓名。**文件夹命名格式**：专业年级-姓名-算法组.zip，如：电信22-张三-算法组.zip。
4. 关于记录，**每个任务都需要附上说明文件**，可以介绍代码实现思路、你认为的特色点、学习过程记录等等。必须包含**代码内容说明**（如一些比较重要的变量的作用、关键代码的逻辑）和**任务要求的输出记录（控制台截图及OpenCV图像输出截图，需对应到要求点）**，其他可以自行添加。也可以附上学习笔记等，能体现你学习过程的材料，注意保证文件及内容整齐、重点突出。
5. **文末有资料补充**供参考。
6. 有些任务有前后关系，但代码应该独立分开（不要直接在上一任务的代码上修改而只保留一份，要分成两份代码）。注意在完成阶段性功能后备份代码（不用提交备份）。

基础部分

如果还没有搭建好开发环境，建议使用Visual Studio，它的界面比较友好，且网上安装OpenCV并使用Visual Studio开发的教程也比较多，在“学习建议补充”中有相关介绍。

注意：每个需要输出的内容留下截图

任务一：排序

1. 不允许使用排序库函数（如std::sort），排序功能本体需要手写实现，其他如随机数生成、输入输出可以调用库函数。
2. 随机生成长度为10的整型一维数组，并输出在屏幕上。
3. 从冒泡或插入排序（基础）、快速排序（提高）中选择一种，实现升序排序。
4. 输出每一轮排序后数组的内容。
5. 输出最终数组的内容。
6. 添加程序注释，写明重要变量含义/作用、关键步骤逻辑。

任务二：矩阵类

计算机中，图像的本质是矩阵。就像无数微小的细胞构成可见的叶片，许多细微的像素排列成矩形，以一个整体显示出视觉信息。像素对应计算机内的数字，而数字排列为矩形就可以视作矩阵。

用 C++实现矩阵类。

要求：

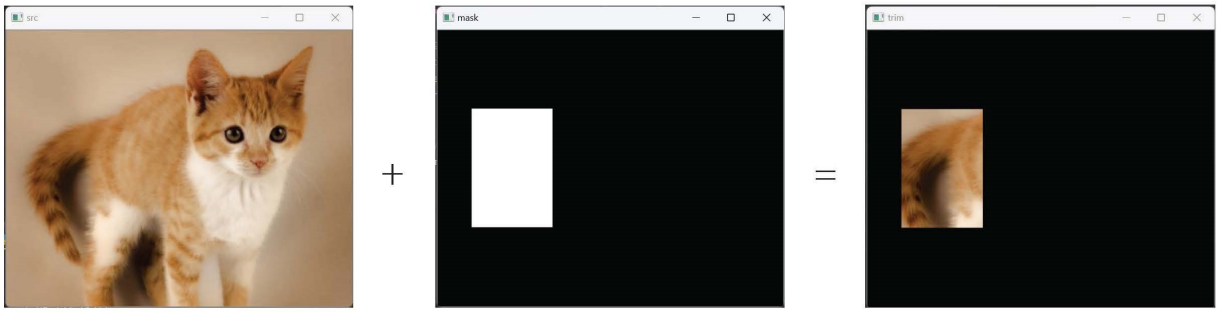
1. 数据成员和方法自行设计（提示：内部可以是二维数组），要求能通过构造函数指定产生的矩阵对象的长宽
2. 数据元素为整型。
3. 提供矩阵加法、减法、乘法功能，并能指定行列对元素赋值。对于不正确的（无法完成相应运算）输入，应该有提示信息。
4. 能够以矩阵的形状输出内容，元素间以空格分隔开。（按照整型输出）
5. 自己生成两个矩阵，输出原矩阵；依次完成3中的运算，并输出结果。

注：如果认为难以使用”类“完成任务，可以只用二维数组和一系列函数完成任务二、三。

任务三：用自己的矩阵类实现图像操作

图像就是矩阵。

掩膜（mask）是图像处理中的一种方法。如图所示，利用掩膜我们可以从图像中选出自己需要的部分，以便后续处理。实际上，掩膜记录的是一个区域，即位置信息；黑与白只是这一信息的一种直观表示，如何去解释和利用掩膜所携带的信息，正是编程者需要考虑的。



用任务三中的矩形类实现这一操作（用掩膜分割图像）。

要求：

1. 不限方法，将任务文件夹中的“image.txt”的数据（50*50矩阵）导入到矩阵类对象“src_image”。
2. 创建另一矩阵类对象"mask"， 将其作为mask，即将一个50*50矩阵元素全赋值为0，把图像中想要保留的部分对应的位置赋值为1（提示：调用任务三实现的赋值函数，二重循环赋值）。形如：

```
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 1 1 1 1 0 0 0 0
0 0 1 1 1 1 0 0 0 0
0 0 1 1 1 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0
```

3. 实现功能：类似上图所示，根据“mask”的信息将“src_image”中需要的部分保留，不需要的部分赋值为零。创建新的矩阵类对象“dst_image”保存该结果。
4. 将“src_image”、“mask”、“dst_image”分别用任务三实现的方法显示在屏幕上（即按整型数组显示）。

你看到的“图像”（实际是字符）可能很抽象，但接着做就能看到其本貌。

任务四：OpenCV的安装与Mat类

在Windows下结合Visual Studio安装OpenCV

网上教程很多。

https://blog.csdn.net/BRMBC/article/details/131762935

注意在系统环境变量（PATH）配置动态库（dll文件）路径，解决“找不到opencv_world_XXXd.dll”的问题。

https://blog.csdn.net/m0_66701835/article/details/125761693

如果使用Ubuntu，会需要进行更多环境配置，更加困难。推荐使用VSCode编写代码、CMake辅助编译代码。OpenCV安装和开发也都可以在网上找到教程。

要求：

1. 安装OpenCV，**记录**安装过程中遇到的问题、学到的知识或操作等。（如果你已经提前安装，则可介绍自己已经学到的内容）

2. 学习使用OpenCV的Mat类，构造一个Mat对象，并利用Mat类的at方法、指针或"迭代器"（进阶）将之前的50*50矩阵数据输入到Mat内。（注意"通道数"及数据类型、有无符号，你得到的应该是8位灰度图）
3. 使用imshow函数显示2中的Mat，保留截图。（提示：注意使用waitKey函数，可以思考一下原因）
4. 可以将图像放大一些

```
// 定义放大比例
double scale_factor = 8.0; // 放大两倍

// 调整图像尺寸
cv::Mat enlarged_image;
cv::resize(image, enlarged_image, cv::Size(), scale_factor, scale_factor);
```

附加任务：阈值分割

阈值分割是基本的图像操作之一，本质是筛选，选出数值在给定范围内的像素。通过这种方法，我们可以选出图像中特定颜色、特定亮度的区域。

通过这一任务，我们希望你不只是学习这阈值分割一种操作，而是在过程中总结学习使用新技术的有效方法。

要求：

1. 使用imread读取任务文件夹中的图片"car.png"并显示。
2. 了解色彩通道、RGB、阈值分割。将图像按BGR（OpenCV默认顺序为BGR）三通道进行阈值分割，使得蓝色部分被保留而红绿部分被削弱。显示得到的二值图像。（提示：可以使用inRange函数）
3. 调整阈值，使得到的二值图像中尽量只有车牌部分变成白色，其他区域尽可能为黑色。
4. 进阶：创建滚动条，动态调整阈值并实时查看效果。

进阶任务

任务一：矩阵类

用 C++实现矩阵类。

要求：

1. 数据成员和方法自行设计，要求能通过构造函数指定产生的矩阵对象的长宽，构造时动态申请内存。
2. 数据元素为整型。
3. 提供矩阵加法、减法、乘法、**求逆**功能，并能指定行列对元素赋值。应能判断非法输入。
4. 以矩阵的形状输出内容，元素间以空格分隔开。（按double输出）
5. 自己生成两个矩阵，输出原矩阵；依次完成3中的运算，并输出结果。
6. 进阶：支持三通道操作：在一个对象内维护三个相同大小的矩阵，分别进行加法、减法、乘法。

任务二：利用自己的矩阵类求坐标变换

从图像还原真实世界信息，离不开坐标变换。

*本部分内容建议结合《视觉SLAM十四讲》的“三维空间刚体运动”一章学习。

1. 了解旋转矩阵、“齐次坐标”。
2. 三维空间中旋转$R=\begin{bmatrix}0.22 & -0.43 & 0.88\\ 0.62 & 0.75 & 0.22\\ -0.75 & 0.5 & 0.43\end{bmatrix}$，平移$\vec{t}=\begin{bmatrix}-1\\ 9\\ 2.5\end{bmatrix}$，向量$\vec{a}=\begin{bmatrix}29\\ 6\\ 2.5\end{bmatrix}$，用你所实现的矩阵类表示，并求出 $\vec{a}$ 依次经过R、$\vec{t}$变换后得到的新向量。

   每一步输出格式：[矩阵实例1] [操作/运算符] [矩阵实例2] = [结果]
3. 使用齐次坐标（将 $\vec{a}$ 用齐次坐标表示）完成上述运算，输出计算过程，并将结果从齐次坐标恢复到三维坐标输出

任务三：OpenCV的安装与轮廓检测

要求：

1. 安装OpenCV，**记录**安装过程中遇到的问题、学到的知识或操作等。
2. 使用imread读取任务文件夹中的图片”QRcode.png“并显示。
3. 提取图片中的所有轮廓，画在原图上并显示。
4. 尝试从轮廓中筛选出二维码左上角”双层正方形“中内层的正方形，并用蓝色画出其外轮廓，你是如何做的?

任务四：灯条识别

装甲板的识别和自动击打是RoboMaster比赛中算法组的经典任务。装甲板的一大特征就是灯条，其颜色、尺寸等都可以作为识别的依据。

要求：

1. 读取并根据图片“armor.png”编写灯条识别程序，分开识别红色、蓝色灯条（可以切换模式），“识别”指判断出图像中是灯条的部分，“灯条”仅包括装甲板两侧的小灯条，而不包括横向大灯条（血量条）。
2. 在原图画出你所识别的灯条的外接矩形，并为每个灯条编号，将编号也显示在灯条旁边（显示文字）。（提示：使用颜色空间、阈值分割、边缘检测、轮廓检测等知识）
3. 分别读取视频“1b2r_armors.mp4“,试着在视频上使用你的识别程序分别识别蓝色、红色灯条，截取效果好和效果不好的帧，取有代表性的分析效果不好的原因。
4. 尝试优化识别程序在视频上的效果，说明你尝试的方法、思路、效果及结果分析。

附加任务：使用OpenCV的目标跟踪器

在目标运动过程中，不断变化的环境因素（如光照）可能使检测器丢失目标或误识别，这是图像色彩信息变化超出了检测器设计的预期。但除了同一帧图像的信息，我们其实还有时间轴上的信息（过去的图像）可以利用。

要求：

1. 了解OpenCV有哪些目标跟踪器，选择要使用的跟踪器并学习用法。

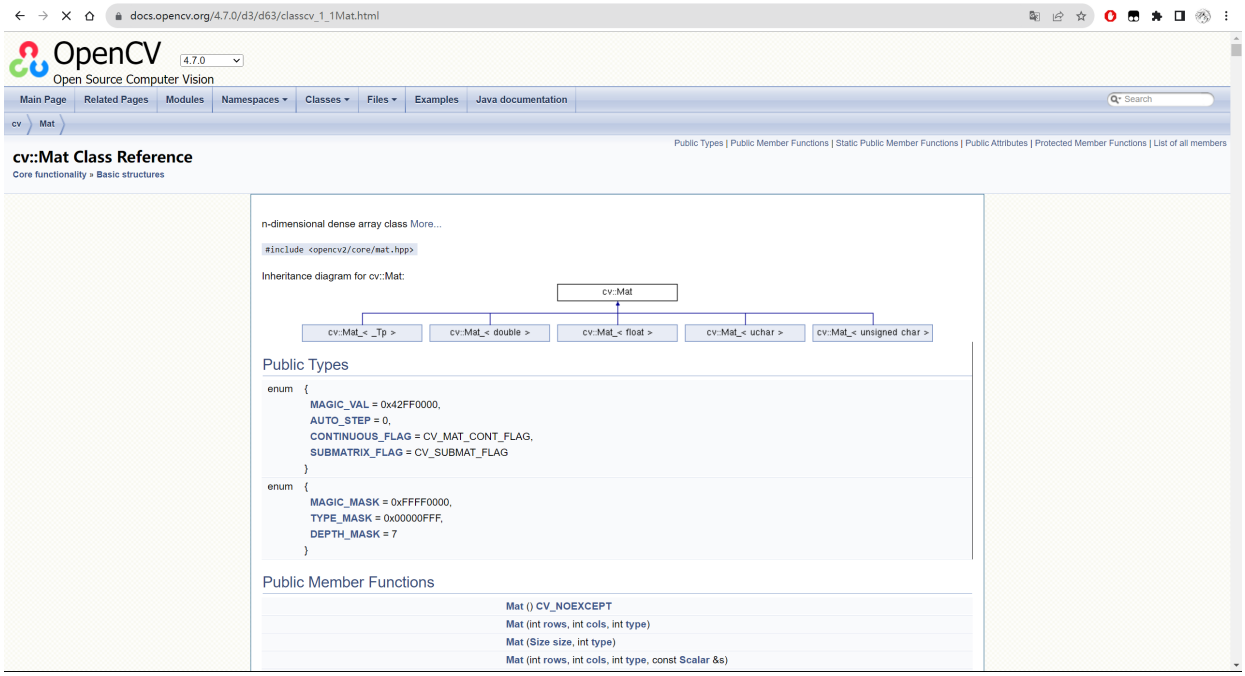
2. 跟踪单个目标，将视频“1b2r_armors.mp4”中蓝色3号车作为目标，将装甲板灯条共同的外接矩形作为整个车的识别框，取第21帧的识别框作为初始状态，初始化跟踪器。
3. 每帧画出自己检测器的检测框和由跟踪器得到的检测框（不必保存提交）。
4. 如果存在目标丢失，截取相应输出并尝试分析原因。
5. 添加跟踪器的重新初始化逻辑，用自己检测器的结果适时更新跟踪器初值。
6. 结合检测器与跟踪器结果，增加输出每帧最终认为的装甲板检测框，截取有代表性的帧加入说明文件。
7. 尝试在“1b2r_armors.mp4”对红色车使用增加了追踪器的检测器，会出现什么问题？将装甲板依次编号并在结果框边画出。

资料补充

OpenCV文档

有时网上找到的资料质量参差不齐甚至有误，如果对要使用的函数、类等找不到有效的资料，或是对OpenCV的概念定义有疑惑，可以参考OpenCV官方文档。OpenCV的所有类、函数都在其文档中有说明，但官方文档可能不太好找。

链接： https://docs.opencv.org/4.7.0/d3/d63/classcv_1_1Mat.html



左上角可以选择版本，右上角搜索。

齐次坐标

这又是一个突然出现的概念。

学习过线性代数的同学应该了解，如果以三维向量表示三维空间的点，那么可以用三维向量表示点的平移（向量相加），此外还可以用矩阵表示向量旋转（乘法）。向量也是矩阵，在上述操作中，矩阵表示了平移和旋转这两种变换。

可以发现，表示同一向量旋转和平移的矩阵形状不同、其与向量进行的运算也不同。算法组主要通过相机获取真实世界的信息，直接获取的也是相机和世界的位置关系。而对于机器人上的相机，平移（质心位置在空间的移动）和旋转（朝向的变化）往往同时发生。从某一时刻t1到下一时刻t2，相机的运动需要用两个矩阵来描述。

但我们对于机器人的状态也往往以时间划分，两个时刻对应两个状态。我们希望从一个状态到下一个状态之间的变换能够只用一个矩阵就描述出来，于是引入“齐次坐标”的概念。

齐次坐标（以及变换矩阵）可以视作一种计算技巧，只是换了一种书写算式的形式。