默认针对相机已运行run\_00\_calibrate\_camera.py，获得了yaml文件，xmap\_\*, ymap\_\*

**代码运行：**

运行run.py即可

结果参数：

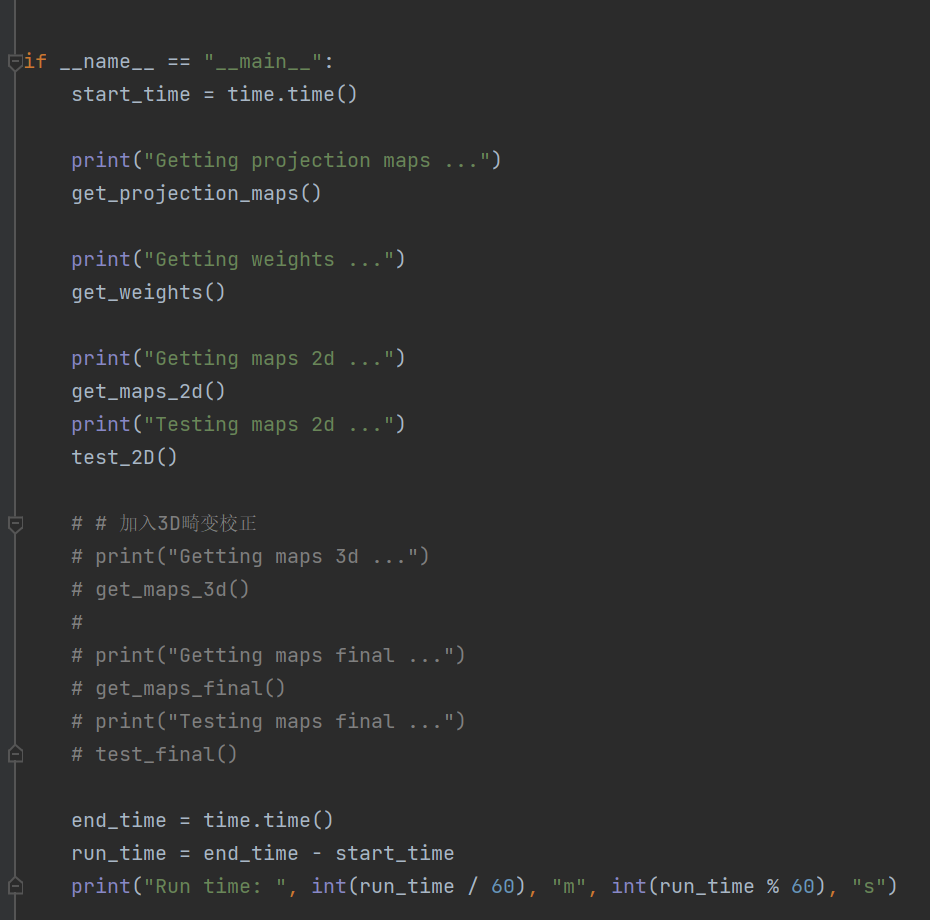
命令行界面的ROIW, ROIH

maps目录下的

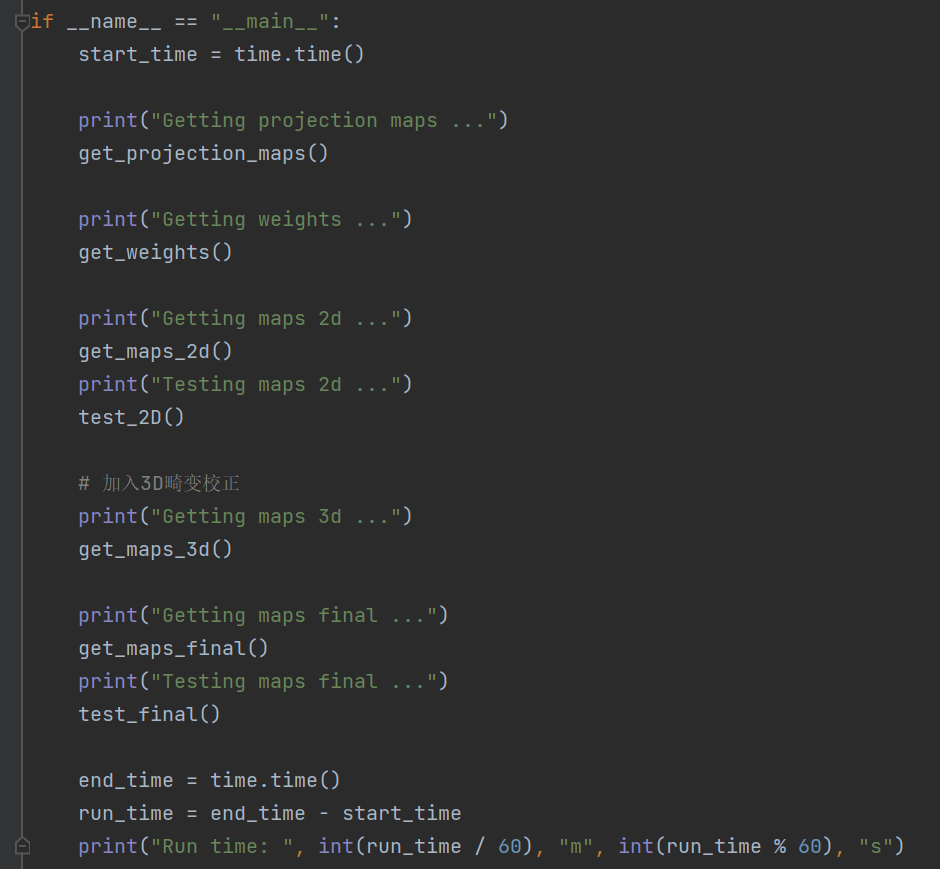
nearest\_0-3.bin

weight\_0-3.txt

如下注释可得到2D拼接参数



如下不注释可得到3D拼接参数



**可调整参数：**

surround\_view/param\_settings.py

**输出拼接结果：**

results/result\_2d.jpg

**源图片路径**

images\_root = "images/img5\_20230314"

**确定视野范围：**

per\_size = 50 # 棋盘格尺寸，数值越大，画面视野越小，值大概在10~70

**确定车长宽比例，如武警两栖车摄像头之间的长宽距离比例为2.357：**

car\_w\_num = 1 # 汽车横向占棋盘格数量

car\_h\_num = car\_w\_num \* 2.357 # 汽车纵向占棋盘格数量

**确定棋盘格区域大小，注意保持与实际一致：**

w\_num = 11.8 #横向棋盘格总数量

h\_num = 18.7 #纵向棋盘格总数量

尽量如下公式保持标定板之间的空隙的长宽比例与车长宽比例一致，比如

对于4m\*4m的标定板，2.8m的摄像头横向间距，6.6m的摄像头纵向间距，标定区域整体的长宽为：

宽：4+2.8+w+4，长4+6.6+w\*(6.6/2.8)+4

其中的w即为标定板与摄像头的摆放间距。为了尽量将标定板摆放至摄像头视野中央，对于两栖车摄像头的高度和角度，建议w取1.2m比较合适, 即标定区域外围长**17.4m**（车辆前后方向），宽**12m**（车辆左右方向）。

**前后摄像头画面的权重比增加比例**

weight\_front\_back\_b = 50 # 越大拼接羽化范围越小，b越大，拼接缝隙越靠近左右摄像头，即画面以前后摄像头内容为主

weight\_front\_back\_a = 1 # 越大拼接羽化范围越小，a越大，拼接缝隙越靠近前后摄像头，即画面以左右摄像头内容为主

**# 选点信息**

manual\_project = 0 # 是否需要人工选点，1为需要，0为不需要

# 若要自动选点，需要debug运行run.py，手动将选点信息保存至param\_settings.py中，参照src\_gz\_5\_new 数据，并更改 src 值

在保存选点信息后，后续可通过微调选点坐标，来微调拼接效果。

**# 改变透视偏移值**

# 透视偏移值正数两侧向内凹，视野更大；负数两侧向外凸，视野更小，但可以与头尾角度融合更好

# 缩进值，正数压缩，视野变大；负数拉伸，视野变小

# 头透视偏移值

offset\_t\_r = 0

# 尾透视偏移值

offset\_w\_r = 0

# 头缩进

offset\_t\_h = 0

# 尾缩进

offset\_w\_h = 0

# 左右透视偏移值

offset\_zy\_r = 0

# 左右缩进 default = offset\_tw\_r

offset\_zy\_h = 0

其余参数在代码中均有注释。

param\_settings\_3d.py

本文件调整3D畸变矫正，是否需要添加看实际情况。

**输出拼接结果：**

results/result\_final.jpg

**3D模型参数：**

# 图片规格1080p，一个棋盘格对应像素数 >9px

# 收缩偏移量

shift = 220

# 边缘收缩速率

compress = 4

# 横向缩放倍数

x\_scale = 2.2

# 纵向缩放倍数

y\_scale = 2.2

# 半径大小为变成的 1/r\_ratio

r\_ratio = 8

# 窗口调试 1调试，0不调试 ， 由于窗口调试运行过慢，不建议用

test\_loop\_3d = 0

其余参数已在代码中作详细注释

**函数说明：**

get\_projection\_maps()

输入：无

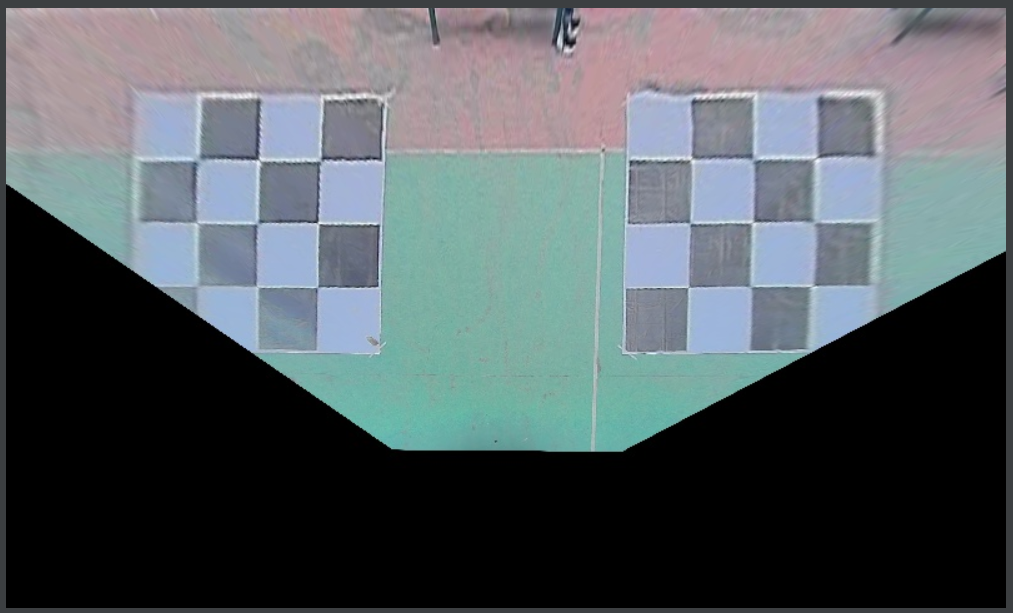
输出：该相机的画面畸变校正后的x坐标投影矩阵，y坐标投影矩阵

需要操作：第一次时需要手动标记作为变形参照物的矩阵四个角特征点

输入依赖文件：\*.yaml, param\_settings.py

输出保存文件：\*.yaml, xmap\_\*.txt(int16), ymap\_\*.txt(int16), project\_\*.jpg, warped\_2d\_\*.jpg

功能：获取每个相机的投影矩阵，以将画面变形为适合拼接的形状（此处近景周视系统中为鸟瞰图形状）



get\_weights()

输入：无

输出：拼接权重

需要操作：无

输入依赖文件：\*.yaml, param\_settings.py, 相机照片

输出保存文件：weight\_\*.txt(float32)

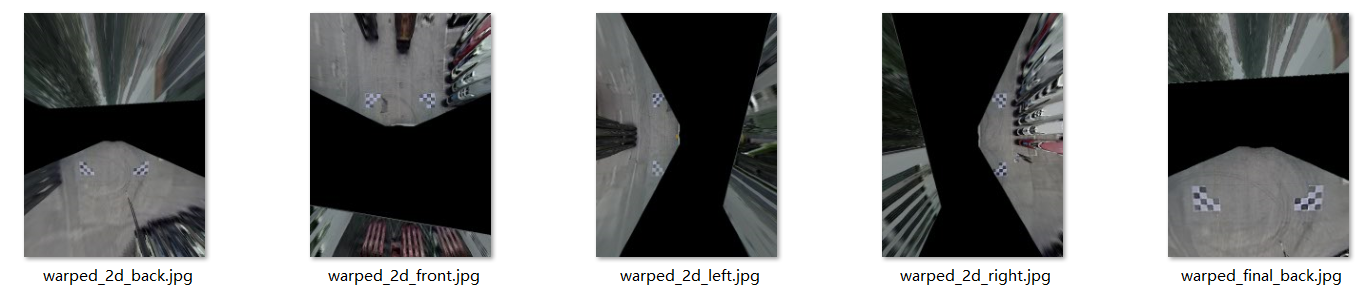
功能：获得所有相机的图像的拼接与平滑矩阵

get\_maps\_2d()

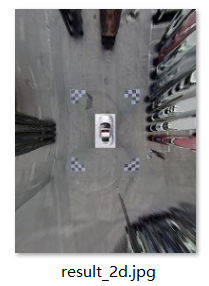
保存2D情况下的拼接映射表

test\_2D()

测试未经投影的映射表，生成warped图片，保存为results目录下的“warped\_2d\_\*.jpg”



生成对应的全景图，“result\_2d.jpg”



get\_maps\_3d()

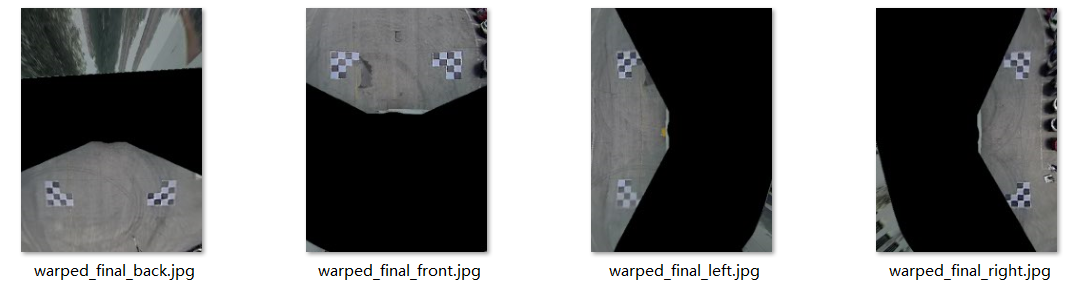
获取该相机经过鱼眼透视及投影的 投影结果图，经过3D投影 的映射表，保存为maps目录下的“map\_3d\_\*.bin”

get\_maps\_final()

整合鱼眼透视投影映射表与3D投影映射表，保存为maps目录下的“map\_final\_\*.bin”

test\_final()

测试最终整合映射表的拼接融合效果，保存为results目录下的“warped\_final\_\*.jpg”



最终全景结果为“result\_final.jpg”

