

# 1. 实验名称及目的

## 1.1. 实验名称

简易yolo传感器使用实验

## 1.2. 实验目的

通过简易yolo传感器识别无人机，熟悉其配置方法，该类传感器可应用于大规模仿真训练

## 1.3. 关键知识点

本实验主要是实现通过Python接口VisionCaptureApi.py（见RflySimAPIs\RflySimSDK\vision目录）获取RflySim3D图像并实时更新相机参数（姿态、位置、FOV等），然后通过yolo算法检测无人机，并通过平台接口调用cv库输出传感器图像。关键代码解析如下：

### 1) 视觉接口使用

```
1 | 创建视觉传感器实例：vis = VisionCaptureApi.VisionCaptureApi()  
2 |  
3 | 加载传感器配置文件：vis.jsonLoad()  
4 |  
5 | 发送取图请求：isSuss = vis.sendReqToUE4()  
6 |  
7 | 开启图像采集：vis.startImgCap()
```

### 2) 相机数量和参数配置

其中，视觉传感器的初始状态由本文件夹下的Config.json决定，主要包含以下配置项：

```
1 | "SeqID":0：使用自动更新ID的方式，共8个传感器（4个简易视觉传感器和4个RGB传感器）  
2 |  
3 | "TypeID":30：传感器类型为简易视觉传感器注，本文件夹下的Config.json中还配置了4个TypeID为1的RGB传  
4 |  
5 | "otherParams":[100,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]，传感器识别距离
```

### 3) 飞机控制指令

接口详细使用方法见：

```
1 | 创建飞机控制实例：使用 PX4MavCtrl.PX4MavCtrl() 创建多个控制实例，用于控制多架飞机。  
2 |  
3 | 初始化数据接收：mav[i].InitMavLoop() 启动 MAVLink 数据接收循环。  
4 |  
5 | 解锁与起飞：通过 SendMavArm(True) 和 sendMavTakeOff() 控制飞机解锁并起飞到目标位置。
```

### 4) UE控制

接口详细使用方法见：[UE4CtrlAPI.py](#)

```
1 | 创建 UE4 控制实例：ue = UE4CtrlAPI.UE4CtrlAPI()  
2 |  
3 | 设置窗口分辨率和刷新频率：通过 sendUE4Cmd 设置显示窗口分辨率和最大刷新频率，以优化资源使用。
```

### 5) 图像采集与处理

while True:之后这部分代码主要是在一个循环中将传感器采集的图像输出

采集图像并处理目标检测：循环中通过视觉传感器获取图像数据，对检测到的目标绘制矩形框并标注

CopterID。

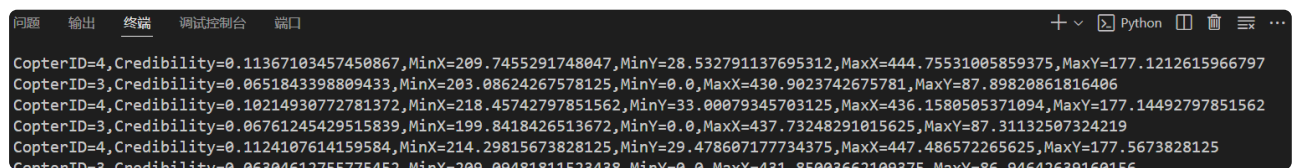
显示标注后的图像：使用 cv2.imshow 显示包含标注的图像，实现实时观察。

防止抖动干扰：初始化姿态角 (LastSensorAngEular) 来减少传感器数据抖动。

控制频率设置：按 30Hz 控制循环的执行频率，保证系统稳定运行。

## 2. 实验效果

在终端中返回识别目标的位置信息



```
问题 输出 终端 调试控制台 窗口 + Python 11 100% ...  
CopterID=4,Credibility=0.11367103457450867,MinX=209.7455291748047,MinY=28.532791137695312,MaxX=444.75531005859375,MaxY=177.1212615966797  
CopterID=3,Credibility=0.0651843398809433,MinX=203.08624267578125,MinY=0.0,MaxX=430.9023742675781,MaxY=87.89820861816406  
CopterID=4,Credibility=0.10214930772781372,MinX=218.45742797851562,MinY=33.00079345703125,MaxX=436.1580505371094,MaxY=177.14492797851562  
CopterID=3,Credibility=0.06761245429515839,MinX=199.8418426513672,MinY=0.0,MaxX=437.73248291015625,MaxY=87.31132507324219  
CopterID=4,Credibility=0.1124107614159584,MinX=214.29815673828125,MinY=29.478607177734375,MaxX=447.486572265625,MaxY=177.5673828125  
CopterID=3,Credibility=0.0630461275575452,MinX=209.00481811523438,MinY=0.0,MaxX=431.85003662109375,MaxY=86.94642639160156
```

## 3. 文件目录

例程目录：

[安装目录]\RflySimAPIs\8.RflySimVision\2.AdvExps\e7\_ObjDetectYolo\SimpleSensorBaseOnYolo

文件夹/文件名称	说明
VisionCtrlDemo.py	Python实验脚本
Config.json	视觉传感器配置文件
CameraCtrlApi.py	视觉取图接口
Python38Run.bat	Python程序运行脚本

## 4. 运行环境

### 4.1 软件要求

Windows 10及以上版本；RflySim工具链；VS Code。

①：若使用Pixhawk 6X飞控，平台安装时的编译命令为：px4\_fmU-v6x\_default，推荐PX4固件版本为：1.12.3。其他配套飞控及编译命令请见：

<https://rflsim.com/doc/zh/1/Hardware.html>

### 4.2 硬件要求

笔记本/台式电脑① 1台。

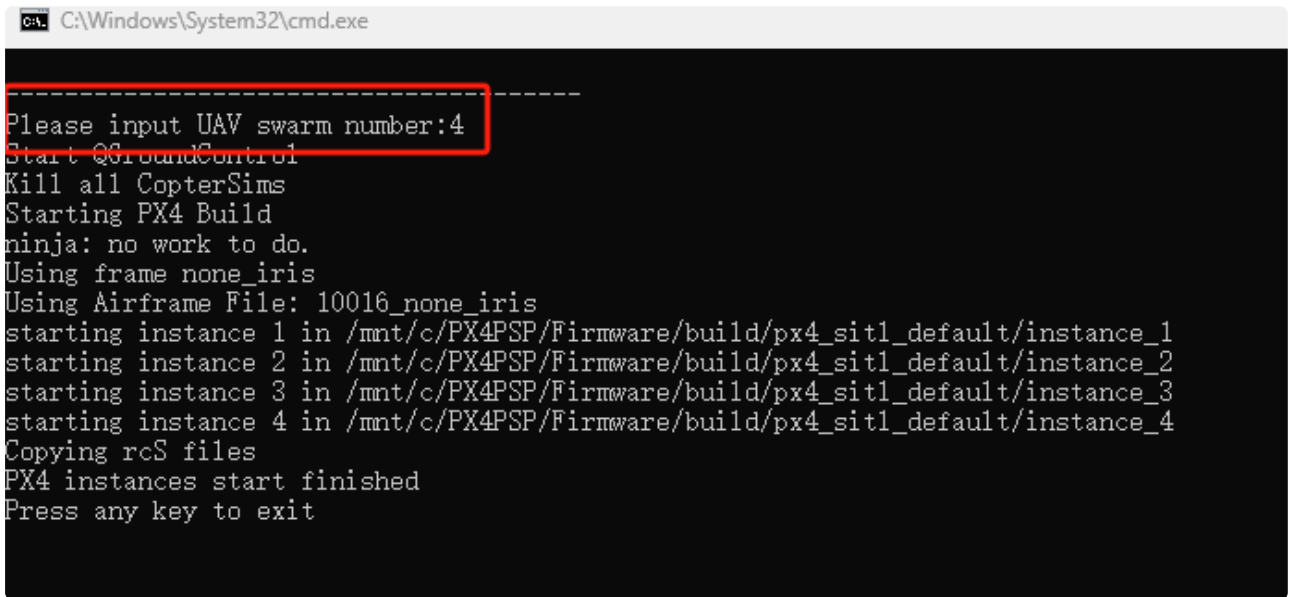
①：推荐配置请见：<https://rflsim.com/doc/zh/HowToInstall.pdf>

# 5.实验步骤

## 5.1.必做实验：Windows取图控制

### Step 1：启动软件在环

双击运行\Desktop\RflyTools目录下的SITLRun.bat文件，启动4架飞机的软件在环仿真；



```
C:\Windows\System32\cmd.exe

Please input UAV swarm number:4
Start QGroundControl
Kill all CopterSims
Starting PX4 Build
ninja: no work to do.
Using frame none_iris
Using Airframe File: 10016_none_iris
starting instance 1 in /mnt/c/PX4PSP/Firmware/build/px4_sitl_default/instance_1
starting instance 2 in /mnt/c/PX4PSP/Firmware/build/px4_sitl_default/instance_2
starting instance 3 in /mnt/c/PX4PSP/Firmware/build/px4_sitl_default/instance_3
starting instance 4 in /mnt/c/PX4PSP/Firmware/build/px4_sitl_default/instance_4
Copying rcS files
PX4 instances start finished
Press any key to exit
```



注：这里之所以启动4架飞机的软件在环，是由于config.json中将对应的传感器分别绑定到了4架载具上

## Step 2: 启动简易传感器

在本文件夹下，双击 `Python38Run.bat`，打开默认集成好的python环境，在该环境下运行 `VisionCtrlDemo.py` 文件，输入 `python [VisionCtrlDemo.py](VisionCtrlDemo.py)`

```
C:\Windows\system32\cmd.exe x + v
Python3.8 environment has been set with openCV+pymavlink+numpy+pyulog etc.
You can use pip or pip3 command to install other libraries
Put Python38Run.bat into your code folder
Use the command: 'python XXX.py' to run the script with Python
F:\d2\8.RflySimVision\2.AdvExps\e7_ObjDetectYolo\SimpleSensorBaseOnYolo>python VisionCtrlDemo.py
```

## Step 3: 观察结果

此时在RflySim3D中观察到四架飞机起飞到相同高度



有传感器默认对准机头前方，在图像窗口中可见绑定在copter1和copter2飞机上的传感器分别识别到了copter3和copter4.



在step2打开的终端可以看到识别目标的信息

```
问题 输出 终端 调试控制台 窗口 Python 终端
CopterID=4,Credibility=0.11367103457450867,MinX=209.7455291748047,MinY=28.532791137695312,MaxX=444.75531005859375,MaxY=177.1212615966797
CopterID=3,Credibility=0.0651843398809433,MinX=203.08624267578125,MinY=0.0,MaxX=430.9023742675781,MaxY=87.89820861816406
CopterID=4,Credibility=0.10214930772781372,MinX=218.45742797851562,MinY=33.00079345703125,MaxX=436.1580505371094,MaxY=177.14492797851562
CopterID=3,Credibility=0.06761245429515839,MinX=199.8418426513672,MinY=0.0,MaxX=437.73248291015625,MaxY=87.31132507324219
CopterID=4,Credibility=0.1124107614159584,MinX=214.29815673828125,MinY=29.478607177734375,MaxX=447.486572265625,MaxY=177.5673828125
CopterID=3,Credibility=0.0630461275575452,MinX=209.08481811523438,MinY=0.0,MaxX=431.85003662109375,MaxY=86.94642639160156
```

## 5.2.选作实验 (VS Code调试运行)

### 准备工作:

- 先确保已经按 [RflySimAPIs\1.RflySimIntro\2.AdvExps\3.PythonConfig\Readme.pdf](#) 步骤，正确配置VS Code环境。或者配置了自己的Pycharm等自定义Python环境。
- 其他步骤与上文相同，在Step3运行VisionCtrlDemo.py时，可使用VS Code（或Pycharm等工具）来打开VisionCtrlDemo.py文件，并阅读代码，修改代码，调试执行等。

## 扩展实验：

- 请自行使用VS Code阅读 [VisionCtrlDemo.py](#) 源码，通过程序跳转，了解每条代码的执行原理；再通过调试工具，验证每条指令的执行效果。

🔗 VisionCapAPIDemo.py ×

```
> RflySimAPIs > 8.RflySimVision > 0.ApiExps > 1-UsageAPI > 0.VisionSensorAPI > 1.Camera
8   ue = UE4CtrlAPI.UE4CtrlAPI()
9
10  #Create a new MAVLink communication instance, UDP sending
11  mav = PX4MavCtrl.PX4MavCtrl(1)
12
13  # The IP should be specified by the other computer
14  vis = VisionCaptureApi.VisionCaptureApi()
15
16  # Send command to UE4 Window 1 to change resolution
17  ue.sendUE4Cmd('r.setres 1280x720w',0) # 设置UE4窗口分辨率，设置
18  ue.sendUE4Cmd('t.MaxFPS 30',0) # 设置UE4最大刷新频率，同时也
19  time.sleep(2)
20
21  # VisionCaptureApi 中的配置函数
22  vis.jsonLoad() # 加载Config.json中的传感器配置文件
--
```

- 请尝试修改代码，实现飞机位置改变、相机姿态角改变、相机参数改变等功能。

## 6.参考资料

无

## 7.常见问题

Q1: 无

A1: 无