# 2022年江西省研究生数学建模竞赛

论文题目：基于强化学习的冰壶比赛策略方法研究

队号：20220033B

**摘 要**

冰壶比赛方案编排问题是一个多目标、多约束条件下的学习优化问题。首先，为了确定冰壶的运动规律和轨迹，需要对其进行受力分析并建立运动学与动力学模型。同时对冰壶比赛中所处的不同情况进行分析，对环境情况进行相应的数字化设置。最后在冰壶动力学模型的基础上通过强化学习的方法对冰壶比赛策略生成方法进行了探索，对算法的原理、优缺点、使用场景进行细致的分析。

我们完成了以下三方面的工作：一是不刷冰的情况下分别给出五种情况的对策，此时冰壶与冰面的摩擦系数处处相等。二是在考虑刷冰的情况下对五种情况给出对策，刷冰可以降低冰壶与冰面间的摩擦系数，从而减少冰壶的动能损失，使冰壶滑行地更远。同时，刷冰也能够调整冰壶的前进方向，可以让冰壶达到目标位置。三是在营垒前方已有冰壶的情况下对五种情况进行不刷冰及刷冰的分析，在冰壶比赛中通常会在营垒前方打出冰壶，作为占位壶或保护壶，分别对营垒前的壶是己方还是对方的进行分类讨论。

**关键词 冰壶比赛 动力学模型 强化学习**

目录

[1. 问题重述 1](#_Toc2277)

[1.1 问题背景 1](#_Toc16418)

[1.2 问题描述 1](#_Toc16418)

[2. 冰壶模型 3](#_Toc23700)

[2.1 实际场景中的冰壶战术 3](#_Toc16418)

[2.1.1剩余壶数 3](#_Toc30690)

[2.1.2自由防守区 3](#_Toc15154)

[2.1.3比赛得分 3](#_Toc12363)

[2.1.4冰面状况 4](#_Toc12363)

[2.2 冰壶运动学和动力学模型的实现 4](#_Toc29017)

[3.基于深度强化学习算法的策略生成方法 8](#_Toc29529)

[3.1问题一模型建立与求解 8](#_Toc28486)

[3.2问题二模型建立与求解 8](#_Toc20633)

[3.3问题三模型建立与求解](#_Toc2509) 8

[3.4问题四模型建立与求解](#_Toc2509) 8

[4.结论](#_Toc20361) 11

[5.参考文献](#_Toc24960) 12

[附录A 训练程序](#_Toc28847) 13

**第1章 问题重述**

* 1. **问题背景**

随着2022北京冬季奥运会的成功举办，冰壶以其趣味性，观赏性，技术性和协作性等特性成为备受广大观众的欢迎和关注的运动项目。

冰壶由壶体、手柄和螺栓组成，圆周约为91.44厘米，高度约为11.43厘米，质量约为19.96千克。比赛的标准场地长44.5米，宽4.32米，其中一端为本垒，画有一个直径为1.83米的圆作为球员的发球区；另一端为营垒，由4个直径分别为0.15米、0.61米、1.22米和1.83米的同心圆组成，如图1所示。场地前后两端各有一条蓝色的实线，分别称为前卫线和后卫线。冰壶掷出后，如果未通过前卫线或者超过后卫线都视作无效，将被清出场外。

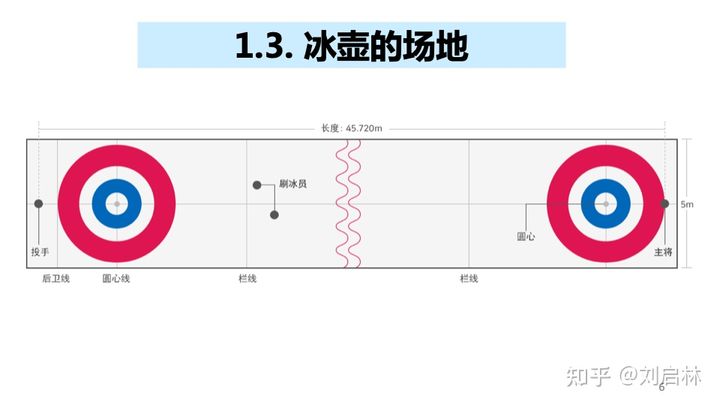


图1-1 冰壶场地图

每场比赛两队参加，各派4个队员上场，分为一垒、二垒、三垒、四垒，按顺序依次交叉投壶，每局16壶，共进行10局。一名队员投壶时，本队两名同伴可以在冰壶滑行前方用冰刷快速擦刷冰面，以控制冰壶的前进方向和速度。投壶方在力求将冰壶滑向圆心的同时，也可用冰壶将对方的冰壶撞出营垒或将场上本方的冰壶撞向营垒圆心。冰壶越过圆心线后，双方仅有一名队员可为己方的冰壶刷冰，且仅有主将可为对方的冰壶刷冰，以破坏对方战术。

冰壶运动不仅考验运动员对投掷冰壶的角度与速度的把控，同时也是双方队员博弈的过程，是一种兼具运动性、策略性与对抗性的体育项目。由于冰壶比赛的特性，计算机可以通过算法完成投掷策略的生成，实现计算机与人的对抗。但由于运动员投掷冰壶的角度与速度均为连续型变量，且投掷过程中由于场地因素导致不确定性，算法如何生成投掷策略，是一个有意义的科学研究问题。

**1.2问题描述**

从本质上来说冰壶比赛方案编排问题是一个多目标、多约束条件下的学习优化问题。首先，为了确定冰壶的运动规律和轨迹，需要对其进行受力分析并建立运动学与动力学模型。此外，需要对冰壶比赛中所处的不同情况进行分析，对环境情况进行相应的数字化设置。之后在冰壶动力学模型的基础上通过强化学习的方法对冰壶比赛策略生成方法进行了探索，对算法的原理、优缺点、使用场景进行细致的分析。

该建模比赛赛题的任务是针对不同的情况给出相应的对策。一共有五种情况：(1) 当营垒中没有冰壶时，使得冰壶停在营垒中；(2) 当营垒中有一个对方的冰壶时，将对方的冰壶撞出营垒，而自己的冰壶停在营垒中；(3) 当营垒中有一个对方的冰壶时，将对方的冰壶撞出营垒，同时自己的冰壶一起出营垒；(4) 当营垒中有两个对方的冰壶时，将对方的两个冰壶都撞出营垒，而自己停在营垒中；(5) 当营垒中有对方的冰壶，也有己方的冰壶时，将对方的冰壶撞出营垒，而己方的冰壶仍留在营垒中。

在问题一中，不刷冰的情况下分别给出五种情况的对策，此时冰壶与冰面的摩擦系数处处相等。问题二，在考虑刷冰的情况下对五种情况给出对策，刷冰可以降低冰壶与冰面间的摩擦系数，从而减少冰壶的动能损失，使冰壶滑行地更远。同时，刷冰也能够调整冰壶的前进方向，可以让冰壶达到目标位置。问题三是在营垒前方已有冰壶的情况下对五种情况进行不刷冰及刷冰的分析，在冰壶比赛中通常会在营垒前方打出冰壶，作为占位壶或保护壶，这需要对该营垒前的壶是己方还是对方的进行分类讨论。

1. **冰壶模型**

**2.1 实际场景中的冰壶战术**

**2.1.1 剩余壶数**

冰壶是一个团队合作的比赛，四个球员分别负责一至四垒，每人连续投两个壶。在一局比赛中每一垒次的投壶都有其重要的意义，从一垒的投壶中大致确定比赛战术，要求一垒具有极佳的技术稳定性，以确保本队战术执行的前提；二垒的投壶开始出现一些破坏对方战术布置的作用，也是一局比赛的转折点，当二垒的两只壶投掷结束代表着一局比赛已经进行了一半，此时队长会根据此时场上的情况来决定后四个壶的战术；三垒需要随时投出从设置障碍壶到清理障碍壶各个阶段的力量，需要极强的力量调整能力与处理复杂局面的准备，从而给四垒的最后两投留下选择的空间；四垒的两只壶决定着本局比赛的最终分数，所以战术选择时需慎重进行决定。

随着一局比赛的进行，剩余壶数的不断减少，比赛战术的选择与调整要具有一定的预见性，队长要根据总体战术的方向布置不同垒次的任务，需要提前预见本场比赛接下来4-5个壶的发展，由此来及时转换攻防战术。

**2.1.2 自由防守区**

自由防守区的定义是比赛端的 T 线与前掷线之间，除大本营外的区域，被指定为自由防守区。撞击过自由防守区内的冰壶的未出界的冰壶（在前掷线上或前掷线前）所在的区域，也被认为是在自由防守区。

自由防守区内的壶并不在得分范围内，主要用于设置占位壶，不同战术决定着自由防守区的情况，在较为进攻性的战术打法中自由防守区会更为重要，尤其是后手想拿大分、先手想偷分的这些情况下自由防守区成为必争之地，总体来说先手会在中心区域布置对自己有利的障碍壶，从而用于在之后的投壶中保护自己的得分壶以及给对手布置障碍等待机会偷分，后手则是在边区布置属于自己的障碍壶，设法在边区形成得分局势并清理中区障碍，从而为四垒最后一投建立良好的局势。

**2.1.3 比赛得分**

在真实的冰壶比赛中，一局比赛的得分情况分为先手控分、偷分，后手得分、空局四种情况，赛中的比分、本局想要得到的分数或接下来几局需要取得的分数都会影响战术的选择。

大比分领先的队伍通常情况下会选择防守型战术，以保持自己的得分优势不给对手偷分或得两分及以上的机会，最大程度降低自己在本局中的损失；而比分落后的队伍则会更多的进攻型的战术，从而在复杂局面中伺机偷分或得两分及以上，但风险和利益总是相伴的，在这种局势中失败也是送对手一个得分的好机会；最后一种就是比分接近，这时双方都有可能会选择混合型战术，在相互试探中及时转换攻守时机，抓住关键壶得分或逼迫对手投出更难的壶来得分，但混合型战术并不代表会摇摆在进攻与防守之间，除了队员连续的失误不得不改变战术外，队长也是会根据自己队伍的节奏来制定战术。

**2.1.4 冰面状况**

在实际比赛中，冰面状况至关重要。整个冰面也并不是完全的水平，冰点的密度和数量以及冰面的弧度与形状都不是完全可控的，现实中不存在两个完全相同的冰面状况。而冰壶的每一场比赛开始前都需要制冰师重新制冰，上半场与下半场的休息间也要对场地进行适当的修整，甚至在比赛当中随着时间的进行、温度和湿度的变化、冰壶走的次数和路线、擦冰的频率和力度等等都在无时不刻的改变着冰面的状况。

而不同的冰面条件会有不同投壶战术的选择，若冰面光滑，即弧线较小时比较适合击打技术的发挥；若冰面较涩，即弧线较大时，比较适合投旋壶。应对不同的冰面情况，采取不同的刷冰策略，对取得比赛的优胜也至关重要。

**2.2 冰壶动力学和运动学模型的实现**

冰壶运动的场地坐标系如图2-1所示。

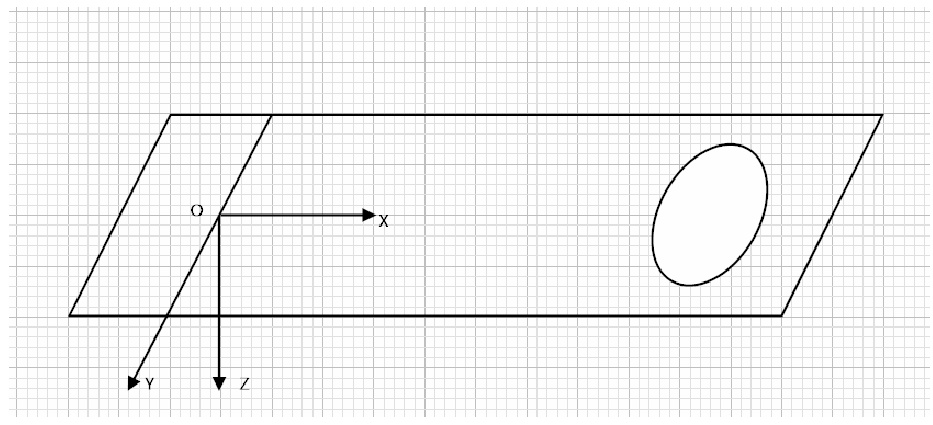


图2-1冰壶运动的赛道坐标系

其中轴与栏线垂直，轴正向指向营垒区;轴与栏线平行;轴垂直于场地平面。冰壶运动过程可以近似认为二维平面坐标有关，在后续建模只考虑场地的轴与轴。

冰壶运动及碰撞过程如图2-2所示。

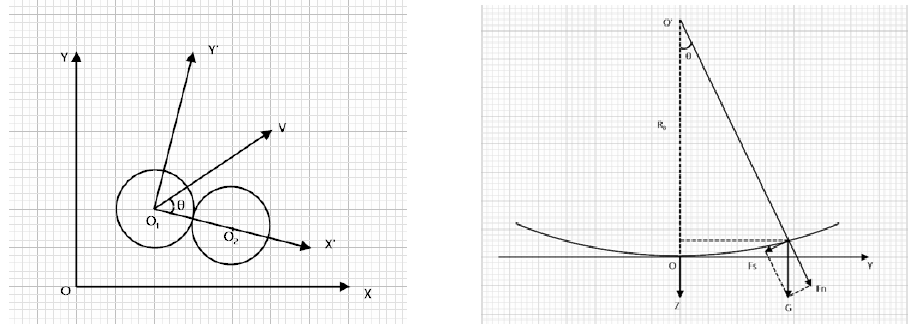


图2-2 冰壶在U型赛道碰撞及滑行的受力分析

冰壶在滑行过程中，主要受摩擦力、下滑力、旋转压力三种力影响。摩擦力,源于冰壶向前方运动的过程，方向与速度方向相反;下滑力,源于重力被U型赛道的斜面分解;旋转压力源于冰壶自身旋转与冰面产生的作用力。三者计算公式如下:

 (2-1)

 (2-2)

 (2-3)

碰撞过程建模较为复杂，假设和,均为非光滑冰壶，发生非对心碰撞，建立图2-3左侧所示的坐标系。设两冰壶之间的摩擦系数为，恢复系数为。当两冰壶接触时，力的分解示意图如下图所示。

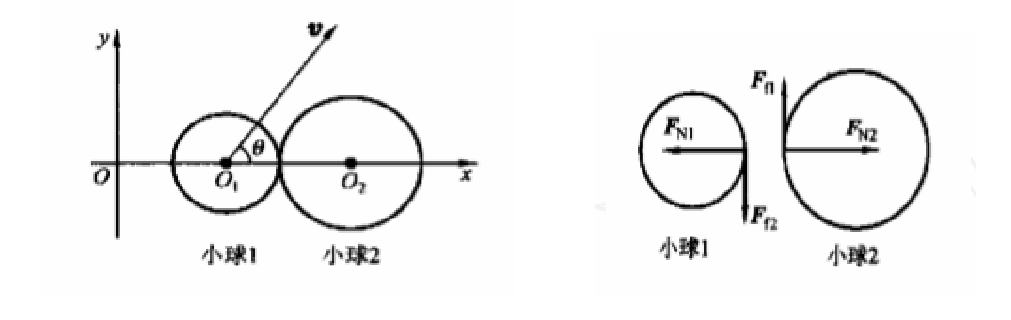


图2-3 冰壶碰撞过程受力分析

设运动壶碰撞前后的速度分别为和，角速度为和，静止壶碰撞前后的速度分别为和，角速度为和。

根据动量定理，列出方程组:

 (2-4)

与的转动惯量与,为:

 (2-5)

恢复系数为:

 (2-6)

解方程得到:

 (2-7)

其中。

当碰撞角小于临界角时，恢复系数改变，方程联合求解会有所变化：

 (2-8)

其中

1. **基于深度强化学习算法的策略生成方法**

**3.1 强化学习算法**

策略生成的强化学习算法采用深度确定性策略梯度方法，在网络的基础上进行改造，用来解决连续状态与动作空间上的深度强化学习问题。实验采用深度确定性策略梯度方法，以行动-评论为框架，进行策略评估与策略改进，具体流程如图3-1：

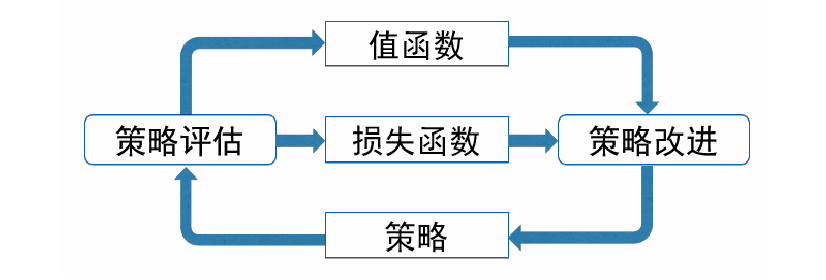


图3-1基于行动-评论框架的策略学习结构

在冰壶的策略生成场景中，我们应采用确定性策略进行策略生成，但同时又需要对搜索环境进行探索。因此首先使用随机策略产生样本存放到经验回放机制中，训练时随机抽取样本，进行确定性策略的改进。

强化学习实验中用于训练的网络结构如下图3-2:

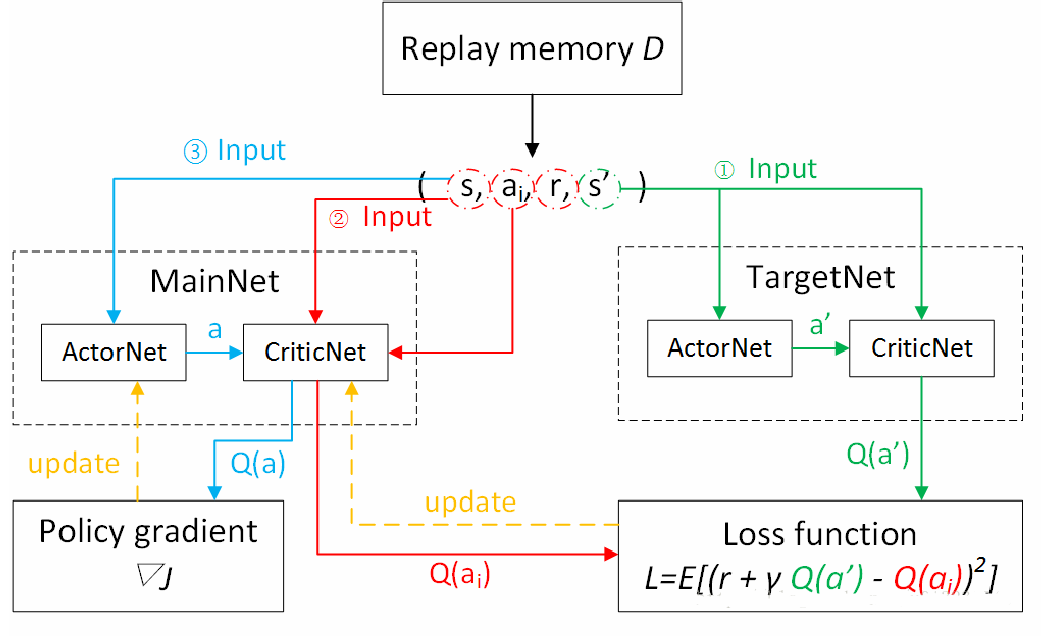


图3-2深度强化学习网络结构

其中称为策略网络，用两层全连接网络对确定性行为策略进行模拟，记其参数为，称为评判网络，同样使用两层全连接网络，对函数进行模拟，记其参数为。策略网络在训练过程中，目标为寻找最大化策略。评判网络在训练过程中，目标为精准预测值，拟合函数。根据评判网络的损失函数网络进行训练。

实验证明，单个网络结构的学习算法，学习过程很不稳定，因为网络的参数在频繁进行更新的同时，又用于计算网络和策略网络的梯度。因此将网络复制为两份，一份称为主网络()，另一份称为目标网络()。主网络主要用于参数的更新，而目标网络用于梯度的计算。同时目标网络通过 算法随着主网络进行更新。网络每次训练时，从回放内存中选取一定量的样本四元组，其中作为主网络中策略网络的输入，由策略网络生成状态下的最优动作解，然后经由评判网络评估动作的函数值，根据与对策略网络进行更新。同时目标网络计算得到状态下的最值，根据公式3-1与3-2:

 (3-1)

 (3-2)

可以认定为，为，从而依照损失函数式3-3更新网络参数：

 (3-3)

根据两个损失函数梯度更新主网络中的策略网络与评判网络后，目标网络参数进行 ，更新式为3-4与3-5:

 （3-4）

 （3-5）

目标网络中的参数在每次迭代后都以微小量逼近主网络中的参数。算法在迭代过程中,根据方程,评价网络的函数会逐渐减小,如此就越逼近真是的函数值，场景建模也就越来越精准。

**3.2 模型的建立与求解**

问题分析：

在组员相互讨论的过程中，我们从University of Northern British Columbia的物理学教授Mark Shegelski的论文中得到启发：

首先摩擦力是影响冰壶运动的至关因素，特别是冰壶的速度变慢，就会导致朝一个方向偏转，路径也变成了弧线。如果是逆时针偏转，冰壶就会旋向左边，如果是顺时针，就会旋向右边。Shegelski表示，擦刷的动作通过加热冰面和减少摩擦力，令冰壶石的旋转角度变小，从而让它的路径变得更直。与没有任何干扰相比，在冰壶石前方大力地擦刷冰面，能让冰壶石前进数米更远。

我们将冰壶的运动进行了两种类型的划分，根据冰面的不同状态，分为刷冰和不刷冰两种摩擦属性。我们这里假设在每种摩擦属性中，模型遵循不同的非线性变化规则，得到一个非线性模型(s, a, h\_x) = f(x, y)其中x, y是要在该模型下期望到达的位置，s为得到的速度，a为旋转的角度，h\_x为x坐标轴的位置,。这里我们通过人造的数据进行实验，通过数据拟合得到了不同阶段之间分隔点的速度阈值、不同模型中的参数变化、各个参数随着速度与刷冰不刷冰的变化而变化。此处的物理模型不再是套用公式对整体轨迹进行建构，而是找出速度角度与位移的关系，通过时间序列的参数模拟来得到冰壶运动的轨迹。通过这种方法，在我们的模型上冰壶正常运动和冰壶之间碰撞时，可以基本上做到实际位置与期望达到的目标相比误差很小。

问题1在不刷冰的情况下分别采取何种策略实现以下目标

（1）当营垒中没有冰壶时，使得冰壶停在营垒中。

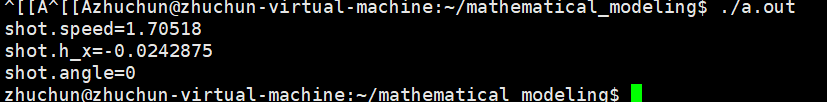
我们假设冰与冰壶的摩擦因数为0.02，刷冰时摩擦因数为0.01透过代码多次拟合我们得到v与y的非线性关系给定y来求解需要的速度v与y一次关系拟合：

你把它变成公式呀

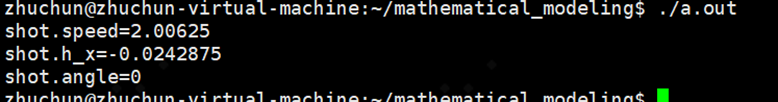
v = -0.1118y + 3.5378(sqrt(-0.6324 \* y + 12.047) + sqrt(-0.5912 \* y + 11.768) + (-0.1118 \* y + 3.5378)) / 3;

设置要到达的位置（0，28.3）得出

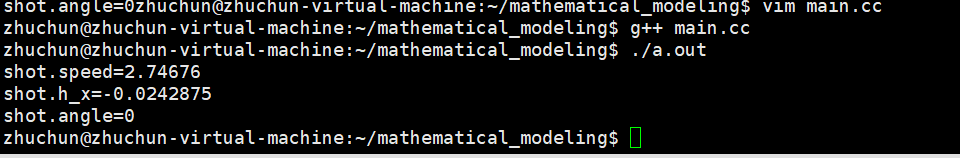
即得出当我们设置松手时速度为2.74676 ，在x轴的偏移量为-0.0242875 角度为0，就可以，到达营垒



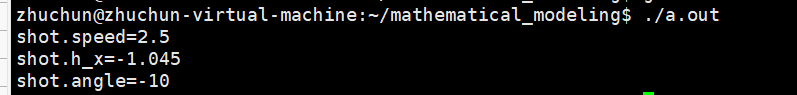
（2）当营垒中有一个对方的冰壶时，将对方的冰壶撞出营垒，而自己的冰壶停在营垒中。



（3）当营垒中有一个对方的冰壶时，将对方的冰壶撞出营垒，同时自己的冰壶一起出营垒。

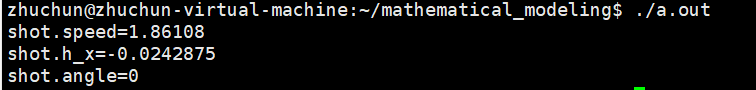


（4）当营垒中有两个对方的冰壶时，将对方的两个冰壶都撞出营垒，而自己停在营垒中。

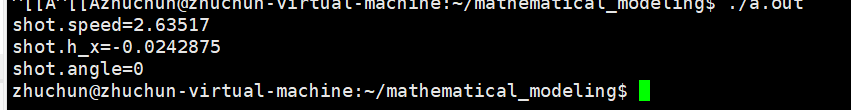


问题2在刷冰的情况下分别采取何种策略实现以下目标

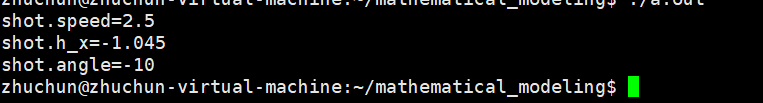
1. 当营垒中没有冰壶时，使得冰壶停在营垒中



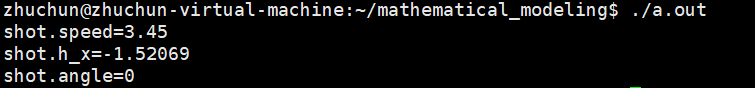
1. 当营垒中有一个对方的冰壶时，将对方的冰壶撞出营垒，而自己的冰壶停在营垒中。



（3）当营垒中有一个对方的冰壶时，将对方的冰壶撞出营垒，同时自己的冰壶一起出营垒。

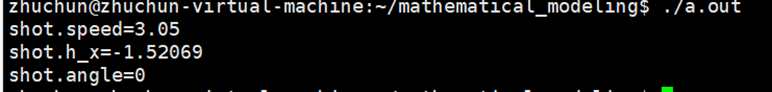


（4）当营垒中有两个对方的冰壶时，将对方的两个冰壶都撞出营垒，而自己停在营垒中

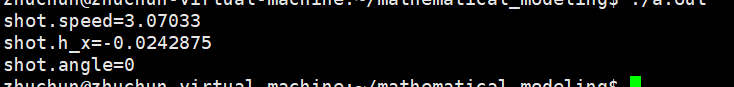


问题3在不刷冰的情况下分别采取何种策略实现以下目标

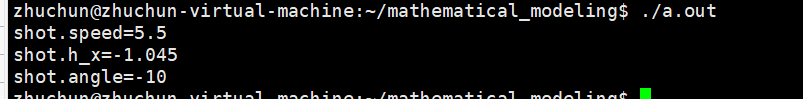
（1）当营垒中没有冰壶时，使得冰壶停在营垒中



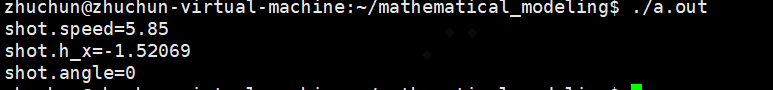
（2）当营垒中有一个对方的冰壶时，将对方的冰壶撞出营垒，而自己的冰壶停在营垒中。



（3）当营垒中有一个对方的冰壶时，将对方的冰壶撞出营垒，同时自己的冰壶一起出营垒。



（4）当营垒中有两个对方的冰壶时，将对方的两个冰壶都撞出营垒，而自己停在营垒中。



1. **结论**

**5.参考文献**

[1]Mark R. A. Shegelski,Ross Niebergall, Mark A.Walton.The motion of a curling rock.[J].Canadian Journal of Physics, 1996, Vol.74,No.9-10: pp.663-670

[2]Barto A G,Duff M.Monte Carlo Matrix Inversion and Reinforcement Learning[C].Advances in Neural Information Processing Systems.Denver: NIPS,1993:687-694.

[3]柳悦.浅谈冰壶比赛中的投壶技术特点[J].科技资讯,2019,17(31):251-253.

[4]刘家楠. 中国女子冰壶队历届冬奥会比赛战术运用的研究[D].首都体育学院,2019.

[5]于黎冰.我国冰壶运动发展现状及研究进展[J.冰雪运动，2007，29(2):45-47.

[6]国际冰壶联合会规则委员会:冰壶与冰壶竞赛规则，2011-2-9

[7]穆亮. 冰壶竞技战术研究[D].东北师范大学,2015.

**附录A 主要计算程序**

//主要计算函数

bool shotForStraightBall(double x\_coordinate, double y\_coordinate) {

/\*投直线球到达x y的位置 \*/

/\*当直线轨迹有球时 返回false\*/

vector<vector<int>> distribution;

distribution = distributionAroundABall(x\_coordinate, y\_coordinate);

if (distribution[2].size() == 0) {

shot.speed = speedAtFixedDistance(y\_coordinate);

shot.angle = 0;

shot.h\_x = x\_coordinate - TEE\_X - x\_shot\_error;

return true;

}

else {

return false;

}

}

double speedAtFixedDistance(double y) {

/\* 给定y来求解需要的速度v \*/

/\* v与y一次关系拟合：v = -0.1118y + 3.5378 \*/

return (sqrt(-0.6324 \* y + 12.047) + sqrt(-0.5912 \* y + 11.768) + (-0.1118 \* y + 3.5378)) / 3;

}

bool shotForCurveBall(double x\_coordinate, double y\_coordinate) {

/\* 给定坐标 发射曲线球 曲线贴或者曲线定位 \*/

/\* 若有球阻挡 则返回false \*/

/\* w最大为10时 纵坐标越小 x\_err绝对值越大 \*/

/\* 可调节的参数：tmpx\_err y\_dist\_max overlap\*/

double x\_err; // 初始横坐标与目标横坐标的差值

vector<vector<int>> distribution;

distribution = distributionAroundABall(x\_coordinate, y\_coordinate);

if (distribution[2].size() == 0) {

// 如果坐标的正下方没有球

return shotForStraightBall(x\_coordinate, y\_coordinate);

}

else {

// 如果球的正下方有球

// 离他最近得球最可能碰到

// 判断离y\_coordinate最近的球的距离 是否足够远

double y\_dist;

int j = 0;

double tmpx\_err = 1.3;

double y\_dist\_max = 4.2;

// 获取下方与它最近的球的距离

if (IfOccupied(x\_coordinate, y\_coordinate) == -1) {

y\_dist = 1e6;

}

else {

j = IfOccupied(x\_coordinate, y\_coordinate);

y\_dist = GameState.body[j][1] - y\_coordinate;

}

if (y\_dist < y\_dist\_max) {

// 如果不是足够远，则不能旋到准确位置，有一定偏差

// 确定重合的系数overlap

double overlap = -0.36\*y\_dist + 1.522;

if (overlap > 1) {

// 下方的球靠的很近 肯定碰上

return false;

}

if (fabs(GameState.body[j][0] - x\_coordinate) < 0.015) {

if (rand() % 2 == 0) {

// 从左边打

x\_coordinate = x\_coordinate - overlap \* STONE\_R;

}

**附录B 模型设计程序**

typedef struct \_GAMESTATE {

int ShotNum; // number of Shot // 第几次投递

// if ShotNum = n, next Shot is (n+1)th shot in this End

int CurEnd; // (number of current end) - 1， 当前是第几局

int LastEnd; // number of final End ，初赛4局，决赛8局

int Score[8]; // score of each End (if Score < 0: First player in 1st End scored)

bool WhiteToMove; // Which player will shot next

// if WhiteToMove = 0: First player in 1st End will shot next,

// else (WhiteToMove = 1) : Second player will shot next

double body[16][2]; // body[n][0] : x of coordinate of n th stone // stone冰壶的坐标

// body[n][1] : y of coordinate of n th stone

// 每队8个，共16个冰壶

}GAMESTATE, \* PGAMESTATE;

typedef struct \_ShotInfo //每次击打的信息：速度

{

\_ShotInfo(float s, float h, float a)

{

speed = s; // 速度

h\_x = h; // 宽度x的坐标

angle = a; // 角度

};

float speed;

float h\_x;

float angle;

}SHOTINFO;

typedef struct \_MOTIONINFO // 运动的信息

{

float x\_coordinate; // x的位置

float y\_coordinate; // y的位置

float x\_velocity; // x轴的速度

float y\_velocity; // y轴的速度

float angular\_velocity; // 角速度