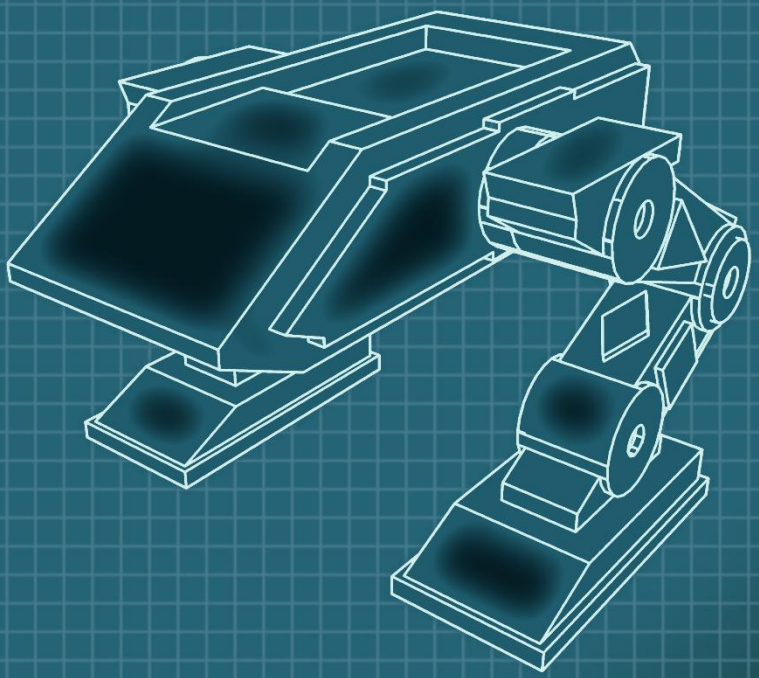
机甲帝国

第二季机甲帝国 C++智能体编程争霸赛

开发者手册（Windows VS 开发包）



By 狄学长

开发者手册

目录

1 闪电入门

解压下载的【Windows 平台 VS 开发包】
其中包含了两个文件夹【战斗调试器】和【RobotAIFactory】

1.1 战斗调试器

【战斗调试器】文件夹中是本地的战斗程序，可以调用玩家开发生成的机甲.dll 进行战斗生成录像。
其中玩家可以编辑 ConsoleConf.txt 配置进行作战的机甲。默认已经配置为 RobotAIFactory 开发目录中的机甲 PK 测试机甲。沿用此配置，玩家可以极其方便地将自己最新开发生成机甲与测试机甲进行战斗和调试。

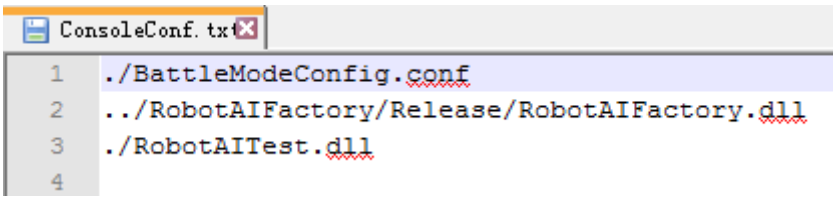


图 1 编辑 ConsleConf.txt 中的两个 dll 路径即可指定参战机甲

双击【机甲帝国战斗调试器.exe】即可进行一场战斗，并生成以时间戳命名的战斗录像文件。使用从网站下载的【Flash 战斗播放器】或【Unity 战斗播放器】即可播放观看战斗。

详细配置请参考后文。

1.2 RobotAIFactory

【RobotAIFactory】是开发属于你自己的机甲智能体的 VS 工程文件。
请确保计算机上安装了 VS2012（或其他相似版本，MS 提供学生免费版；不能使用 VC6）
双击【RobotAIFactory.sln】打开工程。

进行设置：项目》属性》配置属性》C/C++》常规》附加包含目录》编辑
添加你计算机上的 RobotAIFactory/RobotAIFactory/sys 文件夹

之后就可以编译生成机甲 DLL 了。

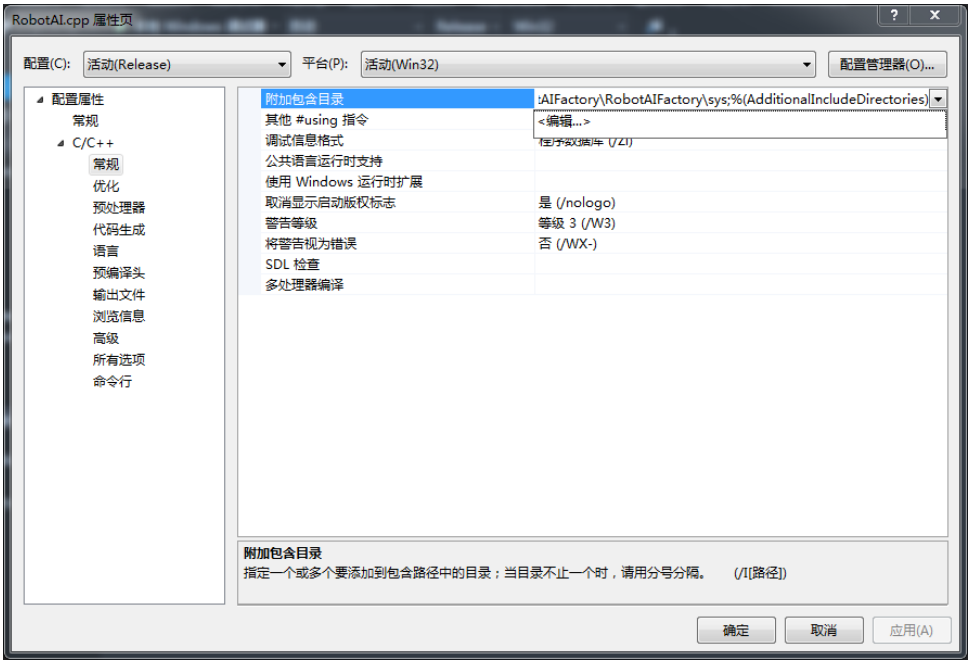


图 2 设置包含目录

1.3 开发

初步的开发只需要关注【RobotAI.cpp】即可。
进阶开发：若要创建任何新的文件，请放置于【myAI】文件夹下。任何时候都无需修改除【myAI】文件夹中的文件。
请不要使用任何 Windows 平台特有的 API 和特性。

Update 函数，机甲的主要行动函数，通过对 **order** 结构体赋值，来在每一帧操纵机甲。

```
void RobotAI::Update(RobotAI_Order& order,const RobotAI_BattlefieldInformation&
info,int myID)
{
    //帧操纵函数
    //功能：在每一帧被调用，完成你的机甲在这一帧的动作决策
    //参数：order... 机甲操纵指令，你在函数体中给它赋值以操纵机甲在这一帧的行为
    //      info... 战场信息
    //      myID... 自己机甲在info中robot数组对应的下标
    //      (这几个参数的详细说明在开发手册可以找到，你也可以在RobotAIstruct.h中直
接找到它们的代码)
    order.fire = 1;
    order.wturn = 1;
}
```

ChooseArmor 函数，机甲的装备选择函数，通过对 weapon 和 engine 赋值来选取装备。

```
void RobotAI::ChooseArmor(weapontypename& weapon,enginetypename& engine,bool a)
{
    //挑选装备函数
    //功能：在战斗开始时为你的机甲选择合适的武器炮塔和引擎载具
    //参数：weapon    ... 代表你选择的武器，在函数体中给它赋值
    //       engine   ... 代表你选择的引擎，在函数体中给它赋值
    //tip:   括号里的参数是枚举类型 weapontypename 或 enginetypename
    //       开发文档中有详细说明，你也可以在RobotAIstruct.h中直接找到它们的代码
    //tip:   最后一个bool是没用的。。那是一个退化的器官

    weapon = WT_Cannon;   //啊，我爱加农炮
    engine = ET_Spider;   //啊，我爱小蜘蛛
}
```

参数详情请见后文。或者浏览【RobotAIstruct.h】

1.4 提交

直接将 myAI 整个文件夹打包成 zip 格式，到大赛官网提交（mechempire.cn）。
请尽量使用 windows 自带的打包方式（右键 myAI 文件夹》发送到》压缩 zip 文件夹），避免使用各种流氓第三方压缩软件。

至此你已经入门了。如果你天赋异禀骨骼清奇，你现在就可以去编写自己的机甲智能体了！

2 参数详解

这里详细介绍与开发有密切关系的一些参数。

2.1 装备 weapontypename enginetypename

两个枚举类型，在【RobotAIstruct.h】中可以找到相应的部分，在 **ChooseArmor 函数**中赋值即可选定相应机甲

表格 1 装备枚举类型

weapontypename	对应武器中文名称	enginetypename	对应载具中文名称
WT_Cannon	加农炮	ET_Spider	蜘蛛 V3
WT_Shotgun	霰弹枪	ET_Robotman	毁灭者
WT_RPG	火箭筒	ET_AFV	战锤坦克
WT_Machinegun	重型机枪	ET_UFO	幽浮
WT_Prism	光棱	ET_Quad	四轴飞行器
WT_Tesla	磁暴线圈	ET_GhostTank	幽灵坦克
WT_PlasmaTorch	等离子发射器	ET_XCraft	X 战舰
WT_MissileLauncher	枭龙飞弹	ET_Shuttle	太空要塞
WT_ElectricSaw	钢牙电锯		
WT_GrenadeThrower	手雷弹射器		
WT_MineLayer	布雷塔		
WT_Apollo	阿波罗电磁炮		

类似的枚举类型还有 **bullettypename**，在获取战场中的子弹信息会用到，与 **weapontypename** 类似，在【RobotAIstruct.h】中可以找到，在此不再赘述（能用到这个肯定不是菜鸟了，不必再用这种方法讲解了吧☺）。

各装备的参数详情请参见下标，玩家参阅之后即可根据这些数据选择最适合自己打法的机甲。参数如有更新请及时关注大赛官网或官方 QQ 群。

中文名称	枚举类型（weapontypename）	命中伤害	子弹速度	弹仓容量	冷却时间	射击精度范围（度数）	炮塔旋转速度	炮塔半径
加农炮	WT_Cannon	25	11	10	30	5	5	95
霰弹枪	WT_Shotgun	10	10	8	80	5	2	65
火箭筒	WT_RPG	35	8	5	100	3	5	85
旋转机枪	WT_Machinegun	7	11	30	5	7	4	69
光棱	WT_Prism	20	-	7	70	5	1	76
磁暴线圈	WT_Tesla	22	-	11	40	15	3	45
等离子发射器	WT_PlasmaTorch	18	5	12	25	5	1.5	75
爱国者飞弹	WT_MissileLauncher	15	4	8	100	5	2	50
钢牙电锯	WT_ElectricSaw	5	-	50	1	5	2	95
手雷弹射器	WT_GrenadeThrower	25	6	15	40	20	5	50
布雷塔	WT_MineLayer	45/4	0/5	4/40	50	5	3	50
阿波罗电磁炮	WT_Apollo	25	14	8	50	3	5	100

武器特殊功能描述：

【霰弹枪】每个母弹包含 5 枚子弹，开火时散射而出，每两个子弹之间发射角相差 5 度。

【火箭筒】发射的火箭弹有推进器，做加速直线运动（v=1.05*v）。击中任意物体时引发爆炸，对半径 100 像素内的任何非击中机甲（包括发射者）造成 35 的伤害。

【光棱】发射高能激光束（射线 Beam），可理解为子弹射速为无穷大。

【磁暴】发射高能击穿电弧（射线 Beam），与光棱类似，其他数据上有差别。

【等离子发射器】发射出的等离子体在命中敌方机甲前可以触壁反弹 2 次。

【爱国者飞弹】发射出跟踪导弹，自动追踪敌方机甲，但请注意他并没有智能以绕开障碍物，旋转角速度 1.5。

【钢牙电锯】近战武器。

【手雷弹射器】手雷在飞行时间（100 帧内）不会爆炸，触壁反弹，飞行时间结束后立刻爆炸产生范围伤害。

【布雷塔】两种子弹模式：

中文名称	枚举类型（enignetypename）	最大行驶速度	护甲生命	加速度	引擎旋转速度	引擎半径
蜘蛛 V3	ET_Spider	4	85	-	-	45
毁灭者	ET_Robotman	5	90	-	3	48
战锤坦克	ET_AFV	6	95	0.15	3.5v	50
幽浮	ET_UFO	6	100	0.2	4.5	46
四轴飞行器	ET_Quad	4	85	-	-	45
幽灵坦克	ET_GhostTank	6	95	0.2	3.5v	50
X 战舰	ET_XCraft	6	100	0.2	4.5	46
太空要塞	ET_Shuttle	1	180	-	2	70

引擎移动方式描述：

【蜘蛛 V3，四轴飞行器】最普通的 RPG 游戏中的人物走法：上下左右四个方向，无需加速。

【毁灭者，太空要塞】只能沿当前朝向前后移动，无需加速，旋转半径为 0，且同一时刻旋转与移动不能同时发生。

【战锤坦克，幽灵坦克】移动方式最接近现实中的汽车：只能沿朝向前进（不能倒车），需要加速，旋转的角速度正比于当前径向速度。（vr=3.5*v）

【幽浮，X 战舰】漂浮式，需要加速，当前速度方向与加速度方向无关，旋转时旋转的是加速度方向而非当前速度方向。（加速度方向的旋转角速度 4.5）

2.2 指令 RobotAI_Order

对 Update 函数的第一个参数 RobotAI_Order& order 它赋值即可在该帧对机甲进行操纵。

```
struct RobotAI_Order
{
//用于RobotAI中Update(..)方法的对机器人下达的操作命令
int fire;    //控制武器开火与否
int wturn;   //控制武器旋转与否
int run; //引擎操纵码之一，影响速度，具体功能因所选引擎而异
int eturn;   //引擎操纵码之一，影响旋转，具体功能因所选引擎而异
RobotAI_Order(){fire=0;wturn=0;run=0;eturn=0;}
};
```

对于不同的装备（特别是载具），操纵码的含义不同。

表格 2 武器操纵码

操纵码成员变量	值>0	值==0	值<0
fire(大多数)	武器开火	无动作	
wturn	武器顺时针旋转	无动作	武器逆时针旋转
操纵码成员变量	**0**	**1**	**其他**
fire(枭龙飞弹)	跟踪目标 id=0	跟踪目标 id=1	无动作
操纵码成员变量	**1**	**2**	**值<0**
fire(布雷塔)	埋雷	发射小型骚扰子弹	无动作

表格 3 载具操纵码

操纵码成员变量	值>0	值==0	值<0
run(毁灭者，太空要塞)	全速前进	立即停止	全速后退
run(战锤坦克，幽灵坦克)	加速前进	刹车	
run(幽浮，X 战舰)	沿当前 rotation 加速	自然减速	沿当前 rotation 加速

操纵码成员变量	1	2	3	4	其他值
run(蜘蛛 V3，四轴飞行器)	左	右	上	下	无动作

操纵码成员变量	值>0	值==0	值<0
eturn(毁灭者，太空要塞)	原地顺时针旋转	停止旋转	原地逆时针旋转
eturn(战锤坦克，幽灵坦克)	右拐（顺时针）	无动作	左拐（逆时针）
eturn(幽浮)	加速方向顺时针旋转	无动作	加速方向逆时针旋转
eturn(蜘蛛 V3)	无动作		

【注意 1】操纵码的赋值与事件的发生并不等价。例如，当武器弹药耗尽或冷却时间未归零时，即使 fire>0 也无法进行开火生成子弹；再如，当引擎达到最高速时（战锤坦克和幽浮），或行进方向被阻挡时，即使 run>0 也无法使机甲行进到相应位置。

【注意 2】毁灭者的特性：当 eturn!=0 时（即旋转时），无法前后移动，即使 run!=0。（即，移动和旋转不能同时执行，旋转优先级大于移动）

【注意 3】武器炮塔的角度是独立的，与引擎载具的角度无关，即不会因为引擎载具的转动而转动。

【注意 4】只要机甲与军火库接触，并且军火库已经冷却完成，不需任何操作，机甲将补满弹药，军火库重新冷却。

【提示】如何理解四种引擎载具的移动方式：蜘蛛 V3 是最普通的 RPG 人物走法：上下左右四个方向，无需加速和旋转；毁灭者是只能沿当前朝向前后移动，无需加速，旋转半径为 0，且同一时刻旋转与移动不能同时发生；战锤坦克的移动方式最接近现实中的汽车：只能沿朝向前进，需要加速，旋转的角速度正比于当前径向速度；幽浮就是飞碟那样的漂浮状，需要加速，当前速度方向与加速度方向无关，旋转时旋转的是加速度方向而非当前速度方向。

2.3 战场信息 RobotAI_Information

Update 函数的第二个参数 **RobotAI_Information & info** 提供了你做决策所需要的几乎全部即时信息。它的定义同样你可以在【RobotAIstruct.h】中找到。其中聚合了一些类：

```
//每帧的战场总信息结构体
struct RobotAI_BattlefieldInformation
{
//使用最基本的数组存储

int num_robot;    //机甲数量（本届争霸赛衡为2了，包括已经挂了的）

RobotAI_RobotInformation robotInformation[Info_MaxRobots];


int num_bullet;   //当前子弹数量（便于循环访问于何时终止）

RobotAI_BulletInformation bulletInformation[Info_MaxBullets];


int num_obstacle; //地图上的障碍物数量（便于循环访问于何时终止）

Circle obstacle[Info_MaxObstacles];


int num_arsenal;  //地图上的军火库数量（便于循环访问于何时终止）

RobotAI_ArsenalInformation arsenal[Info_MaxArsenals];



Box boundary;     //战场边界，Box结构体
};
```

需要注意:

Circle 就是圆形；(x,y)圆心坐标，r 半径；

Box 就是矩形；(x,y)矩形中心坐标，width 水平宽度，height 高度；

Robot（机甲）的形状都是 Circle；

Bullet（子弹）的形状都是 r=0 的 Circle；

Obstacle（障碍物）的形状都是 Circle；

Arsenal（军火库）的形状都是 Circle；

boundary（战场边界）是 Box；

光棱和磁暴的子弹是瞬间的射线，所以不会在 bulletInformation 中出现。

Update 函数的第三个参数 int myID 则是自己机甲在 robotInformation[] 数组的下标。

2.4 静态信息获取

想要获取装备的数据？调用这些函数。正如注释所说的那样，你只需要 get_weapon_ammo(WT_Cannon)即可，传入的参数为 weapontypename ,enginetypename 或 bullettypename。

```
//这里是一些可以获取装备静态信息的做成宏的函数指针
//你可以理解为函数
//调用方法和调用函数看起来一样，比如我希望获得加农炮的弹仓子弹数：
//get_weapon_ammo(WT_Cannon);
#define get_weapon_name (*getWeaponName)
#define get_weapon_ammo (*getWeaponAmmo)
#define get_weapon_coolingTime (*getWeaponCoolingTime)
#define get_weapon_inaccuracy (*getWeaponInaccuracy)
#define get_weapon_rotationSpeed (*getWeaponRotationSpeed)
#define get_engine_name (*getEngineName)
#define get_engine_maxSpeed (*getEngineMaxSpeed)
#define get_engine_maxHp (*getEngineMaxHp)
#define get_engine_rotationSpeed (*getEngineRotationSpeed)
#define get_engine_acceleration (*getEngineAcceleration)
#define get_bullet_name (*getBulletName)
#define get_bullet_speed (*getBulletSpeed)
#define get_bullet_damage (*getBulletDamage)
```

2.5 触发函数组

这组函数是会在特定时刻被触发的。不是必须完成，你完全可以置之不理。

```
//Trigger Function
    virtual void onBattleStart(const RobotAI_BattlefieldInformation&,int)=0;
    //一场战斗开始时被调用
    virtual void onBattleEnd(const RobotAI_BattlefieldInformation&,int)=0;
    //一场战斗结束时被调用


    virtual void onSomeoneFire(int)=0;              //有机甲开火时被调用
    virtual void onHit(int,bullettypename)=0;       //被击中时被调用
```

onBattleStart 一场战斗开始时被调用，可能可以用来初始化
onBattleEnd 一场战斗结束时被调用，可能可以用来析构你动态分配的内存空间（如果你用了的话）
onSomeoneFire 当有机甲开火时被调用
onHit 被子弹击中时被调用

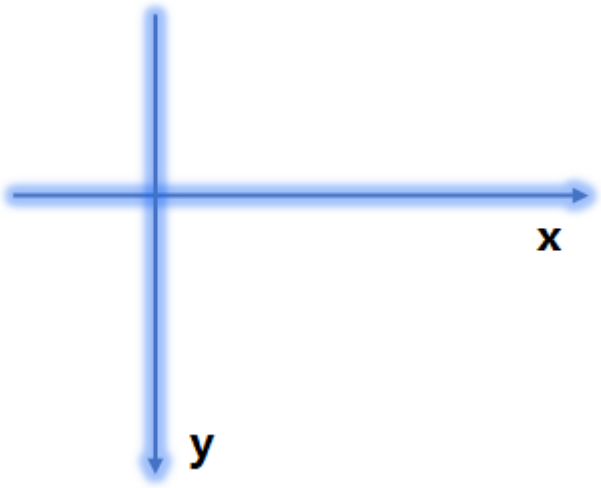
2.6 个性函数组

在【RobotAI.cpp】中。获取机甲名称，开发者（团队）名称，武器和引擎的颜色 RGB 偏移值。非常简单，真的只需要看代码注释就行了，我这里就不写了……

2.7 struct.h & GlobalFunction.cpp

主要是一些通用结构体和函数，可以视情况在代码中使用。也可以不使用而自己开发完整的代码，比如高手们。

2.8 坐标系统

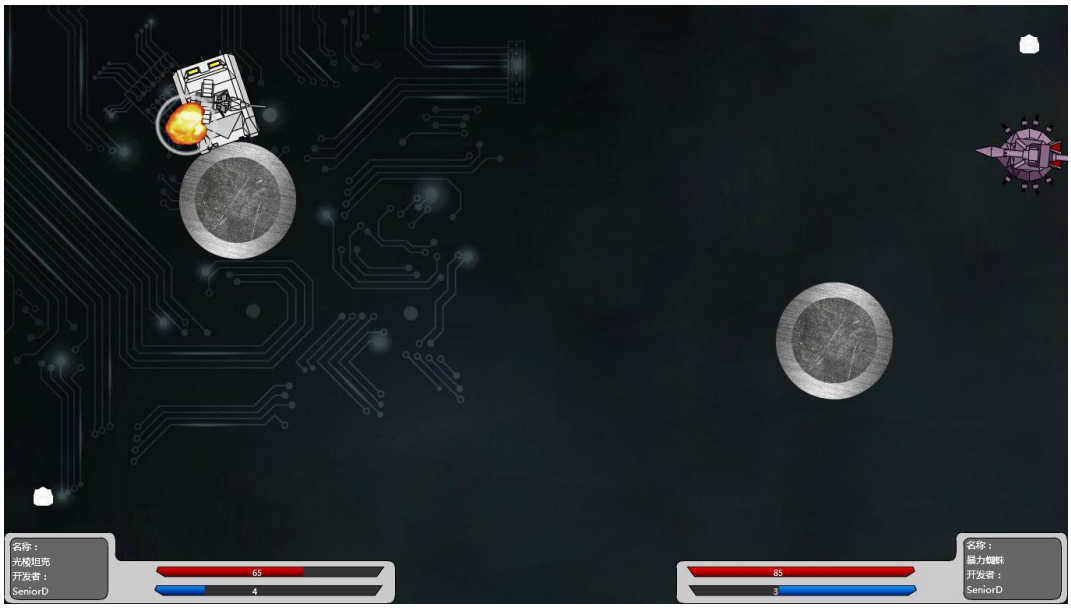


如图所示。地图的左上角坐标为(0,0)

这样一来，象限数从右下角开始顺时针数过来就是：一、二、三、四。在计算角度时（如用 atan2 时可能需要知道）

2.9 地图

本届大赛使用一张固定的地图以减小 AI 开发难度。地图的信息如下：



战场尺寸：1366*680
对于 Box boundary 而言
boundary.width=1366
boundary.height=680
boundary.x=683
boundary.y=340

出生点 0 坐标(50,50)
出生点 1 坐标(1316,630)

障碍物：2 个
障碍物 0：坐标(300,250)，半径 75
障碍物 1：坐标(1066,430)，半径 75

军火库（踩到上面吃掉可补充子弹）：2 个 
军火库 0：坐标(50,630)，半径 5
军火库 1：坐标(1316,50)，半径 5
重生时间：1000

当然也可以在代码中的 info 获取这些数据，动态规划路径

3 交流沟通

如果你还存在任何问题，欢迎加入官方 QQ 群：207730813，和大家一起讨论。

登录大赛官网获取其他信息：mechempire.cn