拒止环境下对抗性目标搜寻

拒止环境

加了一个关于通信的设定上的变更。简单来说遵循下面两个规则：

1. 无人机之间只有在发现目标时才会尝试通信，通信次数存在上界，此处为80（注：每次向一架无人机传输一次信息即算作一次通信）
2. 无人机之间的通信可能会受到干扰，即有概率想通信但无法通信，由于第一点变更，通信失败的概率现在降低为1/10.

对抗

对抗体现为两点：

1. 争夺目标，一方占据目标时另一方无法从目标上获取正得分。
2. 攻击，双方无人机可以使用碰撞的手段进行攻击，假设双方无人机势均力敌，因此碰撞时双方失败的概率相当，具体而言，有2/5的概率红方失败，2/5的概率蓝方失败，剩下1/5的概率双方同归于尽。

目标搜寻

双方的目标都是搜索处于狭长的“山谷”地带的目标，只能确定目标会在山谷地带移动，但移动规律和位置都是未知的。一旦无人机与目标处于同一坐标，该无人机就可以获取到正的得分。

uav_project

该项目以python3构建，需要的第三方库如下：

- pyglet
- numpy
- matplotlib
- tensorflow
- sklearn

上述第三方库在window上可通过`pip install xx`安装, linux上推荐使用virtualenv构建虚拟环境。

minimal example

实现策略时无需关心环境内部是如何运作的，可以假定你的策略为一个函数policy(state, positive)，该函数根据state信息决定采取何种动作。一个最简单的policy实现如下：

```
'''
state数据结构如下
        {
            'self_die': False|True,
            'self_pos': (a, b),
            'positive_pos': set(),
```

```
            'negative_pos': set(),
            'object_pos': set(),
            'obstacle_pos': list()
        }
'''

def policy(state, positive):
    # positive指定当前是为红方还是蓝方指定策略，True|False，True表示红方，False表示蓝方
    # 补充代码，使用人类的智慧指定你认为的最佳的动作~
    return random.randint(0, 4) # 返回0或者1或者2或者3
```

该策略将会像下面这样调用：

```
from env import Env

env = Env(debug=True) # debug = True 开启可视化环境

state = env.reset()
done = False

while not done:
    action = []
    for i in range(16):
        action.append(policy(state[i]), i < 8) # 0-7为红方，8-15为蓝方
    state, reward, done, info = env.step(action) # 第i个元素表示第i架无人机的动作
    env.render() # 注意在debug = False的情况下该语句没有任何作用
```