2024～2025 届浑南智能车实验室竞速组招新考试

视觉组

试题说明：本试卷满分 120 分，共三大题。代码无特殊要求使用 C++编写，并编写 README.md。

一、系统设计题。（60 分）

系统设计：**基于 ROS2 的智能车无 GUI 视觉系统的设计**

1、原始图像数据发布功能包的设计

使用 GStreamer 等高效的库来读取摄像头原始数据（640x480，30fps），并将其手动编码为 NV12 格式储存起来并通过自定义消息发布图像数据，话题为/image_publisher。

2、图像数据的预处理（发布的话题格式自行选择）

订阅图像数据发布的主题，将接收到的图像数据解码并转换为 OpenCV 可处理的格式。

预处理 1：编写预处理函数 red_process，函数实现了从图像中提取 R 通道，按照灰度图的原理设计并生成一张“红度图”；

预处理 2：在红度图的基础上手动实现高斯滤波函数，并在 GPU 上对该函数进行加速；

预处理 3：在上一步的图像中，设计一套提取特征提取函数，使用 ORB 算法从图像中提取 500 个特征点，并使用黄色标记这些特征点，将该图片发布话题为/web_pub；

3、图像的 web 端展示

将上述图像和处理信息通过局域网发布到 8080 端口，设计 Web 页面展示，并提供以下功能：在界面左上角提供复选框，供用户选择是否启用以下功能：是否显示红度图、应用高斯滤波、标记特征点；默认只显示原图，其他优先级和逻辑等自行拓展。

提交要求：

1、提交源码文件夹与使用录屏，可以使用一段长的视频代替摄像头调用；

2、提交一张截图：运行所有节点后截图 ros2 topic hz /web_pub。

二、数据通信题。（40 分）

在比赛中，不同模块间的数据传输至关重要。除了 ROS 提供的通信机制外，我们还可以采用其他方式实现数据的传递，例如通过 CSV 文件进行数据的写入与读取。

我们现在使用 python 代码 pub.py **生成并持续发布**（发布速率 15 帧）模拟 Yolo 识别后的数据：

1、数据的随机生成：假设有一张大小为 640×480 的图像，发布的识别框数据包括：

坐标数据：(x, y, w, h)，其中 w > 50 且 h > 100，保留整数；

类别：共三类，分别为 0（标志牌 A）、1（标志牌 B）、2（红绿灯）；

置信度：在 0.30 ~ 0.90 范围内随机生成，保留两位小数。

2、数据的展示与储存：

在终端中使用红色字体输出发布以下数据：时间戳、类别原名、坐标数据、置信度，例如：

时间戳: 2025-01-25 12:30:45，类别：标志牌 A，坐标: (100, 150, 60, 120)，置信度: 0.85

时间戳: 2025-01-25 12:30:45，类别：红绿灯，坐标: (200, 100, 80, 150)，置信度: 0.78

然后我们将类别号、坐标数据、置信度写入 yolo.csv 文件中，例如：

0, 100, 150, 60, 120, 0.85

2, 200, 100, 80, 150, 0.78

然后我们使用 C++代码 sub.cpp 来**读取这些数据**，读取后在终端中使用蓝色字体输出发布：类别原名、坐标数据、置信度，例如：

类别：标志牌 A，坐标: (100, 150, 60, 120)，置信度: 0.85

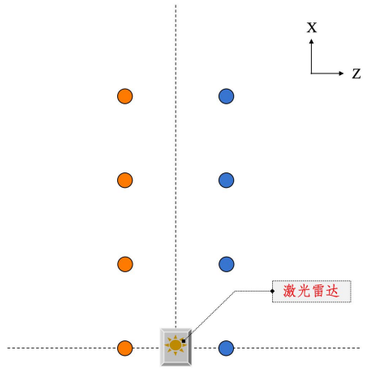
类别：红绿灯，坐标: (200, 100, 80, 150)，置信度: 0.78

注意事项：写入和读取要协同工作；随机数据生成要符合逻辑；时间戳的展示方式请自行选取；

提交要求：请提交源码文件夹与运行截图（截图需要展示 csv 文件的内容，以及发布与接收的终端输出）

三、算法编程题。（20 分）

在使用二维激光雷达进行障碍物检测时，常常会遇到以下问题：点云数据量大且密集；存在孤立点等噪声数据；因此我们采用点云聚类算法对点云进行处理和分组。其中，DBSCAN 是一种基于密度的聚类算法，通过点云的密度标准将数据划分为不同的点云簇，同时能够有效识别噪声点。



假设在如图所示的情景中，使用一种**二维激光雷达**进行扫描扫描范围为 1.5 米；通道宽度恒定为 1.5 米，同侧障碍物间隔 2 米。我们定义前进方向为 X，水平向右为 Z，竖直向上为 Y；请使用 DBSCAN 算法对以下三帧数据（我们只取 X 大于等于 0 的数据）进行聚类处理，并在终端中使用黄色字体输出当前数据的关键点簇的中心坐标

第一帧：

(0.028, 0.014, 0.759)　(-0.008, -0.011, 0.737)

(0.013, -0.006, 0.767)　(0.014, -0.011, 0.754)

(-0.003, 0.015, 0.747)　(0.005, -0.007, -0.743)

(0.022, 0.003, -0.749)　(0.009, 0.001, -0.738)

(0.011, -0.002, -0.762)　(0.007, 0.009, -0.756)

第二帧：

(0.532, 0, -1.231)　(1.320, 0, 0.972)

(0.874, 0, -0.413)　(0.481, 0, 1.281)

(1.112, 0, 0.542)　(0.328, 0, -1.273)

(1.374, 0, -0.932)　(1.001, 0, 0.658)

(0.415, 0, -1.349)　(1.267, 0, 1.438)

第三帧：

(1.119, -0.005, 0.744)　(1.087, 0.010, 0.752)

(1.105, -0.007, 0.735)　(1.091, 0.013, 0.759)

(1.098, -0.008, -0.734)　(1.110, 0.009, -0.746)

(1.083, 0.006, -0.754)　(1.101, -0.011, -0.763)

(1.307, 0, 0.542)　(0.411, 0, -1.321)

编写要求：可以自己编写算法或者调用已有的库；其他输出自行发挥。

提交要求：源码+结果运行截图