目录

一．介绍

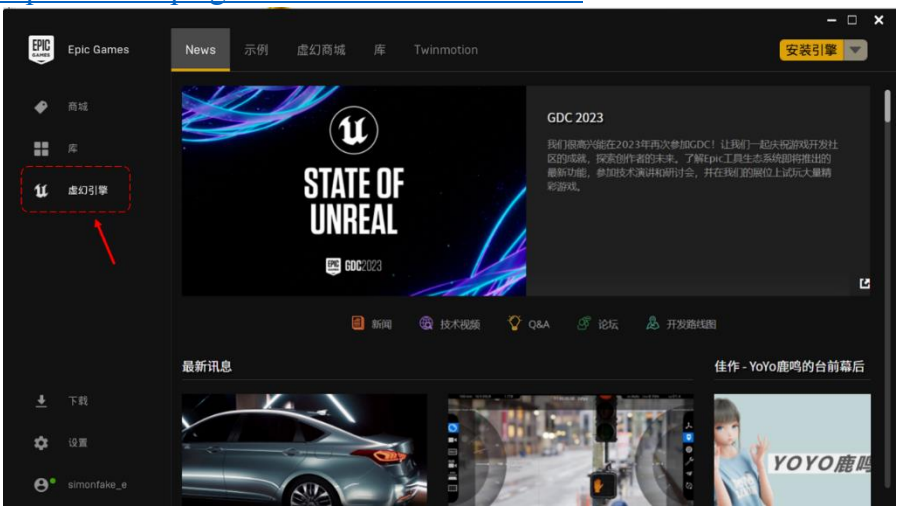
AirSim 是微软创建的一款基于虚幻引擎 UNREAL ENGINE 构建的无人机、汽车等模拟器（有 Unity 版本，但还没放出来）。AirSim 是开源的，跨平台的，并支持使用流行的飞行控制器（如 PX4 和 ArduPilot）进行软件在环仿真，并使用 PX4 进行硬件在环模拟，以进行物理和视觉上逼真的模拟。

微软的目标是将 AirSim 开发为人工智能研究平台，以试验自动驾驶汽车或无人机的深度学习，计算机视觉和强化学习算法。为此，AirSim 公开了以独立于平台的方式检索数据和控制载具的 API。

二. 安装

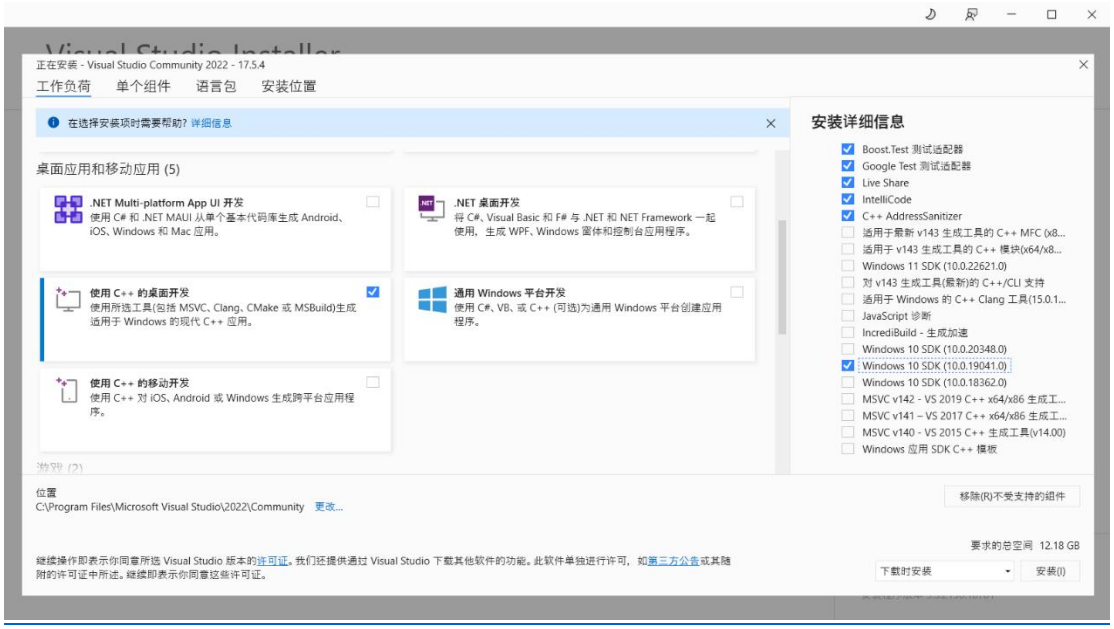
Unreal Engine

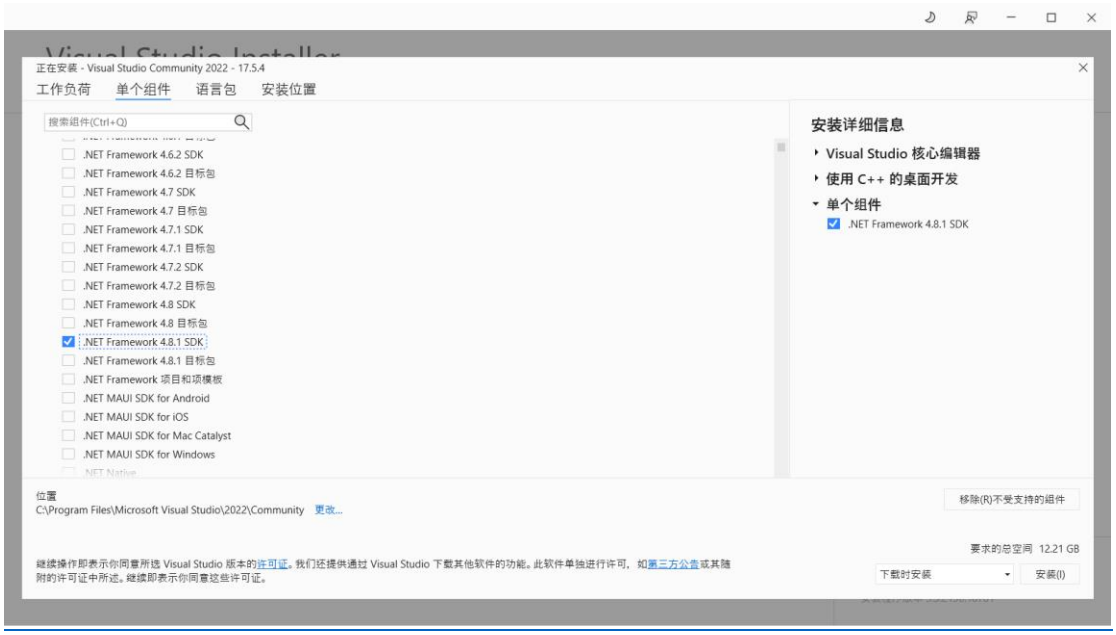
下载安装 EPIC GAMES 客户端，注册登录后安装 Unreal Engine 4.27 以上版本（推荐 4.27 版本）
地址：https://www.epicgames.com/site/zh-CN/home



Visual Studio 2022

安装 `C++桌面开发` 和 `Windows 10 SDK 10.0.19041` 以及在单个组件中选择最新版的 `.NET Framework SDK`
地址：https://visualstudio.microsoft.com/zh-hans/vs/





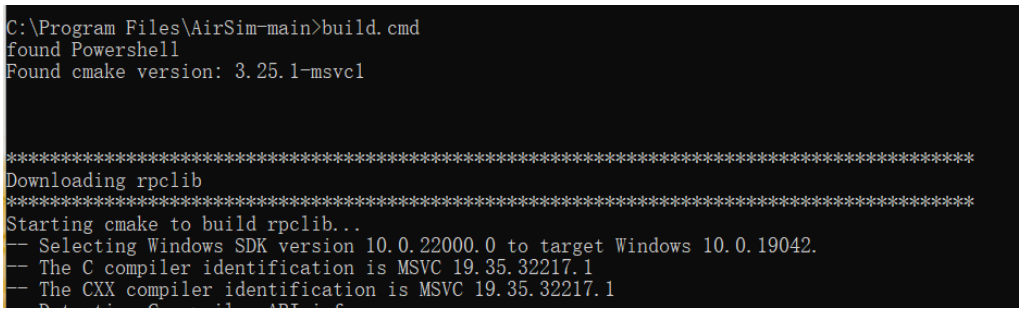
Git

最新版本就行
https://git-scm.com/download/win

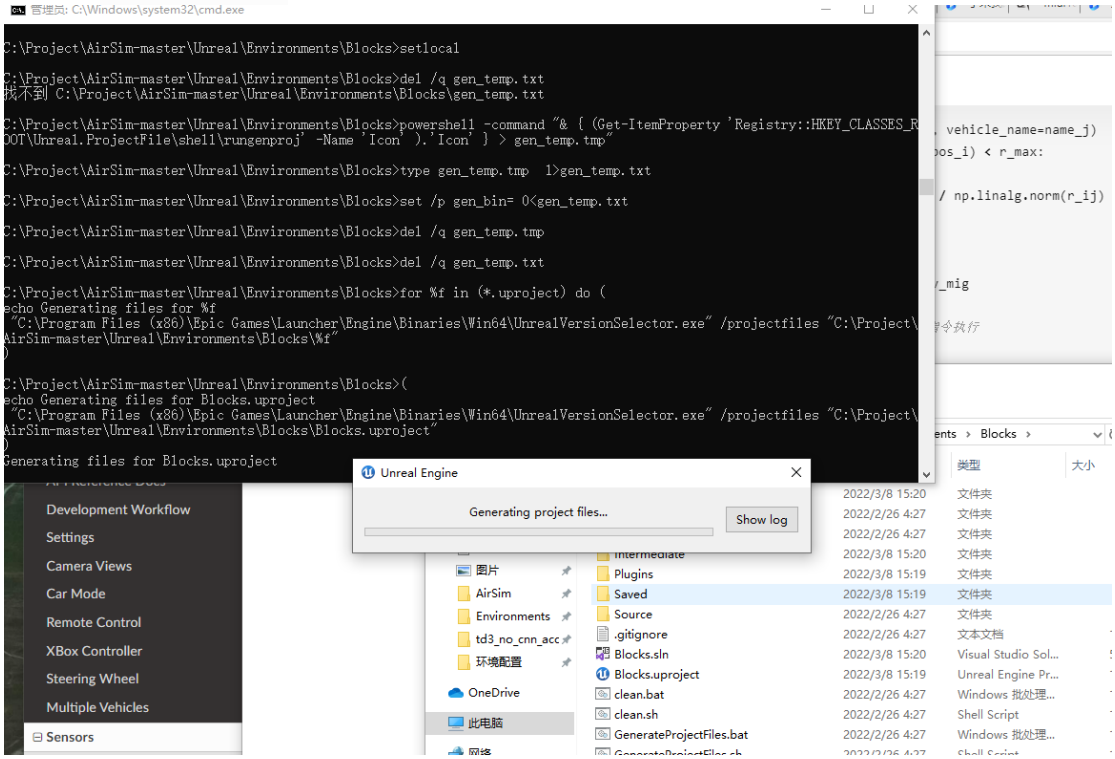
Airsim

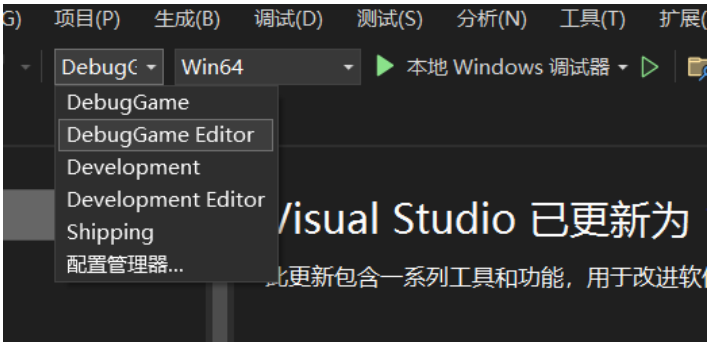
1. 打开 VS2022 的 `Developer Command Prompt for VS 2022`，用命令行下载 AirSim：`git clone https://github.com/Microsoft/AirSim.git` （网络不好的话直接在 GitHub 上下载 code 也行，在附件里有），然后 cd 到刚下载

的 AirSim 目录，运行 `build.cmd`（这里如果显示连接失败需要翻，如果是放在 C 盘需要使用管理员模式运行）。



2. 重启 Epic Games Launcher，会弹出修复关联项目，点是就行了。
3. 在 AirSim 目录下 `Unreal\Environments\Block`，运行 `update_from_git.bat`，选着对应的 unreal engine，等编译完成，会生成 `Blocks.sln` 文件，用 VS2022 打开，选择 `DebugGame Editor` 和 `x64`，运行本地 `windows` 调试器就可以打开 Unreal Editor，再点击运行就成功运行 AirSim 了。





AirSim 免安装版场景

可以选择直接下载微软打包好的免安装版场景，双击 exe 就送，但是场景不能编辑：https://github.com/Microsoft/AirSim/releases （注意下载的是 Linux 还是 windows 版本）。

前置条件：安装 vs2022， EPIC GAMES 客户端（不用装 Unreal Engine，主要是取 epic 安装时自动安装的 Directx Runtime 组件，像安装 steam 也会自动安装，更甚之可以直接下载 Directx Runtime）。

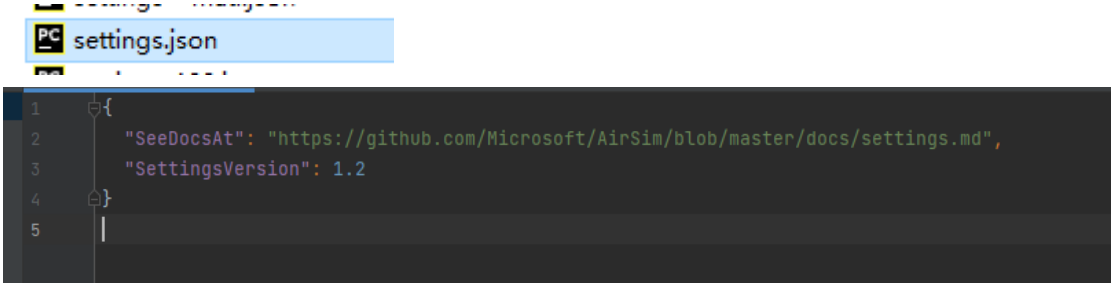
三. 配置

settings.json

注意事项：

1. 文件位于文档下的 AirSim 目录中，C:\Users\你的用户名\Documents\AirSim（如果没有，运行一次打包好的场景即可自动生成，或手动创建目录也行）
2. 编码问题，不能使用记事本编辑，可以使用各种 IDE。
3. 参考文档：
   https://github.com/Microsoft/AirSim/blob/master/docs/settings.md
   https://blog.csdn.net/Zhaoxi_Li/article/details/107946885?spm=1001.2014.3001.5506

初始状态：



载具选择指令：

选择载具为无人机，则指令为`"SimMode": "Multirotor"`

```
"SettingsVersion": 1.2,
"SimMode": "Multirotor",
```

选择载具为车，则指令为`"SimMode": "Car"`

```
"SettingsVersion": 1.2,
"SimMode": "Car",
```

如果选择在打开 AirSim 时在图形界面选择载具，则使用指令`"SimMode": ""`

```
"SettingsVersion": 1.2,
"SimMode": "",
```

如果选择不要载具，纯摄像头模式，则使用指令`"SimMode":"ComputerVision"`

```
"SettingsVersion": 1.2,
"SimMode": "ComputerVision",
```

载具初始化

载具 ID 项可自由编号，无限制。
载具类型常用参数有：

1. PhysXCar ： 无人车
2. SimpleFlight ： 无人机
3. PX4Multirotor ： PX4 模拟器控制
4. ComputerVision ： 纯摄像头模式

若需要使用多智能体载具，则只需增加载具 ID 项内容。

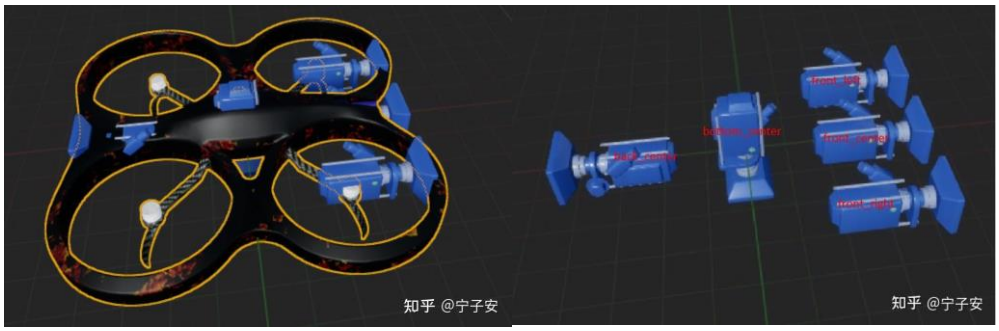
```
"Vehicles": {
  "UAV1": {
    "VehicleType": "SimpleFlight",
    "X": 0,
    "Y": 0,
    "Z": 0,
    "Yaw": 0
  },
  "UAV2": {
    "VehicleType": "SimpleFlight",
    "X": 2,
    "Y": 0,
    "Z": 0,
    "Yaw": 0
  },
  "UAV3": {
    "VehicleType": "SimpleFlight",
    "X": 4,
    "Y": 0,
    "Z": 0,
    "Yaw": 0
  }
},
```

传感器可视化

```
        }
    },
    "SubWindows": [
        {"WindowID": 0, "CameraName": "0", "ImageType": 0, "VehicleName": "Drone0", "Visible": true},
        {"WindowID": 1, "CameraName": "1", "ImageType": 0, "VehicleName": "Drone0", "Visible": true},
        {"WindowID": 2, "CameraName": "2", "ImageType": 0, "VehicleName": "Drone0", "Visible": true}
    ]
```

AirSim 最多支持 3 个可视化窗口，分别对应 `"WindowID": 0, 1, 2`

选择可视化的摄像头：`"CameraName"`，可选参数为 `0, 1, 2, 3, 4`
分别对应 `Multirotor/Car` 的五个视角：前中,前右,前左,底部/驾驶视角，后。
（`012` 视角取景相似）



选择可视化摄像头的类型：`"ImageType"`，可选参数为 `0~9`，分别对应

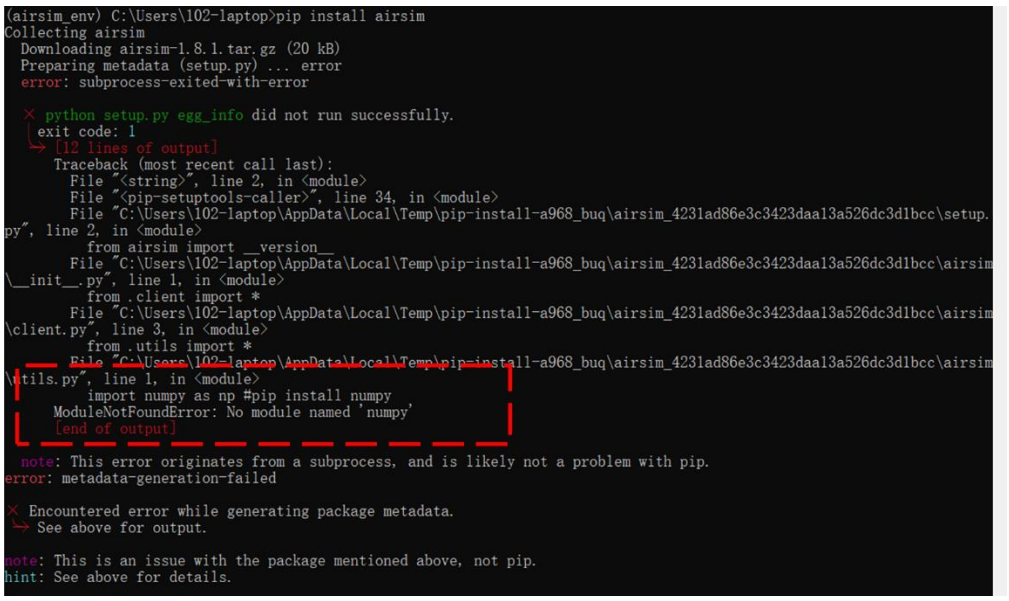
```
Scene = 0, DepthPlanar = 1, DepthPerspective = 2, DepthVis = 3,
DisparityNormalized = 4, Segmentation = 5, SurfaceNormals = 6,
Infrared = 7, OpticalFlow = 8, OpticalFlowVis = 9
```

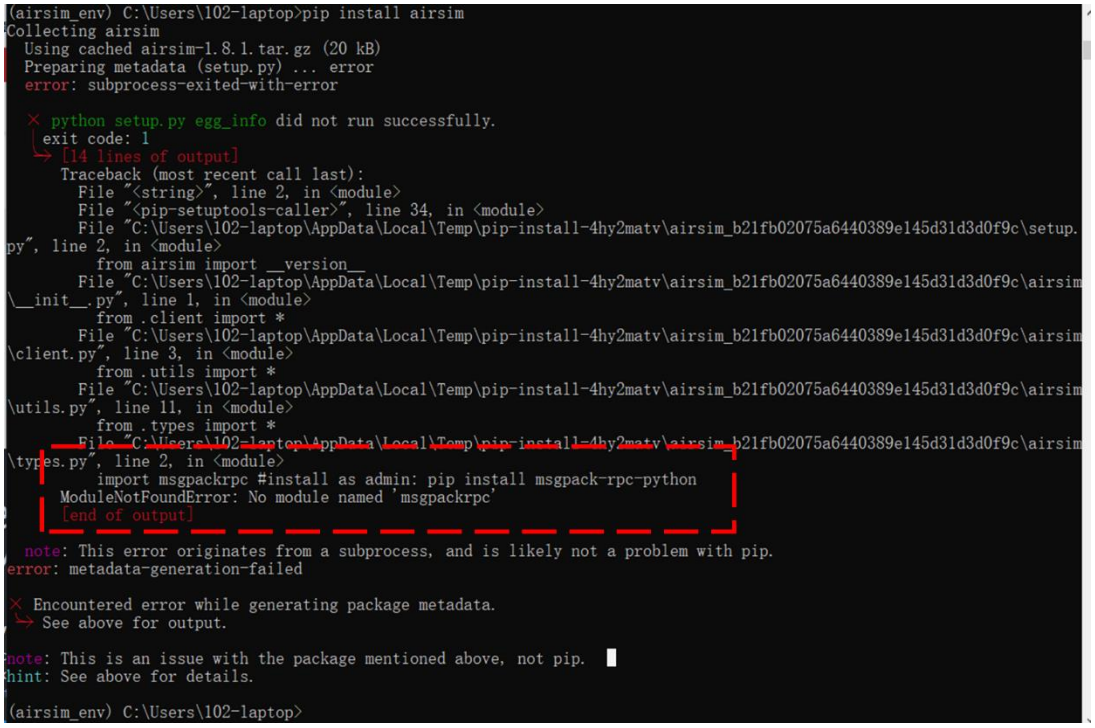
场景=0、深度平面=1、深度透视 = 2、深度视觉 = 3、差异归一化 = 4、
分割 = 5、表面规范 = 6、红外线 = 7、光流 = 8、光流视觉 = 9

以上是较为常用的 `settings` 参数设置，更详细的可查看 `AirSim` 文档。

Python 环境

python 3.8（建议使用 conda 创建新环境）
pip install airsim
（由于是新环境，提示缺啥补啥）





四. 例子

test1：单无人机飞行

settings.json

```
{
  "SeeDocsAt": "https://github.com/Microsoft/AirSim/blob/master/docs/settings.md",
  "SettingsVersion": 1.2,
  "SimMode": "Multirotor",
  "Vehicles": {
      "UAV1": {
            "VehicleType": "SimpleFlight",
            "X": 0,
            "Y": 0,
            "Z": -1
      }
  }
}
```

test1.py

```
import airsim
import time
import numpy as np


client = airsim.MultirotorClient()  # connect to the AirSim simulator
name = "UAV1"
client.enableApiControl(True, name)                    # 开启 Api 控制 UAVi
client.armDisarm(True, name)                           # 解锁螺旋桨
client.takeoffAsync(vehicle_name=name)                 # 起飞

client.moveToZAsync(-5, 1, vehicle_name=name).join()    # 飞到 5m 高度层 (注意 z 轴+为下-为上)
    # moveToZAsync(z, velocity, vehicle_name)
    # 是高度控制 API，第一个参数是高度，第二个参数是速度，第三个是控制的目标

client.moveByVelocityZAsync(1, 1, -5, 10, vehicle_name=name).join()     # 水平速度控制（指定高度）
    # client.moveByVelocityZAsync(vx, vy, z,vehicle_name = '')
```

```
    # (x 轴速度,Y 轴速度，Z 轴，时长，控制对象)

client.moveByVelocityAsync(2, -2, -1, 10, vehicle_name=name).join()   # 三轴飞行
    # moveByVelocityAsync(vx, vy, vz, duration, vehicle_name='')
client.moveByVelocityAsync(2, 0, 0, 15, vehicle_name=name).join()   # vy vz = 0 时，相当于 x 轴方向前进
client.moveByVelocityAsync(0, 2, 0, 6, vehicle_name=name).join()   # y 轴方向前进
client.moveByVelocityAsync(0, 0, 1, 13, vehicle_name=name).join()   # z 轴方向前进


# 结束
client.simPause(False)  # Pauses simulation
client.landAsync(vehicle_name=name).join()              # 降落
client.armDisarm(False, vehicle_name=name)              # 锁定，关闭螺旋桨
client.enableApiControl(False, vehicle_name=name)       # 释放控制权
```

test2：无人机编队飞行

settings.json

```
{
  "SeeDocsAt": "https://github.com/Microsoft/AirSim/blob/master/docs/settings.md",
  "SettingsVersion": 1.2,
  "SimMode": "Multirotor",
  "Vehicles": {
    "UAV1": {
      "VehicleType": "SimpleFlight",
      "X": 0, "Y": 0, "Z": 0,
      "Yaw": 0
    },
    "UAV2": {
      "VehicleType": "SimpleFlight",
      "X": 2, "Y": 0, "Z": 0,
      "Yaw": 0
    },
    "UAV3": {
      "VehicleType": "SimpleFlight",
      "X": 4, "Y": 0, "Z": 0,
```

```
      "Yaw": 0
    },
    "UAV4": {
      "VehicleType": "SimpleFlight",
      "X": 0, "Y": -3, "Z": 0,
      "Yaw": 0
    },
    "UAV5": {
      "VehicleType": "SimpleFlight",
      "X": 2, "Y": -2, "Z": 0,
      "Yaw": 0
    },
    "UAV6": {
      "VehicleType": "SimpleFlight",
      "X": 4, "Y": -3, "Z": 0,
      "Yaw": 0
    },
    "UAV7": {
      "VehicleType": "SimpleFlight",
      "X": 0, "Y": 3, "Z": 0,
      "Yaw": 0
    },
    "UAV8": {
      "VehicleType": "SimpleFlight",
      "X": 2, "Y": 2, "Z": 0,
      "Yaw": 0
    },
    "UAV9": {
      "VehicleType": "SimpleFlight",
      "X": 4, "Y": 3, "Z": 0,
      "Yaw": 0
    }
  }
}
```

test2.py

```
import airsim
import time
import numpy as np

uav_nums = 9
origin_x = [0, 2, 4, 0, 2, 4, 0, 2, 4]       # 无人机初始位置
origin_y = [0, 0, 0, -3, -2, -3, 3, 2, 3]
```

```
8.
9.  def get_UAV_pos(client, vehicle_name="SimpleFlight"):   # 获取 UAV 位置
10.     global origin_x
11.     global origin_y
12.     state = client.simGetGroundTruthKinematics(vehicle_name=vehicle_name)
13.     # Get Ground truth kinematics of the vehicle 获取车辆的地面实况运动学
14.     x = state.position.x_val
15.     y = state.position.y_val
16.     i = int(vehicle_name[3])    # UAVi 无人机列队
17.     x += origin_x[i - 1]        # 移动位置写死
18.     y += origin_y[i - 1]
19.     pos = np.array([[x], [y]])      # 返回 array 类型的 pos
20.     return pos
21.
22.
23. client = airsim.MultirotorClient()  # connect to the AirSim simulator
24. for i in range(uav_nums):
25.     name = "UAV"+str(i+1)
26.     client.enableApiControl(True, name)     # 开启 Api 控制 UAVi
27.     client.armDisarm(True, name)            # 解锁螺旋桨
28.     if i != uav_nums-1:                                 # 起飞
29.         client.takeoffAsync(vehicle_name=name)
30.     else:
31.         client.takeoffAsync(vehicle_name=name).join()
32.         # Async methods returns Future.
33.         # Call join() to wait for task to complete.
34.
35. for i in range(uav_nums):                               # 全部都飞到同一高度层 3m
36.     name = "UAV" + str(i+1)
37.     if i != uav_nums-1:
38.         client.moveToZAsync(-3, 1, vehicle_name=name)
39.         # moveToZAsync(z, velocity, vehicle_name)
40.         # 是高度控制 API，第一个参数是高度，第二个参数是速度，第三个是控制的目标
41.     else:
42.         client.moveToZAsync(-3, 1, vehicle_name=name).join()    # .join()
43.
44.
45. # 参数设置
46. v_max = 2     # 无人机最大飞行速度
47. r_max = 20    # 邻居选择半径
48. k_sep = 7     # 控制算法系数
49. k_coh = 1
50. k_mig = 1
51. pos_mig = np.array([[25], [0]])   # 目标位置
```

```
v_cmd = np.zeros([2, 9])

for t in range(500):
    for i in range(uav_nums):   # 计算每个无人机的速度指令
        name_i = "UAV"+str(i+1)
        pos_i = get_UAV_pos(client, vehicle_name=name_i)
        r_mig = pos_mig - pos_i
        v_mig = k_mig * r_mig / np.linalg.norm(r_mig)
        v_sep = np.zeros([2, 1])
        v_coh = np.zeros([2, 1])
        N_i = 0
        for j in range(uav_nums):
            if j != i:
                N_i += 1
                name_j = "UAV"+str(j+1)
                pos_j = get_UAV_pos(client, vehicle_name=name_j)
                if np.linalg.norm(pos_j - pos_i) < r_max:
                    r_ij = pos_j - pos_i
                    v_sep += -k_sep * r_ij / np.linalg.norm(r_ij)
                    v_coh += k_coh * r_ij
        v_sep = v_sep / N_i
        v_coh = v_coh / N_i
        v_cmd[:, i:i+1] = v_sep + v_coh + v_mig

    for i in range(uav_nums):   # 每个无人机的速度指令执行
        name_i = "UAV"+str(i+1)
        client.moveByVelocityZAsync(v_cmd[0, i], v_cmd[1, i], -
3, 0.1, vehicle_name=name_i)
        #  client.moveByVelocityZAsync(vx, vy, z, duration,
        #  drivetrain = DrivetrainType.MaxDegreeOfFreedom, yaw_mode = YawMod
e(), vehicle_name = '')
        # (x 轴速度,Y 轴速度，Z 轴，时长，控制对象) ，水平速度控制（指定高度）


# 循环结束
client.simPause(False)  # Pauses simulation
for i in range(uav_nums):
    name = "UAV"+str(i+1)
    if i != uav_nums-1:                                    # 降落
        client.landAsync(vehicle_name=name)
    else:
        client.landAsync(vehicle_name=name).join()
for i in range(uav_nums):
    name = "UAV" + str(i+1)
```

```
    client.armDisarm(False, vehicle_name=name)              # 锁定，关闭螺旋桨
    client.enableApiControl(False, vehicle_name=name)       # 释放控制权
```

test3：传感器图片数据收集

settings.json

```
{
  "SeeDocsAt": "https://github.com/Microsoft/AirSim/blob/master/docs/settings.md",
  "SettingsVersion": 1.2,
  "SimMode": "Multirotor",
  "Vehicles": {
      "UAV1": {
            "VehicleType": "SimpleFlight",
            "X": 0,
            "Y": 0,
            "Z": -3
      }
  },
  "SubWindows": [
      {"WindowID": 0, "CameraName": "0", "ImageType": 0, "VehicleName": "UAV1", "Visible": true},
      {"WindowID": 1, "CameraName": "0", "ImageType": 5, "VehicleName": "UAV1", "Visible": true},
      {"WindowID": 2, "CameraName": "0", "ImageType": 7, "VehicleName": "UAV1", "Visible": true}
  ]
}
```

test3.py

```
import airsim
import time
import numpy as np
import datetime


client = airsim.MultirotorClient()  # connect to the AirSim simulator
name = "UAV1"
client.enableApiControl(True, name)                     # 开启 Api 控制 UAVi
client.armDisarm(True, name)                            # 解锁螺旋桨
client.takeoffAsync(vehicle_name=name)                  # 起飞
```

```
12.
13. client.moveToZAsync(-5, 1, vehicle_name=name).join()    # 飞到 5m 高度层 (注意 z 轴+为下-为上)
14.     # moveToZAsync(z, velocity, vehicle_name)
15.     # 是高度控制 API，第一个参数是高度，第二个参数是速度，第三个是控制的目标
16.
17. client.moveByVelocityZAsync(1, 1, -5, 10, vehicle_name=name).join()     # 水平速度控制（指定高度）
18.     # client.moveByVelocityZAsync(vx, vy, z,vehicle_name = '')
19.     # (x 轴速度,Y 轴速度，Z 轴，时长，控制对象)
20.
21. client.moveByVelocityAsync(2, -2, -1, 10, vehicle_name=name).join()   # 三轴飞行
22.     # moveByVelocityAsync(vx, vy, vz, duration, vehicle_name='')
23. client.moveByVelocityAsync(2, 0, 0, 15, vehicle_name=name).join()   # vy vz = 0 时，相当于 x 轴方向前进
24. client.moveByVelocityAsync(0, 2, 0, 6, vehicle_name=name).join()   # y 轴方向前进
25. client.moveByVelocityAsync(0, 0, 1, 5, vehicle_name=name).join()   # z 轴方向前进
26.
27. client.hoverAsync(vehicle_name=name).join()             # 悬停
28.
29. responses = client.simGetImages([
30.     airsim.ImageRequest(0, airsim.ImageType.Scene, pixels_as_float=False, compress=True),           # 0
31.     airsim.ImageRequest(0, airsim.ImageType.Segmentation, pixels_as_float=False, compress=True),     # 5
32.     airsim.ImageRequest(0, airsim.ImageType.Infrared, pixels_as_float=False, compress=True)     # 5
33. ], vehicle_name = name)
34.
35. now = datetime.datetime.now()  # 时间
36. now_string = now.strftime('%Y_%m_%d_%H_%M_')
37.
38. for i in range(3):
39.     image_name = now_string + "scene" + str(i) + ".png"
40.     f = open(image_name, 'wb')
41.     f.write(responses[i].image_data_uint8)
42.     f.close()
43.
44. # 结束
45. client.simPause(False)  # Pauses simulation
46. client.landAsync(vehicle_name=name).join()              # 降落
```

```
client.armDisarm(False, vehicle_name=name)              # 锁定，关闭螺旋桨
client.enableApiControl(False, vehicle_name=name)       # 释放控制权
```

图像类型	解释	调用方式
Scene	彩色图	airsim.ImageType.Scene
DepthPlanar	深度图，像素值代表到相平面的距离	airsim.ImageType.DepthPlanar
DepthPerspective	深度图，像素值代表透视投影下到相平面的距离	airsim.ImageType.DepthPerspective
DepthVis	深度图，为了更好的展示距离的远近，100 米对应白色，0 米对应黑色	airsim.ImageType.DepthVis
DisparityNormalized	深度视差图，每个像素值是浮点型	airsim.ImageType.DisparityNormalized
Segmentation	分割图，不同 ID 号的物体用不同的颜色表示	airsim.ImageType.Segmentation
SurfaceNormals	表面法线，包含了比较丰富的细节信息	airsim.ImageType.SurfaceNormals
Infrared	红外图，与分割图类型，只是颜色变为 ID 号	airsim.ImageType.Infrared

图像相关详细参考：https://zhuanlan.zhihu.com/p/507304210

五. 附录

1. AirSim：https://github.com/microsoft/AirSim
2. Project AirSim（商业版本）：https://aka.ms/projectairsim
3. AirSim 文档： https://microsoft.github.io/AirSim/
4. AirSim API 文档直达： https://microsoft.github.io/AirSim/api_docs/html/index.html
5. Visual Studio：https://visualstudio.microsoft.com/zh-hans/vs/
6. Visual Studio 2019 ：https://docs.microsoft.com/zh-cn/visualstudio/releases/2019/release-notes
7. Epic-unreal Engine：https://www.epicgames.com/site/zh-CN/home
8. settings.json 文档解析：
   https://github.com/Microsoft/AirSim/blob/master/docs/settings.md
   https://blog.csdn.net/Zhaoxi_Li/article/details/107946885?spm=1001.2014.3001.5506