“巧取智夺”赛道 Python SDK

软院、计算机系联合开发组

2021 年 4 月 1 日
版本：0300fb0

1 简介

本 SDK 可以帮助你的 AI 和评测后端通信。这个 SDK 由以下文件组成：

```
python
├── main.py ..... 选手 AI 代码文件
└── aisdk ..... SDK 包
    ├── __init__.py ..... aisdk 包定义
    ├── entities.py ..... 描述玩家和金蛋状态的类型
    ├── gamestate.py ..... 和评测逻辑交互的 API
    ├── player_movement.py ..... 描述玩家移动的类型
    └── utils.py ..... SDK 内部使用的和评测逻辑交互的辅助类
```

2 环境配置

为运行 SDK，你需要配置以下环境：

- Python 3.6 及以上版本

对于开发，我们推荐使用 Visual Studio Code 和其 Python 插件的组合。你也可以使用 JetBrains PyCharm 或者其他你喜欢的集成开发环境进行开发。在下载 SDK 后，你可以尝试运行 `main.py`，以检查本地运行环境。如果运行出错，请检查你的本地 Python 环境是否正确配置。

3 开发

理论上，你只需要修改 `main.py` 这一文件中的 `update()` 函数。这个函数会在每秒 10 次[1]的更新中被调用，在其中你可以尝试做出各种动作。请注意：这些函数都不会返回运行的结果，且并不会在调用后立刻体现效果。所有的操作请求都会在 `update()` 运行结束后一并发送给游戏逻辑。因此，你需要在下一次 `update()` 运行时对是否成功执行动作进行检查。

SDK 中提供的主要数据结构见表1。

3.1 接口

所有公共接口均位于 `gamestate.py` 中。根据 Python 的模块导入，模块本身即为单例模式。代码中要使用相应接口，只需要导入 `aisdk.gamestate` 这一模块即可。

[1]游戏运行于 60fps，每 6 帧运行一次更新函数，即为每秒运行 10 次。

表 1: SDK 提供的数据结构介绍

Player	
`position`	玩家坐标
`facing`	表示玩家朝向的单位向量
`status`	玩家运动状态
`holding`	玩家拿的蛋，`None` 表示空手

Team	
`RED`	红队
`YELLOW`	黄队
`BLUE`	蓝队

Egg	
`position`	蛋坐标
`holder`	拿蛋玩家，`None` 表示放在地上
`score`	蛋的分数

PlayerMovement	
`STOPPED`	玩家停在原地
`WALKING`	玩家正在走路
`RUNNING`	玩家正在跑步
`SLIPPED`	玩家因碰撞滑倒，本回合操作无效

玩家控制 通过对玩家对象属性的读取和赋值，以尝试获取和修改玩家的具体信息。下文中 `p` 代表玩家对象。

注意：在对玩家代理对象赋值操作后，立刻读取得到的仍然是原来的值！这是因为修改状态的操作尚未被评测端接受，所有修改操作会在更新回调结束后一并发送给评测端。你应当在下次调用更新回调函数时加以检查。

- `Player(player_id: int)`
  `Player.get_player_by_team_and_id(team_id: Team, player_id_in_team: int)`
  获得 Player 对象。若设总的编号为 x，则队伍 t 和队内编号 y 由以下公式得出：

  $$t = x \div 4, y = x \bmod 4$$

  其中 $t = 0, 1, 2$ 分别对应红、黄、蓝队。

- `p.player_id`
  只读。玩家的 id，范围为 $0 \sim 11$.

- `p.position`
  只读。玩家的坐标。

- `p.team, p.id_on_team`
  只读。获得这个玩家所在队伍和队内编号，即上文的 t, y。

- `p.holding`
  只读。玩家拿的蛋对应的 `Egg` 对象，若为拿蛋即为 `None`.

- `p.status`
  通过对玩家对象`p` 的 `status` 属性进行赋值，以尝试设置移动状态。赋的新值必须为 `PlayerMovement` 类型；尝试改变不在当前 AI 队伍的玩家的状态会导致抛出异常。如果不满足条件，则设置失败。具体失败的情形为：
  - 该玩家已经摔倒：此时在站起来（恢复成静止）前不能进行任何操作
  - 抱着蛋时尝试奔跑
  - 体力值不够时尝试奔跑

- `p.facing`
  直接读取值即为玩家当前朝向。通过给玩家对象 `p` 的 `facing` 属性赋值，设置其朝向（用于走路、奔跑）。注意：若赋值的是非单位向量，则会将其变为同向单位向量。传入零向量或者模长过小的向量时，评测逻辑行为未定义。

金蛋基本信息 通过对 `Egg` 对象属性的读取，以尝试获取金蛋的具体信息。下文中 `e` 代表金蛋对象。

- `Egg(egg_id: int)`
  传入金蛋编号，获得其基本信息对象。
- `e.egg_id`
  只读。金蛋 `e` 的 id.
- `e.position`
  只读。金蛋的坐标。
- `e.holder`
  只读。拿着这个金蛋的玩家对象，若为 `None` 表示蛋在地上。
- `e.score`
  只读。金蛋的分数。

金蛋控制 下文中 `p` 代表玩家对象。

- `p.try_grab_egg(egg_id: int)`
  让当前 AI 队伍中某玩家对象 `p` 尝试抓取金蛋。只有满足下列条件时，抓取才能成功：
  - 蛋在地上且糖豆人中心和蛋表面距离不超过 0.1 m（即到蛋中心距离不超过0.69 m）[2]
  - 该蛋由别人拿取，且玩家和蛋距离同样不超过0.69 m
  - 多人在同回合抢同一个蛋时，某人和蛋距离最近
- `p.try_drop_egg(radian: float)`
  让当前 AI 队伍中某玩家对象 `p` 尝试放置金蛋。参数中的弧度为以 $+x$ 轴为极轴的极坐标系下，放置蛋相对玩家的方位。蛋在放置后会和玩家刚好相切。只有满足下列条件时，放置才能成功：
  - 该玩家手中有蛋
  - 蛋放下后不会卡在他人或其他蛋碰撞箱内
  - 蛋放下后不会卡在墙内

3.2 上交代码

按照 Saiblo 的要求，提交 Python 语言代码只需要打包上传 SDK 文件夹下的所有文件即可。注意上传文件中，`main.py` 必须位于压缩包的顶层文件夹。

[2] $\varnothing_{\text{玩家}} = 0.48\,\text{m}, \varnothing_{\text{金蛋}} = 0.7\,\text{m}$