# 无人机AI线路规划大赛接口文档



## 1. 概述

本次大赛设置有裁判服务器Judger，所有比赛选手Player需要使用Socket TCP和Judger进行通讯，所有通讯数据均为**JSON**格式。

调试服务器IP地址及TCP端口：

```
Judger's IP:    39.106.111.130
Judger's port:  4010
```

## 2. Socket通讯协议

### 2.1 选手连接裁判服务器Judger

连接成功后，Judger会返回一条消息：

```
{
    "notice": "token",
    "msg": "hello, what's your token?"
}
```

### 2.2 选手向裁判服务器表明身份(Player -> Judger)

```
{
    "token": "eyJ0eXAiOiJKV1",
    "action":"sendtoken"
}
```

其中：
token：选择对战选手后，会生成一个身份验证令牌，可以在“我的对抗”页面找到；
action：客户端连接服务器的事件类型，sendtoken为进行身份验证。

### 2.3 身份验证结果(Judger -> Player)

```
{
    "token": "eyJ0eXAiOiJKV1",
    "notice": "tokenresult",
    "result": 0,
    "roundId": "vvvvv",
    "yourId": "player01"
}
```

其中：
token: 身份验证令牌
notice: tokenresult为进行身份验证结果事件
result: 0 正常， -1 非法
roundId：对战房间**id**
yourId: 我的唯一**id**

#### 2.3.1 选手向裁判服务器表明自己已准备就绪(Player -> Judger)

```
{
    "token": "eyJ0eXAiOiJKV1",
    "action":"ready"
}
```

#### 2.4 对战开始通知(Judger -> Player)，您需要1s之内返回，否则直接判负，调试阶段无此限制。

类型：json字符串，服务器通知选手端

```
js 示例

{
    "token": "eyJ0eXAiOiJKV1",
    "notice": "sendmap",
    "time": 0,
    "map": {***}
}
```

其中：
token: 身份验证令牌
notice: 发送地图信息后,表示比赛开始，该时刻为0，参赛者程序收到地图信息，不能移动无人机、不能进行充电。此时，需要参赛者将他们所控制的无人机时刻为0的位置发送给服务器
**map**：地图数据，参见地图数据格式章节。

#### 2.5 选手返回下一步无人机坐标(Player -> Judger)

收到“比赛下一步骤指令”指令后，需要1s之内返回，否则直接判负，调试阶段没有此限制。取货放货规则：1、必须到了取货点(start_x, start_y, 0)，才能设置goods_no进行取货； 2、到了放货点(end_x, end_y, 0)，服务器会自动放货，千万不要自己将goods_no改为-1，否则直接违规，服务器会在这一步的结果中自动将goods_no设置为-1。

```
{
    "token": "eyJ0eXAiOiJKV1",
    "action": "flyPlane",
    //无人机信息：必须包含我方控制的所有的无人机的信息（除了撞毁的），信息内容包括无人机编号，xyz坐标， goods_no 货物编号，-1表示没有载货。当无人机飞到停机坪后，就默认开始充电，不需要选手设置，由于载货时不能充电，因此载货的时候，不能进入停机坪。
    "UAV_info": [
        { "no": 0, "x": 10, "y": 20, "z": 80, "remain_electricity": 1000,"goods_no": 0},
        { "no": 1, "x": 10, "y": 20, "z": 90, "remain_electricity": 1000, "goods_no": -1},
        { "no": 2, "x": 10, "y": 30, "z": 40, "remain_electricity": 1000,"goods_no": 2 },
        { "no": 4, "x": 70, "y": 20, "z": 20, "remain_electricity": 1000,"goods_no": 3 }
    ],

    //请求购买无人机：该字段可以没有，如果有表示要购买无人机，提交购买请求后，下一次收到的服务器发送过来的我方无人机信息中会增加相应的无人机信息，初始位置为停机坪,若购买2架F1，一架F2，写法如下：。
    "purchase_UAV": [
        { "purchase": "F1" },
        { "purchase": "F1" },
        { "purchase": "F2" }
    ]
}
```

### 2.6 比赛下一步骤指令(Judger -> Player)

收到“比赛下一步骤指令”指令后，需要1s之内返回，否则直接判负，调试阶段无此限制,与2.5形成循环直到比赛结束。

```
{
    "token": "eyJ0eXAiOiJKV1",
    "notice": "step",
    //比赛状态：0表示正常比赛中，1表示比赛结束，收到为1时，参赛者可以关闭连接，
    "match_status": 0
    //当前时间：当前的时间，每次给比赛者都会比上一次增加1
    "time": 16,

    //我方无人机信息": "不同时间，数据不同。 我方无人机的当前信息，根据我方传递给服务器后，服务器经过计算后得到的数据， goods_no货物编号， -1表示没有载货物，否则表示装载了相应的货物,剩余电量remain_electricity，表示当前无人机还剩余的电量
    //状态说明："无人机状态 0表示正常， 1表示坠毁， 2表示处于雾区，3表示正在充电 其他数据暂时未定义"
    "UAV_we": [
        { "no": 0,  "type": "F1","x": 10, "y": 20, "z": 80, "goods_no": -1,"load_weight": 100, "remain_electricity": 1000,"status": 0 },
        { "no": 1,  "type": "F1","x": 10, "y": 20, "z": 90, "goods_no": 0,"load_weight": 100, "remain_electricity": 1000,"status": 0 },
        { "no": 2,  "type": "F1","x": 10, "y": 30, "z": 40, "goods_no": 3,"load_weight": 100,"remain_electricity": 1000,"status": 0 },
        { "no": 3,  "type": "F1","x": 50, "y": 20, "z": 30, "goods_no": 5,"load_weight": 100,"remain_electricity": 1000,"status": 0 },
        { "no": 4,  "type": "F1","x": 70, "y": 20, "z": 20, "goods_no": -1,"load_weight": 100,"remain_electricity": 1000, "status": 1 }
    ],

    //"我方目前总价值": "不同时间，数据不同，表示当前时刻，我方所取到的运送物品价值",
```

```
    "we_value": 10000,

    //"敌方无人机信息": "不同时间，数据不同。 敌方无人机的当前信息，根据敌方传递给服务器后，服务器经过计算后得到的数据，如果敌方无人机在雾区，状态为2， x， y，z坐标都为-1，表示无效。"
    //"状态说明": "无人机状态 0表示正常，2表示处于雾区，3表示正在充电,坠毁的不返回。"
    "UAV_enemy": [
        { "no": 0, "type": "F1","x": 40, "y": 20, "z": 80, "goods_no": -1,"load_weight": 100,"remain_electricity": 1000,"status": 0 },
        { "no": 1,"type": "F1", "x": 20, "y": 20, "z": 90, "goods_no": 7,"load_weight": 100,"remain_electricity": 1000,"status": 0 },
        { "no": 2, "type": "F1","x": 80, "y": 30, "z": 40, "goods_no": -1,"load_weight": 100,"remain_electricity": 1000,"status": 0 },
        { "no": 3, "type": "F1","x": 90, "y": 20, "z": 30, "goods_no": -1,"load_weight": 100,"remain_electricity": 1000,"status": 0 },
        { "no": 5, "type": "F1","x": -1, "y": -1, "z": -1, "goods_no": -1,"load_weight": 100,"remain_electricity": 1000,"status": 2 }
    ],
    //"敌方目前总价值": "不同时间，数据不同，表示当前时刻，敌方获取到的运送物品价值",
    "enemy_value": 30000,

    //物品信息: 不同时间，数据不同，no 货物唯一编号， startxy 表示货物出现的地面坐标，endxy表示货物需要运送到的地面坐标， weight表示货物的重量，value表示运送到后货物的价值,start_time:货物出现的时间,remain_time:货物从开始出现到消失的持续时长,left_time: 货物可被捡起的剩余时长，这是个冗余字段，您可以从start_time+remain_time-step得出；一旦被捡起，remain_time和left_time字段无效。status为0表示货物正常且可以被拾起,status为1表示已经被无人机拾起，status为2表示已经运送到目的地，status为3表示无效（无效包括运送过程中撞毁、货物超时未被拾起等，被删除),其实您只能看见0和1状态，因为其他状态的货物会被删除,status为0时，left_time才有意义,已经消失或送到的货物会在列表中被删除。

    "goods": [
        { "no": 0, "start_x": 3, "start_y": 3, "end_x": 98, "end_y": 3, "weight": 55, "value": 100, "start_time":15,"remain_time": 90, "left_time": 89,"status": 1},
        { "no": 1, "start_x": 98, "start_y": 13, "end_x": 3, "end_y": 3, "weight": 51, "value": 90,"start_time":15, "remain_time": 9, "left_time": 8,"status": 0},
        { "no": 2, "start_x": 15, "start_y": 63, "end_x": 81, "end_y": 33, "weight": 15, "value": 20,"start_time":15, "remain_time": 7, "left_time": 6,"status": 0},
        { "no": 3, "start_x": 3, "start_y": 3, "end_x": 98, "end_y": 3, "weight": 55, "value": 100, "start_time":15,"remain_time": 330, "left_time": 329,"status": 0},
        { "no": 5, "start_x": 3, "start_y": 3, "end_x": 98, "end_y": 3, "weight": 55, "value": 100,"start_time":15, "remain_time": 3, "left_time": 2,"status": 0}
    ]
}
```

## 3. 数据格式描述

### 3.1 地图数据格式

```
{
    "map": {          //地图信息
        "x": 100,
        "y": 100,
        "z": 100     //天空最大高度
    },
```

```
    "parking": {    // 停机坪
        "x": 0,
        "y": 0
    },
    "h_low": 60,    //"飞行最低高度": 固定值
    "h_high": 99,  //"飞行最高高度": 固定值
    "building": [   //xy表示建筑物的起始位置
                    // l表示长度，w表示宽度，h表示高度
                    //因此水平上坐标位置为x->x+l-1, y->y+w-1",
        { "x": 10, "y": 10, "l": 10, "w": 10, "h": 80 },
        { "x": 40, "y": 40, "l": 10, "w": 10, "h": 60 }

    ],
     //雾区: 固定值，整个比赛过程中不变，雾区个数根据地图而不同，
     //xy表示雾区的起始位置，l表示长度，w表示宽度，b表示雾区最低高度，
     //t表示雾区的最大高度，水平上坐标为x->x+l-1, y->y+w-1，垂直区间为b->t",
    "fog": [
        { "x": 60, "y": 60, "l": 10, "w": 10, "b": 55, "t": 90 },
        { "x": 35, "y": 47, "l": 15, "w": 20, "b": 60, "t": 99 }
    ],
     //"一开始停机坪无人机信息": "固定值，整个比赛过程中不变，无人机个数根据地图而不同，无人机信息包括 编号和最大载重量，编号单方唯一,剩余电量 remain_electricity, 表示当前无人机还剩余的电量,初始值为0"
    "init_UAV": [
        { "no": 0, "x":0,"y":0,"z":0,"load_weight": 100,"type": "F1", "status": 0,
"remain_electricity": 0,"goods_no":-1},
        { "no": 1, "x":0,"y":0,"z":0,"load_weight": 20 ,"type": "F3", "status": 0,
"remain_electricity": 0,"goods_no":-1},
        { "no": 2, "x":0,"y":0,"z":0,"load_weight": 20 ,"type": "F3", "status": 0,
"remain_electricity": 0,"goods_no":-1}
    ],

    //"无人机价格表": "固定值，整个比赛过程中不变，no表示无人机购买编号，价格表根据载重不同，价值也不同，初始化的无人机中的载重必定在这个价格表中，方便统计最后价值,电池容量 capacity，单位时间充电量 charge"
    "UAV_price": [
        { "type": "F1", "load_weight": 100, "value": 300,"capacity": 9000,"charge": 1000},
        { "type": "F2","load_weight": 50, "value": 200,"capacity": 8000 ,"charge": 800},
        { "type": "F3","load_weight": 20, "value": 100,"capacity": 5000 ,"charge": 400},
        { "type": "F4","load_weight": 30, "value": 150,"capacity": 6000 ,"charge": 500},
        { "type": "F5","load_weight": 360, "value": 400,"capacity": 20000,"charge": 3000}
    ],

}
```