独创性声明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。据我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得安徽大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

学位论文作者签名：彭发恩　　签字日期：2013 年 5 月 30 日

学位论文版权使用授权书

本学位论文作者完全了解安徽大学有关保留、使用学位论文的规定，有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许论文被查阅和借阅。本人授权安徽大学可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。

（保密的学位论文在解密后适用本授权书）

学位论文作者签名：彭发恩　　导师签名：李永清

签字日期：2013 年 5 月 30 日　　签字日期：2013 年 5 月 30 日

学位论文作者毕业去向：

工作单位：　　电话：

通讯地址：　　邮编：

摘要

RoboCup 机器人世界杯赛是近年来规模最大并且参与人数最多的高智能机器人足球比赛，其中仿真 2D 组比赛更是 RoboCup 世界杯中最古老的比赛项目之一，其中的多 agent 智能系统协作问题也是当前最热门的研究方向之一。它为研究人员提供了一个标准的比赛平台、命令和动作接口。RoboCup 联盟期望通过提供这样的一个标准问题，鼓励研究人员通过各种技术手段来获得人工智能和 MAS(Multi-agent System,多智能体系统)方面的解决方案，并且可以通过该平台进行测试和实践。

在 RoboCup 联盟提供比赛平台的基础上，各个学校和研究机构开始研究学习算法，使得每个 agent 具有一定的学习能力，这样才能保证在比赛中球队能立于不败之地，其中 Q 学习是最广泛采用的学习算法之一。

本文在 Q 学习基础上对球队的改进如下：

1、 采用随机对策的决策过程来代替单 agent 的马尔可夫决策过程以解决 MDP 决策过程不能对球场的状态变化有很好的反应的问题

2、 根据随机对策的内容将复杂的球场环境分成三种情况分别讨论(自己控球，我方控球和对方控球)，并确定每种情况对应不同的状态-动作对。

3、 通过对球场的精确划分和采用相应的补充控制方法，来对 Q 学习中的关键参数奖赏值进行计算以确保奖赏值的连续性和单调性，同时根据此参数和改进的算法来更新 Q 值使得机器人的学习效果更好。

实验证明，通过对局部范围内的机器人都采用此算法，可以让球员间在局部范围内的协作能力有明显的加强，从而使球队的整体攻防能力有所加强，但是受到计算机硬件、软件和计算能力的限制，无法对全局 agent 都采用此算法，否则无法保证比赛的实时性。在此理论基础上实现的安徽大学 DreamWing2D 队伍在 RoboCup 中国公开赛仿真 2D 组中获得的成绩证明了此算法的可行性。

关键字： RoboCup 世界杯；局部战术策略层；Q 学习；多 agent；随机对策

Abstract

Being the largest intelligent football match in recent years, RoboCup (robot World Cup) has the largest number of people involved. And the Simulation 2D match, whose problem of multi-agent system cooperation is one of the hot research topics, is ranked as one of the most traditional matches in the World Cup. It offers the researchers a standard platform as well as orders and operations. By doing so, RoboCup league encourages its researchers to obtain the solutions to problems concerning artificial intelligence and Multi-agent System through a variety of methods. It also enables them to test and practice via this platform.

Based on the platform provided by RoboCup league, every school and research institute start researching study algorithm, which enables each agent to acquire the learning ability, thus guaranteeing the invincible status of the team created by schools and research institutes. And Q method is one of the widely used study algorithms.

The improvements of the team based on the Q-Learning are as follows:

1、Owing to the dynamic course and the multi-agent environment, MDP decision making process is not capable of reacting properly to the changes in the court. We adopt the Stochastic Game to replace the MDP.

2、According to the Stochastic Game, complicated field environment will be discussed in three cases separately (agent controls the ball, our team control the ball, the other team control the ball), each case has a different state--action pair.

3、By adopting supplementary control method and dividing the court precisely, Reward values of the key parameters in Q learning are calculated to ensure the continuity and monotonicity of reward value. At the same time, the Q value will be updated through the parameters and the improved algorithm, making the robot learning effect better.

Experiments show that using this algorithm can strengthen the cooperation ability among the players obviously. Therefore the team's overall attack and defense ability can be enhanced. However restricted by computer hardware, software and

computing capacity, this algorithm can not be used in all agents; otherwise it is impossible to ensure a real-time game. Based on the theory, DreamWing2D of Anhui University team achieved success in the RoboCup China Open Simulation in 2D, which proves the practicability of this algorithm.

Keyword: Robot World Cup; local tactics layer; Q-learning; Multi-agent; Stochastic Game

目 录

第一章 绪论

1.1 RoboCup 介绍

机器人足球竞赛以其集高科技、比赛和娱乐于一体的特点在近年来国际上迅速开展起来。它涉及到人工智能、智能控制、视觉技术、通讯传感技术以及材料制造技术等多领域的前沿研究和技术融合，其不仅可以极大培养广大学生对前沿学科的兴趣，还可以激发学生们在技术上的创造性。目前国际上推出了不同类型的机器人比赛，如机器人足球、机器人救援、机器人投篮、机器人灭火以及机器人舞蹈等多种比赛。其中以机器人足球赛最为引人注目的。

目前国际上比较流行的是两大机器人足球竞赛——RoboCup 和 FIRA，其中 RoboCup 机器人足球世界杯赛及学术大学(The Robot World Cup Soccer Games and Conferences)是国际上级别最高、影响最广泛、规模最大的机器人足球赛事和学术会议，每年举办一次。它是由总部设在瑞士的 RoboCup 联盟发起组织并举办起来的，目前有 40 多个国家参加了这个组织。随着 RoboCup 的不断发展，其在世界的影响力也在逐年增大。1999 年 5 月，由清华大学计算机系和中国科技大学计算机系一同发起，组织并建立了 RoboCup 中国分会，负责中国境内的一切 RoboCup 活动。

1.2 RoboCup 赛事概况

1997 年 8 月 25 日，第一届 RoboCup 机器人足球世界杯赛在日本名古屋举行，同时举办了国际人工智能联合会议(the International Joint Conference on Artifical Intelligence, IJCAI-97)。有 40 多支分别来自美、欧、日、澳的球队参赛。

1998 年 7 月，在第 16 届世界杯足球赛开赛的同时，第二届机器人世界杯赛在法国巴黎举行，参赛队达到 60 多支。

1999 年 7 月 28 日，第三届 RoboCup 世界杯赛在瑞典的斯德哥尔摩举办，与此同时 RoboCup 学术大会和 IJCAI-99 也同时举行，参赛队增长到 90 余支。

第四届 RoboCup 世界杯赛及学术大会于 2000 年 8 月 25 日至 9 月 3 日在澳

大利亚墨尔本隆重举行，正式参赛队达到 104 支，首次突破 100 大关。一些著名大学、国立研究院和大公司均参与了相关的活动，中国科学技术大学代表中国首次参赛，并取得第九名的好成绩。

第五届世界杯赛以及学术会议于 2001 年 8 月 2 日至 8 月 10 日在美国的西雅图举行，约有 100 余支球队参加。清华大学参加了仿真组，中国科技大学参加了四腿组以及仿真组的比赛，清华大学仿真组获得冠军，中国科技大学队进入双八。

第六届 RoboCup 于 2002 年 6 月 19-25 日在日本福冈举行，参赛球队数量更多。清华大学和北京理工大学分别获得 RoboCup 仿真组冠军和亚军，中科大仿真组和四腿组也取得了非常好的成绩。

第七届 RoboCup 于 2003 年 7 月 2-11 日在意大利帕多瓦举行，中国的中科大、清华、浙大、北京理工和上海交大参加了比赛并取得了不错的成绩，是 RoboCup 有史以来最盛大的比赛。比赛项目从最初的仿真组、小型组、中型组到仿真组(分为仿真 2D 组和仿真 3D 组)、小型组、中型组、Sony 四腿组、机器人营救组、中小学初级组以及类人机器人、机器人舞蹈表演赛。

第八至第十一届 RoboCup 分别在葡萄牙的葡京、日本的大阪、德国的不莱梅港、美国亚特兰大、中国苏州、奥地利格拉茨、新加坡、土耳其的伊斯坦布尔和墨西哥城举办，在举行机器人竞技比赛的同时，RoboCup 国际学术会议也同时举行，为机器人足球的进步作出了很大的贡献。

RoboCup 中国公开赛于 2006 年开始举办，每年一次，在 RoboCup 国际竞技项目的基础上，有中国科技大学发起并组织了项目的比赛，比如机器人救援仿真和服务机器人。将动态规划等问题列入研究范围，用于解决生活中的实际问题，这也将是机器人未来的研究重心。

1.3 RoboCup 仿真 2D 概述

RoboCup 仿真 2D 机器人足球是 RoboCup 最古老，参赛队伍最多的比赛项目之一，其由 RoboCup 仿真平台开发小组提供一个标准比赛软件平台用来模拟机器人进行足球比赛。该平台在设计上模拟了在控制、传感(视觉、听觉等)、通讯和人体体能等方面的实际限制，这些使得仿真球队程序容易转化成硬件球队的控制软件。还因为避免了现实球场环境(空间、大小等)和机器人制造技术的限制，

仿真机器人足球可以把主要研究重点放在球队的高级功能上，包括动态不确定环境中的多 Agent 合作、实时推理-规划-决策、机器学习和策略获取等当前人工智能的热点问题。正是由于不受到机器人硬件等的限制，使得仿真组比赛比较容易实现，也使其成为 RoboCup 世界杯赛中历史最老、参赛队伍最多的一个项目，其研究步伐也快于其他项目。

RoboCup 的比赛项目为人工智能的研究提供了一系列挑战计划，同时还提供了一个动态、多 Agent、实时的平台，通过比赛来检验研究成果。设计 RoboCup 仿真 2D 球队的根本问题就是设计一个多 Agent 系统，并且能够提供对现场环境实时的反应，并目标驱动进行理性行为。同时其目标与环境都是在动态地变化并且是实时的。由于足球比赛环境中要考虑的因素非常多，状态空间极大，用手工的方法来编码来完成所有可能的情形和实现 Agent 的行为变的几乎不可能。因此必须使 Agent 能学习如何有策略的进行比赛。在这个方面的挑战包括以下的研究问题：

(1) 在多 Agent 合作及对抗环境中的机器学习

(2) 多 agent 体系结构，为了团队合作而进行实时的多 Agent 规划和执行

(3) 对对手的建模

因此，agent 的学习、团队合作和对对手建模这三个挑战也就成为当前的研究重点。

1.3.1 agent 学习

RoboCup 中，agent 学习的目的是寻找综合全面的学习方案，不要要能应用与需要适应的多 agent 系统的学习中，还要能用标准任务评价学习方法的优缺点。学习是智能系统最重要的方面，在 RoboCup 的学习挑战中，其任务是为一组 agent 创建学习和训练的方法。学习的方法可以分为几类：

(1) 单 agent 的离线技术学习，例如踢球、射门等技术

(2) agent 团队内的离线合作学习，例如协防、传球等

(3) 多 agent 的在线对抗学习，例如预测对手行为并采取相应对策

(4) 多 agent 的在线协作学习，例如跑位

由于比赛一般只有 10 分钟，所以需要将产生结果慢的学习方法放在比赛之

前就训练结束，例如球员的踢球和射门的行为，不可能在比赛现场实时学习，因此将学习分为在线学习和离线学习。离线学习的内容主要是比较固定，受比赛的影响不大的行为，而在线学习则必须要很快的产生结构，比如，在球场上根据对方的阵型和进攻防守方式及时调整自身的策略的行为，这些针对具体对手的学习行为，必须要在比赛结束之前学习到，并及时运用。

1.3.2 团队合作

RoboCup 的团队合作提出了在动态环境中多 agent 之间在团队合作时的动态实时规划和规划执行等问题来充分应对对抗环境。在足球比赛这样动态、复杂的多 agent 系统中需要非常灵活的团队合作和通讯来克服其中的不确定性。比如，团队成员由于体力不足或计算过大等意外而导致计划中的任务不能完成时，团队的目标就需要动态变化。然而，当前的智能体体系结构中，要实现多 agent 系统必须经常将相关的领域事先规划出来，这不能提供上述的灵活性。所以要实现团队合作就必须建立一个适合团队实时规划和执行的体系结构，另外这样的体系结构也应当易于转化到其他非 RoboCup 应用中。

可以通过一个两层的体系结构来建立一种可以支持团队行为规划的体系结构的规划和规划执行，这也是多 Agent 在团队合作中的基本体系结构。在 RoboCup 仿真比赛中，由于 agent 会根据不同策略在 75-300ms 接收到传感数据，每 100ms 发送一条动作命令，所以体系结构也必须能够实时运行。环境以毫秒量级改变，所以规划、重规划和执行必须是实时完成的。

因此团队合作的研究任务可以分为：对多 agent 比赛环境中意外事件的规划；规划的合并和分解；团队规划、重规划和执行等。

1.3.3 对对手的建模

对 agent 的建模——即为其他 agent 的目标、知识、能力、规划或者情感建模和推理，是多 agent 交互的关键问题。RoboCup 对手建模是研究在动态多 agent 领域内对一组对手进行建模。RoboCup 中的建模问题可以分为三个部分：

(1) 追踪

队员可以观察对手动作来完成对单个对手的目标和意图进行实时、动态的在

线追踪并且通过这种追踪方法来预测对手可能的行为并做出适当、实时的反应。这样不仅可以用于防止被对方球员欺骗，还可以将得到的数据作为在线规划或在线学习的输入数据。

(2) 策略识别

球队的教练可以从场外观察比赛，获得对方球员采用的高层策略。与在线追踪相比，教练可以进行更高层次的抽象分析，同时也不必局限于实时反应，只要在比赛结束之前完成对对方策略的识别即可。教练的分析更详细，可以为球员改变跑位阵型或攻防策略提供必要的数据。这要求是一种在线的策略识别。

(3) 审阅

教练还可以在比赛后通过观察球队的表现来分析球队的优缺点，从而调整下次比赛时采用的策略或者阵型。这需要离线训练来获得。

1.4 RoboCup 仿真 2D 的研究现状

自 1997 年的第一届 RoboCup 以来，仿真 2D 经过了十几年的发展，已经取得了极大的进步，从开始见到“球”就追到现在有策略的攻防，从球员全场跑动到按固定角色在一定范围内进行决策，说明仿真 2D 的只能决策方面已经初步形成，正在朝着打败人类球队的策略方向发展。其间也出现了很多优秀的队伍，如日本的 Helios 队，中科大的蓝鹰队还有清华队，他们都是或者曾经是 RoboCup 比赛中的佼佼者，也为仿真 2D 的发展做出过重要贡献。但是受到软硬件条件和理论研究的限制，现在的 RoboCup 还处于初级阶段，还要走很长的一段路。接下来会介绍几支有代表性的球队。

1.4.1 UVA_Trilearn

荷兰阿姆斯特丹大学的 UVA_Trilearn 队是 RoboCup2003 仿真 2D 组的冠军，同时在第五届、第六届、第八届和第九届的 RoboCup 仿真 2D 组比赛中，都获得前十名的优秀成绩。阿姆斯特丹大学的 LAS 实验室早在 1996 年就已经开始了对多 Agent 系统的研究，并且参与了 RoboCup98 仿真 2D 组的比赛，获得第三的好成绩。UVA_Trilearn 队设计了一个简单有效的高层策略，后来他们的研究重点主要集中在球员的高层策略中多 agent 间的协作策略上。

UVA_Trilearn 球队主要有以下几个特点：

(1) agent 的结构采用的是混合式结构，采用柔性窗口的同步策略使 agent 中的三条线程相互配合工作（三条线程分别用于接收信息，决策和动作执行）。

(2) 提出了一种基于贝叶斯概率的射门技术，使射门技术大大提高

(3) 采用了微粒过滤的方法来进行球员和球的定位，并采用了信心优先模型进行动作的选择。

(4) 基于 CMUited99，UVA 改进的截球技术和一种传统路线选择方法。

(5) 将协作图应用于多 agent 中。

(6) 在决策中应用了 Max-Plus 算法。

(7) 在 UVA_Trilearn 的整个开发过程中，采用了软件工程的思想，保障了以后球队的发展。

1.4.2 FC Portugal

葡萄牙阿维罗和波尔图大学的 FC Portugal 球队是 RoboCup2000 仿真 2D 组的冠军，在第五届、第六届、第七届和第十届的 RoboCup 仿真组比赛中都获得前 5 名的优秀成绩，在第八届和第九届的 RoboCup 仿真组比赛中获得第 6 名和第 12 名。

FC Portugal 队的主要成员是 Luis Paulo Reis 和 Nuno Lau，他们采用 CMUited99 公开底层源代码并对踢球技术和世界模型进行了优化，通过时模拟人类球队的战术和策略，在策略层的设计上获得了很大的成功。之后，他们把精力主要集中在教练模型和机器人之间的通讯的研究上。

总的来说，FC Portugal 队有如下的特点：

(1) 提出了一套科学合理的策略体系并且易于实现

(2) 提出了 DPRE 动态角色变换体术和 SBSP 站位技术

(3) 采用了智能感知与通信技术

(4) 对踢球技术进行了优化

(5) 自己开发了一套集成系统调试工具

1.4.3 CMUnited99

美国卡耐基梅隆大学的 CMUnited 是 RoboCup98 和 RoboCup99 仿真 2D 组的冠军，在 RoboCup97 和 RoboCup2000 的仿真组比赛中，分别获得第四名和第三名。

CMUnited 队的创建人 Peter Stone 提出了机器人分层学习的概念，其不仅大力推进了 RoboCup 仿真组的发展，同时还为提出了当时赛事的研究方向。在 Peter Stone 毕业之后，CMUnited 队改用了 Tsinghuaeolus 的公开底层代码，其主要研究重点在策略层和教练模型上。

简单介绍一下 CMUnited99 的基本技术：

(1) 通过对世界模型的准确预测，促使机器人对未知环境进行建模

(2) 在机器人间协作方面采用了高级通信协议

(3) 定义了阵型，角色和角色的动态变换

(4) 采用了 SPAR 站位技术

(5) 对对手进行建模

1.4.4 Tsinghuaeolus

清华大学的 Tsinghuaeolus 队是第四届到第六届 RoboCup 仿真 2D 组的冠军，在 RoboCup03、04、05 分别获得仿真 2D 组第二、四和十三名的优秀成绩。在国内的 RoboCup 公开赛中，2000-2003 年获得 RoboCup 仿真 2D 组冠军，2004-2005 年获得 RoboCup 仿真 2D 组的第三名。

早在 1998 年，李实博士已经开始 RoboCup 的研究工作，并于 2000 年，与三名本科生共同组建了 Tsinghuaeolus 队。Tsinghuaeolus 搭建了自己的底层代码，同时设计了良好的策略。2005 年以后，Tsinghuaeolus 把主要研究精力转移到了仿真 3D 上。

Tsinghuaeolus 球队有如下的特点：

(1) 机器人结构采用混合式结构

(2) 采用了基于 BP 网络的截球技术，进行离线训练

(3) 改进了传球策略

(4) 提出了基于感知器网络的机器人站位

1.4.5 Wright Eagle

中国科学技术大学 WrightEagle(蓝鹰)队是 RoboCup2006 仿真组的冠军，在 RoboCup2001-2003，RoboCup2005-2008，都获得了前十的成绩，RoboCup2009-2012 都获得了前两名的好成绩。在国内，获得中国机器人大赛 RoboCup 仿真组 2005-2007 年的冠军，2008 年的亚军，2009-2012 年的冠军。Wright Eagle 的研究工作早在本世纪之前已经开始，由陈小平教授负责，WrightEagle 开发了自己的底层代码，并参照 CM(足球经理人游戏)进行策略层的设计。

1.5 安徽大学仿真 2D 机器人足球研究现状

安徽大学仿真 2D 机器人足球队 DreamWing2D 成立于 2006 年，并与当年在苏州举行的 RoboCup 中国公开赛上取得了前 24 名的成绩。接下来的 2007 年我们继续参加了济南的 RoboCup 中国公开赛并取得了前 10 名的成绩。2009 年，DreamWing2D 参加了安徽省第一届机器人项目比赛，取得了第二名的好成绩。随后 DreamWing2D 参加了再大连举办的一年一届的中国公开赛，拿到了第 12 名。2010 年，在这一年的中国公开赛上，DreamWing2D 闯进八强，取得了第 7 名的好成绩，此后并一直保持在八强之列。在 2011 年的中国公开赛上，DreamWing2D 实现了历史性的突破，进入三强，同年，DreamWing2D 取得了 2012 年 RoboCup 世界杯的参赛权，首次获得了国际认可。在 2012 年的中国公开赛上，DreamWing2D 继续保持八强。

虽然 DreamWing2D 不是很强大，但是一直在慢慢的成长之中，通过不断的比赛逐渐提升了 DreamWing2D 的实力。

1.6 本文的主要工作和内容组织

本文主要是对 RoboCup 仿真 2D 系统进行研究，其中研究重点是仿真 2D 中的智能学习算法，也是一种基于多 agent Q 学习的机器人学习算法。本文研究了 RoboCup 仿真 2D 中的学习算法等的相关理论以及应用还有存在的问题，同时提出了一种新的解决方法。本文组织结构安排如下：

第一章：绪论。主要介绍了 RoboCup 的背景和研究重点，以及现阶段国内外的研究现状，并且介绍了安徽大学仿真 2D 队伍的相关情况和取得的成绩。

第二章：RoboCup 仿真 2D 平台的介绍。该章主要介绍了仿真 2D 的标准比

赛平台结构，以及平台的特点和比赛的流程。

第三章：RoboCup 仿真 2D 中的基本模型。该章主要介绍了仿真 2D 标准平台模拟的各种模型，以及设计球员客户端时要注意的要点。

第四章：仿真 2D 的阵型与跑位。列出了仿真 2D 阵型表示的困难和 delaunay 三角的理论及可行性。同时介绍了不同的跑位方式。

第五章：RoboCup 中的学习算法。详细介绍了 RoboCup 仿真 2D 中可能用到的学习算法的理论基础以及基本的运用方法。

第六章：基于多 agent Q 学习的 RoboCup 局部配合策略。在以前的学习方法的局限性上，提出了新的学习策略，并且进行相关实验，证明了算法了可行性。

第七章：总结与展望。主要是对本文所做的工作和相关进行简要总结，同时对安徽大学的 DreamWing2D 队伍的未来发展做了展望。

第二章 RoboCup 仿真 2D 平台介绍

2.1 平台介绍

机器人足球仿真 2D 组的比赛规则与国际足球联合会颁发的比赛规则基本一致。其比赛的环境是一个由 RoboCup 组织委员会提供的称为 SoccerServer 的标准计算机仿真足球软件平台。该平台是一个真实足球的仿真系统。平台忽略了实体机器人和 3D 机器人所要研究的物体识别、硬件设计和机器人间的通信等限制性问题，这使得仿真 2D 的研究人员可以更多地把主义及集中的 agent 的协作和学习等高层技术上。

SoccerServer 采用的是客户/服务器(client/server)架构的服务模式：服务器端程序通过图形界面提供了一个虚拟比赛场地并且在后台模拟了球员和球等所有物体的移动；每个客户端程序相当于一个球员的大脑，控制场上该球员的移动。客户端和服务器端之间通过 UDP/IP 通信协议进行信息交互。每个客户端程序(即一个进程）只允许控制一名球员，客户端之间禁止私自通信，必须通过服务器根据规则来转发通信信息。一场比赛开始时，双方 11 个主力的球员程序连接到比赛平台进行比赛，每个队的目标就是将球踢进对方的球门同时阻止球进入自己的球门。

2.1.1 服务器端

仿真 2D 比赛平台的服务器端平台由 Soccer server 和 Soccer monitor 两个主要程序组成。其中，Soccer server 是一个模拟器，模拟比赛的实时环境还有球和球员的移动和碰撞检测，球员之间的通讯等，此外 Soccer server 还提供了自动裁判功能，根据比赛的规则进行一些简单的裁判。Soccer monitor 为 Soccer server 提供了一个 GUI 窗口，将其中的信息显示到一个虚拟场地上。Soccer server 与 Soccer monitor 的关系是一对多的关系，即一个 soccer server 可以提供给多个 soccer monitor 来显示。

Soccer monitor 显示的虚拟场地是一个 105×68 的二维场地，全场可以看成

是一个全局的直角坐标系，该坐标系以中场圆圆心为原点，朝向对方球门的方向为 X 轴正方形，x 轴正方向顺时针转动 90° 的方向为 y 轴正方向。如图 1 所示。每个球员用一个圆圈表示，圆圈分为两半，亮的一面表示球员身体的朝向，另一半通过颜色的深浅变化表示球员体力的变化，另外从圆心引出一条线段表示球员脖子的朝向，球员个人是以球员身体朝向为极轴的平面极坐标系，顺时针方向为正，有效角度为：-180° ～180° 。球用一个实心圆点表示。整个场地是一个虚拟的二维世界，也就是说球不能脱离地面飞起来运动。

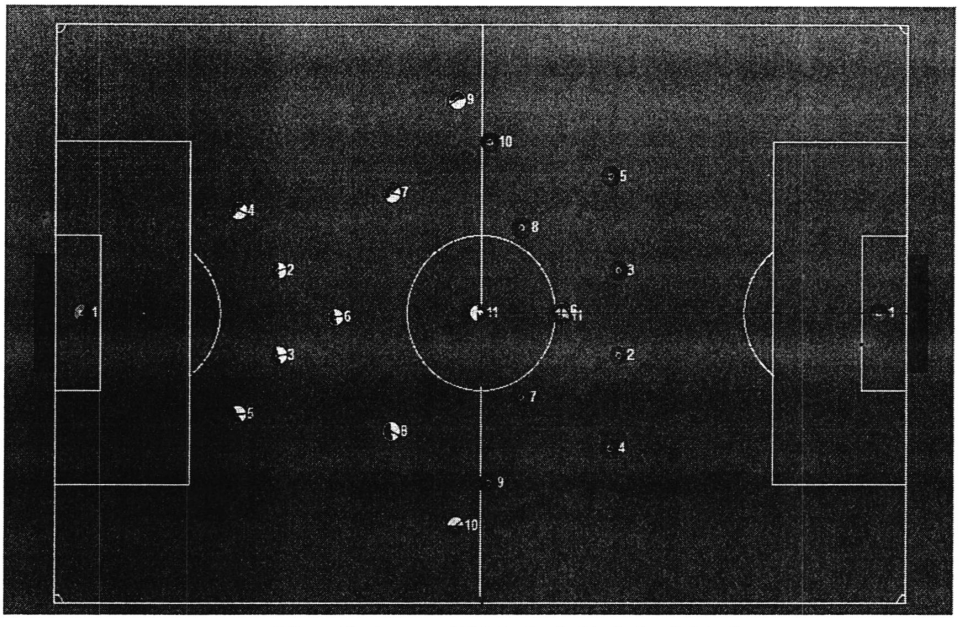


图 1 仿真 2D 球场和相应的直角坐标系

球员可以在比赛提供的平台上进行跑位，射门和传球等的动作。Soccer server 的组成分为：消息板模块、裁判模块和球场仿真模块。

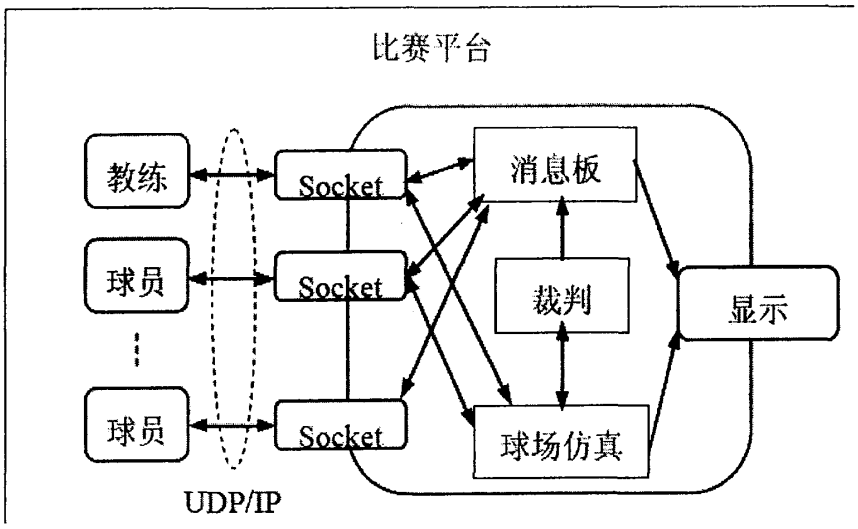


图 2 Soccer Server 的组成结构

球场仿真模块计算球场上物体的运动，并检测它们之间的碰撞。球场上的对象包括双方 22 名球员、球门、球、标记以及标志线等。其中标记和标志线只有绝对位置属性，而球和球员都具有速度、加速度、位置、各随机量等属性，球员更还有颈部方向、体力值、视觉距离等属性。球和球员的部分动态属性(如位置，速度等)在每个周期末根据动力学定律和平台量化规则等计算并更新一次。如果球员之间或者球员与球之间在同一时刻发生了重叠，则作为碰撞处理，此时双方速度都乘以-1。

比赛平台的裁判模块会根据比赛规则来控制比赛的进程。由于仿真 2D 的比赛环境非常复杂，并且动态地实时变化，具有不确定性，所以比赛需要一个裁判对每个球员的动作和球场状态进行判定，防止出现犯规。但是目前的裁判只能根据球场的状态来检测一些简单的形式，如界外球、进球、越位等。还有一些状态，比如陷入僵局，无人踢球等(无人踢球现已在自动裁判中实现判罚)状态比较难以判断，这时就还需要一个人为裁判。

消息板模块主要负责客户端之间的通信。每个客户端程序通过 UDP 协议的 socket 来连接 server。同样的，客户端程序可以通过 UDP 协议发送平台规定的命令来对球员进行控制，同时也可以发送命令获取球员的感知信息。

Soccer server 运行模式为离散化模式，即以周期为单位(缺省周期为 100ms)。在一个周期开始时，Soccer server 通过网络收集到所有球员程序的行为请求，并在每个周期末统一执行并且更新场上信息。这很好的体现了球员感知信息和行动

的异步性。如果一个球员在一个周期内发送了多于一条的独立行为请求(非独立的除外，如发送转头命令时可以同时执行另外一条独立命令)，server 将随机选择一个执行。

2.1.2 球员客户端

每个球员客户端(Soccer Client)程序都是一个独立的线程，其通过特定的 UDP 端口与 server 连接。建立连接以后，球员客户端程序与服务器端都是通过这个端口通信的，客户端可以通过该端口来控制球员的跑动或者其他行为，而服务器端可以通过这个端口将对应球员的感知信息发送出去。

一个球队最多可以连接 12 名队员(包括 11 名球员和一个教练)，这些球员想比赛平台发送请求执行的命令(如踢球、加速、射门等)，平台会分析处理这些请求，并根据规则更新场上的状态，同时向所有队员提供他们可以感知到的信息，例如球员可以看到的视觉信息以及自身的状态信息等。

球员之间的通信必须根据比赛平台规定的，按照 say 和 hear 命令协议执行，而且通信环境具有单信道、窄带宽和有噪声等特点。

为了尽可能模拟现实环境，比赛平台还加了很多限制，如 say 命令说话只有一定区域内的球员能听到，每个球员都有一定的视野范围，每次只能获得局部信息，每个球员都有自己的体力值，随跑动递减，每个周期可以根据规则自动恢复一些。这样就要求球员要注意调整跑动速度，合理分配体力。

除了球员客户端外，还可以有教练客户端连接到 Soccer server 上。教练是一种不直接参与踢球并且有特权的球员。它不仅可以获得比普通球员多的多的信息，还可以在对信息的准确分析后通过特殊的渠道指挥球员，通过调整阵型和策略来帮助他们更好地比赛。教练分为离线(offline)教练和在线(online)教练两种。离线教练也是一种训练器，它不能在正式比赛时使用，但是拥有控制比赛和调整策略的能力。它主要用于帮助球员进行技能训练和团队合作等策略的调试。使用训练器可以改变比赛状态，给场上球员广播各种信息、移动球员和球到任何位置，并可以赋给他们一定的速度和方向等，这些信息对球员程序的自学习和自动管理比赛起到重要作用；在线教练也就是在正式比赛中的教练，它可以在比赛过程中活动全局无噪声的信息，通过对获得信息的分析来进行高层次的策略规划，再通

过平台提供的通信方式根据规则的给场上相应球员提供更多的信息和建议，在线教练需要在比赛结束之前作出决策。

2.2 平台结构特点

这套仿真比赛平台提供了一个很好的全分布的、多 agent 实时环境。其具体特点如下：

⑴具有分布式多 agent 团队合作和团队对抗，所有客户端程序分别控制场上相应的球员或者在线教练，通过自主决策和学习，分布运行，来完成队友间的合作和对手间对抗。

⑵比赛环境具有实时性、动态性、不确定性等特性，比赛是按照以 100ms 为周期的方式进行，所有的球员都必须按照这个周期运行。这就意味着所有球员的决策都必须实时完成。由于多 agent 的存在，导致球场环境在动态的转变，无法预知。

⑶感知和行为异步，比赛平台规定球员的感知周期为 75ms-300ms，而球员的发送行为的周期是 100ms，这说明感知信息的行为和动作选择的行为无法保证同步，所以只靠传统人工智能方法来通过使用感知信息激发动作的方式是不可行的。

⑷球员体力速度等的限制，球场上所有球员的能力都是模拟真实球员来进行相关限制的，如对球员体力值的限制和对速度的限制等。

⑸限制视觉信息，通过对真实球员的视角和视距限制的模拟，每个球员获得的视觉信息都是局部视觉信息，即球员在任何时刻都只能获得一部分球场上的信息，也是相对自己的相对信息而不是绝对信息(即根据球员自己的方向和极坐标运算而来)。这样球员就很难正确地分析场上形势，进而产生相关决策。

⑹通讯受限。平台提供的球员之间的通信环境是一条窄带宽的单信道，每队球员共用一条信道，一个周期内每个球员只能收到一条队友信息，收到的信息也不能确保正确，因此很难将获得通信信息作为团队合作数据来直接应用。

⑺多噪声源，为了模拟真实的比赛，仿真平台对所有球员感知信息和动作信息都进行了量化并人为加入噪声。这使得球员无法获得精确信息来进行决策。

⑻不可靠的连接，平台网络连接使用 UDP/IP，不确保所有信息都正确和及

时，在网络繁忙时一些信息可能会丢失，球员程序必须要能够适应这样的环境。

2.3 比赛流程

整场比赛的过程如下：

(1) 首先比赛人为裁判启动比赛的标准平台(Soccer server 和 Soccer monitor)

(2) 双方领队猜先，决定哪方先上场 (先上场的一方先开球)

(3) 双方球员上场，即双方各启动 11 个球员程序 (如果有在线教练还可以再启动一个在线教练程序)与比赛平台连接，连接上后通过发送 init 命令进行初始化，实现球员上场）

(4)当双方全部球员都上场并且准备好后，比赛裁判按下 Monitor 上的 kickoff 按钮，向平台服务器发送比赛开始命令，上半场比赛开始

(5)上半场比赛默认为 5 分钟。上半场比赛结束时，比赛自动暂定比赛，各队球员程序如需调整可以与 Server 断开连接

(6)中场休息 5 分钟，参赛者可以在此期间修改各自的程序

(7)在下半场比赛开始之前，与 Server 断开连接的所有球员程序需要使用 reconnect 命令与 Server 重新进行连接

(8)当全部球员准备就绪时，裁判按下 kickoff 按钮，开始下半场比赛，下半场由另一方开球

(9)下半场比赛同样为 5 分钟，下半场结束时，比赛平台自动停止比赛

(10)如果比赛结果为平局，则开始加时赛。加时赛时，当一方进球后，比赛即结束(金球法)，也可以通过更改配置文件来更改加时赛方式，加时赛同样分上下半场，如果还是平局，就通过点球决胜

第三章 RoboCup 仿真 2D 中的基本模型

3.1 动作模型

Agent 所能执行的命令有两类：基本命令和并发命令，在每个仿真周期内只能执行唯一一个基本命令，而并发命令可以与任何基本命令同时执行，例如在传球的同时可以转动脖子来寻找传球点。基本命令有加速（dash）、踢球（kick）、转身(turn)、扑球(catch)、瞬移(move)，并发命令有喊话(say)、转脖子(turn_neck)、指向某点（pointto）、改变视觉方式（change_view）、获取自身感知信息（sense_body）、关注某人（attentto）和获取得分（score）。

3.1.1 踢球模型

当一个球员想要踢球时，它必须向 server 发送 kick 命令，其包含：踢球的力量 Power 和踢球的角度 Direction。对球加速的大小由 Power 决定，取值介于 minpower 和 maxpower 之间(即[-100,100])；Direction 是相对于球员身体方向的相对角度，取值介于 minmoment 和 maxmoment 之间(即[-180,180])。 当 server 接收 kick 命令后会对当前状态进行判断，如果该球员不处在越位位置且在其控球范围内有球，那么就会执行 kick 命令。球员的最大控球范围是以自身为原点，player_size+ball_size+ kickable_margine 的圆(默认值为 0.3+0.085+0.7)，即当当球员和球边界之间的距离小于 kickable_margine 时，球员可以执行 kick 命令。

执行 kick 命令时，球获得的有效力量并不一定等于 kick 命令中使用的 power 值，而是受到球员和球的相对位置的影响。作用在球上的实际邮箱力量可以通过公式得到：

$$act\_pow = Power \times (1 - 0.25 \bullet \frac{dir\_diff}{180} - 0.25 \bullet \frac{dist\_diff}{kickable\_margine})$$

其中，dir_diff 是球员身体方向和球之间的绝对角度，dist_diff 是球员和球之间的相对距离。从公式中可以看书，球与球员之间的角度越大，实际有效的力量越小。

踢球的有效力量将被用来计算球所获得的加速度：

$$(a_x^t, a_y^t) = act\_pow \times kick\_power\_rate \times (\cos(\theta^t), \sin(\theta^t)) + (\tilde{k}_1, \tilde{k}_2)$$

其中，kick_power_rate 是 server 的参数，默认取 0.027；θ^t 是 t 周期中秋获得加速度的全局方向；$(\tilde{k}_1, \tilde{k}_2)$是足球运动过程中的噪声，$\tilde{k}_i$是[-$k_{max}$，$k_{max}$]之间的随机数，目前取 0.

3.1.2 加速模型

加速（dash）命令会让自身按照自己的身体方向进行加速，其只有一个参数：加速力量 power。其中 power 值必须介于[minpower, maxpower]之间(即[-100,100])，负数表示向后加速。与 kick 模型累死，球员加速的实际有效力量不一定等于 dash 的参数，而是受到球员体力状态的影响，如果 power 超过了球员的当前剩余体力值，那么 power 将会被降低后再发送给 server。球员获得的加速度公式为：

$$(a_x^t, a_y^t) = power \bullet effort \bullet dash\_power\_rate \bullet (\cos(\theta^t), \sin(\theta^t))$$

其中 effort 为球员体力模型中的参数，最大为 1；dash_power_rate 默认值为 0.006;接下来的参数为身体的方向。

球员执行一次 dash 命令只会在一个周期加速，在下一次执行该命令前，加速度都是 0。执行过后加速度就会转为该球员的速度，并且随着周期衰减，所以球员必须通过不断的使用 dash 命令来保持持续的速度。

3.1.3 转身模型

转身(turn)是用来调整球员的身体方向，该命令只有一个参数：角度 direction，表示该球员要转的角度，其用度数为单位，取值须介于 minmoment 和 maxmoment 之间(即[-180,180]).

如果球员是静止的，那么它实际转过的角度就是 moment，但是，如果球员在运动中，其在惯性的作用下转身会较为困难，，转身的角度也不是简单的 moment，此时实际转身的角度为：

$$act\_ang = \frac{(1.0 + \tilde{r}) \bullet momet}{1.0 + inertia\_moment \bullet player\_speed}$$

其中，$\tilde{r}$ 是在[-player_rand, player_rand]之间的一个随机数(player_rand=0.1); inertia_moment 是 server 中表示球员惯性大小的参数，默认值为 5.0；player_speed 表示球员当前的速度。

3.1.4 扑球模型

扑球（catch）只可以由守门员向 server 发送，该命令只有角度参数，用来表示守门员扑球的角度，以度数为单位，取值为[-180,180]之间。Catch 命令必须在满足一定的条件时才会成功：首先，发送命令的是守门员；其次，当前比赛模式为 play_on，最后，球在守门员的扑球范围并且再本队禁区内。否则扑球失败。

守门员的可扑球范围是指在它扑球方向上长为 catchable_area_l 和宽为 catchable_area_w 的矩形区域(该矩形区域默认长宽值分别为 2.0 和 1.0)，图显示了守门员使用-45° 角扑球的范围。

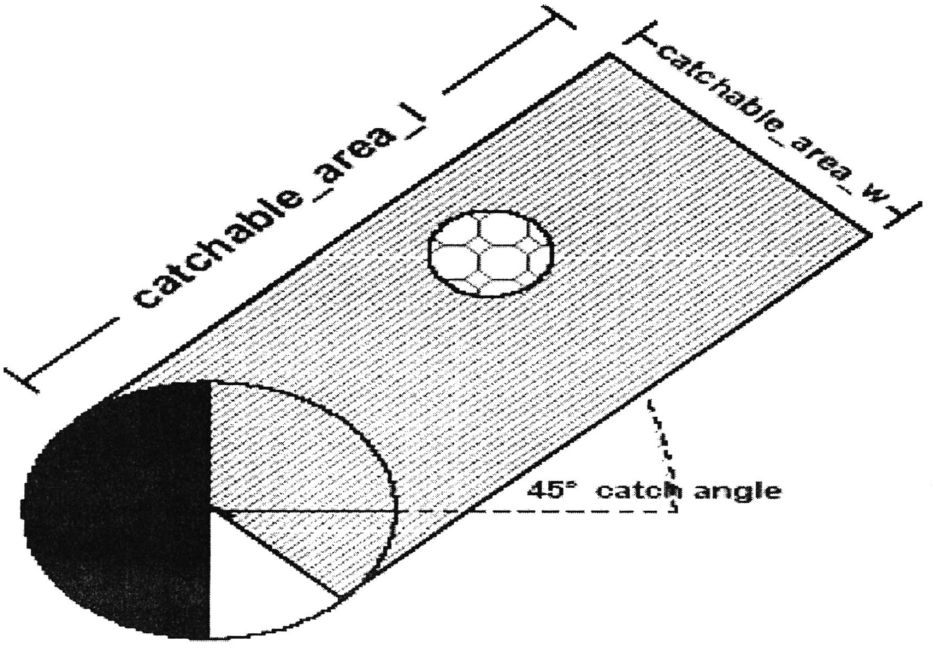


图 3 守门员的扑球模型

守门员执行扑球命令后，只有在 5 周期后才能执行下一次的扑球动作。

3.1.5 瞬移模型

为了节省球员体力，在一定条件下，球员可以使用瞬移（move）命令移动到

球场特定位置，该命令有两个参数：x 和 y，表示球场的全局坐标。使用该命令的条件是：在上下半场开球之前快速形成初始阵型；任意一方进球后，形成开球阵型；守门员在扑球动作成功后，可以有两次 move 的机会来避开进攻球员的进攻。

3.1.7 铲球模型

铲球（tackle）命令可以有助于球员提到当前 kick 命令踢不到的球，其会让球员以一定概率给球在某个方向上一个加速度，tackle 的参数有两个：相对方向 angle 和一个参数 foul。Angle 可以决定 tackle 的的方向但是不能更改力量。铲球的概率与球员和球之间的位置有关，只有当球在球员正前方长宽分别为 5.0 和 2.0 的矩形区域内才能获得大于 0 的概率，执行 tackle 命令后，球员在十个周期后才能改变位置。Foul 参数可以决定是否会引起犯规，当 foul 为 true 时，球员铲球后可以立即移动，但是可能会将对方球员铲倒，会以 50%的概率获得黄牌，两张黄牌则会被罚下场。

3.1.6 其他模型

(1)说话 （say）

球员可以用 say 命令向周围的球员进行广播，在 audio_cut_dist(默认为 50.0)范围内的球员都可以听见广播的消息，消息长度限制在 say_msg_size(缺省为 512)内.球员通过 say 命令传递的消息没有延迟，会立刻传到可听范围内的双方所有球员。但是，球员听觉能力有限，每个球员两个周期只能听到一个队友的一条消息。

(2)转头（turn_neck）

Turn_neck 有一个参数：角度 moment，其没有噪声影响，也没有自身运动速度的影响。

(3) 改变视觉 （change_view）

用于改变球员的视觉模式和视觉质量，其由两个参数：视觉模式和视觉质量，视觉模式取值为：正常，宽模式，窄模式之一；视觉质量为高或者低质量。

(4) 注意（pointto 和 attentionto）

用于从同班出获得听觉信息或者位置信息。

(5)得分 （score）

用于获得当前两球队之间的比分情况。

3.2 运动模型

在 soccer server 中，物体的运动通过一个简单的逐步计算仿真。每周期，移动的物体通过下面的公式计算：

⑴速度计算公式

$$(u_x^{t+1},u_y^{t+1})=(v_x^t,v_y^t)+(a_x^t,a_y^t)+(\tilde{r}_1,\tilde{r}_2)+(w_1,w_2)$$

(u_x^{t+1},u_y^{t+1})为 t+1 周期开始时的速度；(v_x^t,v_y^t)为 t 周期末的速度；(a_x^t,a_y^t)为 t 周期的加速度，球员的加速命令可以使球员产生加速度，而球员的踢球或铲球动作可以使球产生加速度，最终的加速度是各个球员施加到球上加速度的矢量和；$(\tilde{r}_1,\tilde{r}_2)$表示环境中的噪声，其包括两个随机数，每个随机数的取值范围为[$-r_{max}$，r_{max}]，r_{max} 由物体速度和加速度决定，通过 $r_{max}=Rand\times\|(v_x^t,v_y^t)+(a_x^t,a_y^t)\|$计算得到，其中 Rand 是 server 的参数，用来表示物体运动中随机噪声的影响，球员 Rand 取值为 player_rand（缺省为 0.1），对于球则取值 ball_rand（缺省为 0.05）；(w_1,w_2)表示环境中的风力，目前缺省为 0.

⑵物体位置变化公式为：

$$(p_x^{t+1},p_y^{t+1})=(p_x^t,p_y^t)+(u_x^{t+1},u_y^{t+1})$$

(p_x^{t+1},p_y^{t+1})为 t+1 周期开始时物体的位置，(p_x^t,p_y^t)为 t 周期时物体的位置

⑶每周期物体速度变化公式为：

$$(v_x^{t+1},v_y^{t+1})=Decay\times(u_x^{t+1},u_y^{t+1})$$

(v_x^{t+1},v_y^{t+1})为 t+1 周期末物体的速度，Decay 表示衰减参数，球员的 Decay 等于 player_decay(缺省为 0.4)，而球的是 ball_decay（缺省为 0.94）

⑷每周期加速度都会置 0

$$(a_x^{t+1},a_y^{t+1})=(0,0)$$

3.3 感知模型

RoboCup 每个 agent 都有三种不同的感知信息：听觉信息、视觉信息和身体状态。听觉感知用来检测教练、其他球员和裁判发来的信息；视觉信息，用来感知自身与当前可视范围内对象的方向和距离(相对距离和相对方向，方向用自身方向的极坐标表示)；身体状态感知用来检测自身当前的物理状态，如体力 stamina，速度和方向等。这三种感知联合给 agent 提供了一个较好的环境图景。下面将对这三种类型的感知信息分别介绍：

3.3.1 视觉模型

每个球员都可以通过视觉传感器获得当前视野范围内所有物体的相对方向、相对距离的视觉信息。这样球员就可以根据所有物体中的场地标记的相对方位，来进行自身定位，还可以通过这些数据计算出一定距离范围内对象的距离和方向的相对变化。也就是说球员可以根据这些相对信息获得一些物体或自身的绝对信息。这些视觉信息每隔一定的周期(周期数与视觉质量等因素有关)就发送给场上的球员。但是球员一旦从一个位置转移到另一个位置，上一次获得的相对信息不能再用了。比赛平台通过在场地周围放置了各种地界地标(旗帜标，边线和球门标等)来 让球员取得场上物体的全局信息。

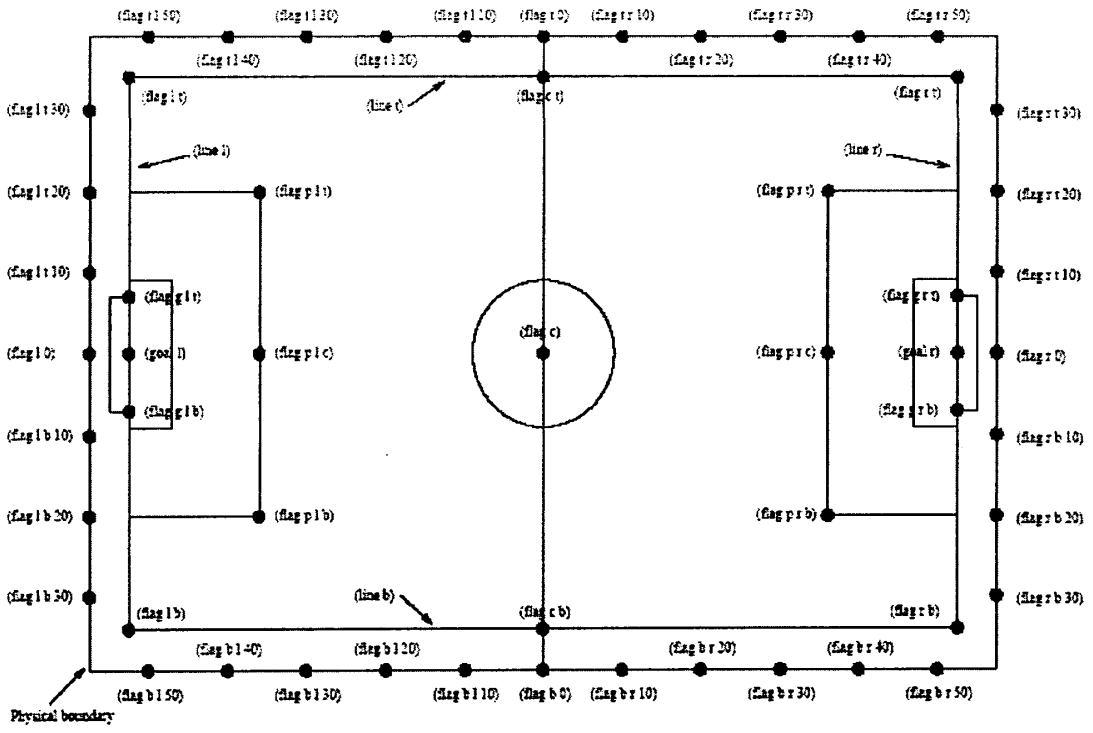


图 4 球场上的表示位置和名称

球员结合这些地标的全局位置以及视觉信息中与这些地标的相对位置可以计算出自身的全局位置以及其他球员或者球的全局位置。仿真 2D 系统中每个球员都有身体方向和脸方向这两个概念。每个球员根据身体方向来决定自身的跑动方向，即每个球员只能朝自己身体朝向给自己加速度或者给球一个该方向的加速度。每个球员的视觉范围由其脸的方向决定，视野范围是一个以脸的朝向为中心，正负二分之一视野角度的扇形。

视觉模式分为视野范围、视觉质量和信息间隔三个部分。根据视野范围的宽度分为正常模式，宽模式和窄模式，其对应的视野范围为：[-45°，45°]，[-90°，90°]，[-22.5°，22.5°]，另外视觉质量分为高质量和低质量，当球员请求的是高质量视觉信息时，接收到的是详细的目标位置信息，而请求的是低质量的视觉信息时，Server 为球员发送的是简化的目标信息，只包含目标的相对方向。信息时间间隔是随着球员请求视觉信息的视野范围和视觉质量的变化而变化的，具体公式为：信息间隔=sense_step×视野范围×视觉质量。其中视野范围根据范围大小取值为 0.5, 1, 2，视觉质量根据质量高低取值为 0.5, 1，sense_step=150.

3.4 体力模型

仿真 2D 平台对体力模型进行了限制，这样球员就不会不受限制地以最大速度奔跑。具体限制为：每个球员都有一定有限的体力值(现在默认为 8000)，体力值会在加速时减少，同时每个周期会根据相关公式恢复一些。

体力模型包含三个部分：stamina, effort, recovery。体力值 stamina 代表球员当前剩余的体力，介于[0,8000]之间，执行 dash 命令会消耗体力，每半场开始时，初始为 8000.当球员向前加速 power 时，体力减少 power 大小，如果向后加速，体力值减少 2 倍的 power 值。Effort 代表球员加速的效率，介于[0.6, 1.0]之间；recovery 表示球员体力恢复的速率，介于[0.5，1.0]之间。三者按照一定规律变化。当 effort 减小时，加速的效率变低，当 recovery 低于一定值使，球员体力恢复速度变慢。

3.5 裁判模型

Server 中提供一个自动裁判来做一些基本的规则限制，以确保动态的比赛的顺利进行。

⑴中场开球(kick_off)

在半场或者进球后，比赛状态会进入到中场开球状态，此时所有球员都必须在各自半场。为了确保这点，进球后，裁判将比赛暂停 5s，球员可以通过 move 指令将自己瞬移到己方办成的某个位置。如果进入开球状态之后仍然有在对方半场的球员，裁判则将该球员移到所属半场的一个随机位置，这样的移动会减少该球员的体力值。

⑵进球(goal)

当某队进球得分后，裁判首先通过广播向所有球员宣布进球，然后更新场上比分并将球移动到球场中心，同时将 play_mode 置为 kick_off_left 或者 kick_off_right。最后暂停比赛 5 秒钟，等到球员回到各自半场。

⑶守门员发球(goalie free kick)

当守门员扑球成功后，就可以直接发球，此时允许守门员发球前可先使用两次 move 指令避开进攻球员，若超过两次，自动裁判使指令无效。

⑷出界(out of field)

当球滚到界外，裁判将根据出界位置的不同将球移动到一个合适的罚球位置，相应的将 play_mode 置为 kick_in(界外球)、goal_kick(球门球)或者 corner_kick(角球)。

⑸越位(offside)

当一个球员满足下面所有情况时会被判越位：

① 在对方半场

② 至少比两个防守球员更靠近对方球门

③ 比球更靠近对方球门

④ 距离球小于 2.5

⑹回传(back pass)

守门员不允许扑队友传回来的球，一旦发生，裁判将判回传，让对方发任意球。若发生在禁区内时，对方会在守门员扑球位置的一个禁区角发。

⑺发球违例(free kick faults)

当球员发角球、任意球，或者守门员发球时，发球者不允许把球传给自己。如果发球者在踢出球后紧接着又踢球，裁判将判 free_kick_fault(发球违例)，对手获得任意球。很多时候，球员为了将球踢到期望的速度不得不连续踢很多脚球，此时裁判不会判罚发球违例，发球违例只是在球员踢球后移动过后(使用了 dash 命令)紧接着又 kick 时才会出现。

⑻扑球违例(catch fault)

当守门员在己方禁区外扑球时，裁判会判扑球违例(catch_fault)。由对方在扑球位置发间接任意球。

⑼球员清除(player clearance)

当 play_mode 为 kick_off, free_kick 或者 corner_kick 等状态时，裁判会清除以球为中心，缺省半径为 9.15 的一个圆内的所有对方球员，移除的球员会被随机的放在圆周的某个位置上。当 play_mode 为 offside(越位)时，所有处于越位位置和距离球 9.15 圆内的进攻球员都会被移回到非越位位置。当 play_mode 是 goal_kick 时，所有的进攻球员都会被移出禁区，并且再踢球门球时，在球出禁区前所有进攻球员不能再进禁区，此时 play_mode 才会改变。

⑽比赛状态控制(play_mode control)

当 play_mode 为 kick_off， kick_in ，free_kick 或者 corner_kick 状态之一时，裁判在球被踢动后会立即将 play_mode 改为 play_on。

(11)半场和终场

裁判在半场结束时会暂停比赛。缺省的半场时间是 3000 个周期，如果正常比赛打平就会开始加时赛。加时赛采用“金球制”，即最先进球的一方获胜。加时赛也分为上下半场共 3000 周期，如果加时赛没有进球则进入点球大战。

第四章 仿真 2D 的阵型与跑位

在人类足球中，为了要有良好的攻守战术，全队人员在球场上要分角色，按照位置进行排列，不同位置有不同的分工，并以此作为比赛阵型来进行比赛。RoboCup 比赛在后来也提出了阵型的概念，球员通过跑位来保持阵型，并尽可能保持攻守平衡。在 RoboCup 仿真比赛中，常见的阵型包括 442、433、424、352 等。为了让球队在整场比赛中都保持阵型，从而达到攻守平衡，需要用一种方式让球员在根据不同球场形式进行不同的跑位。如果只是通过手工编码，球场环境复杂，所以很难考虑完全，工作量也非常大。通常球场形式是根据球所在的位置和控球方来决定的，我们可以通过这两点来设计阵型位置。首先，我们就要将球场划分，根据球所在的位置不同来确定每个球员的位置。

4.1 Delaunay 三角化

对于球场划分以后，要确定划分的方式是唯一的，可以使用 delaunay 三角剖分法将球场剖分。Delaunay 三角剖分即是将球场三角化以后不存在任意四点共圆的特殊三角化方法。

Delaunay 三角化后的 delaunay 三角化网具有以下特征：

① Delaunay 三角网是唯一的，即通过算法构建，在不同场合可以构建出相同的 Delaunay 三角网；

② Delaunay 三角网的外边界是一个凸多边形；

③ 任意三角形的外接圆都不包含 Delaunay 三角网的第四个点，这也是 Delaunay 三角网最大的特点，同时如果一个三角网满足此条件，那么其肯定就是 Delaunay 三角网；

④ Delaunay 三角网是最接近规则化三角网的，也就是说 Delaunay 三角网的所有角度相加之和是所有三角化方法中的最大的。

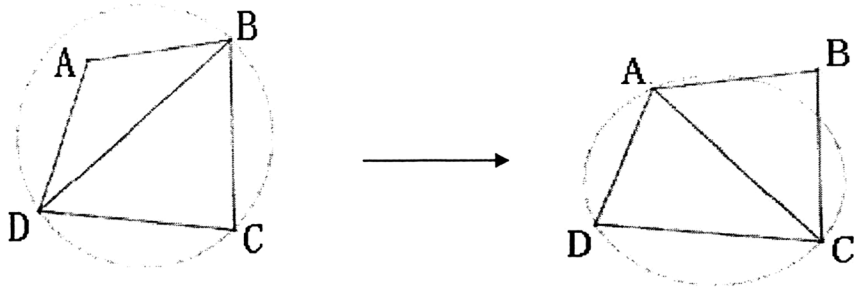


图 5 Delaunay 三角示意图

4.2 Delaunay 三角化算法

经典的 Delaunay 三角化算法主要有两类。

4.2.1 Bowyer/Watson 算法

Bowyer/Watson 算法又称为 Delaunay 空洞算法或加点法。从第一个三角开始，一步一步的加点，并且保证每一步加点构建后是局部最优且为 delaunay 三角网。构造步骤如下：

①首先构造一个大三角形，覆盖所有的位置并且包含所有的离散点，将该三角形放入三角形链表中。

②将离散的点依次插入到三角形中，同时在三角形链表中找出其外接圆包含插入点的三角形(称为该点的影响三角形)。将影响三角形的公共边删除，并将改点与影响三角形的三个顶点(可能为两个)连接起来，这时就完成了一个点的插入。

③根据算法给出的优化规则对新形成的三角形进行局部优化。将形成的三角形放入 Delaunay 三角形链表，并删除原链表中的节点。

④循环执行第 2 步，直到所有离散点插入完毕。

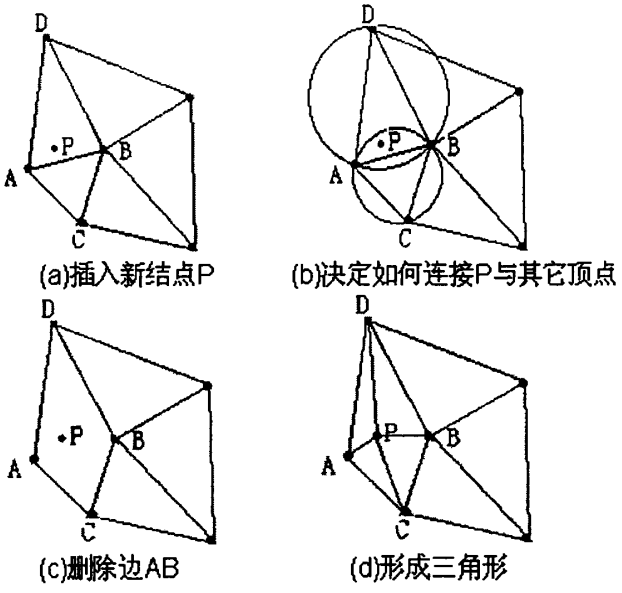


图 6 Bowyer/Watson 算法示意图

该方法利用了 Delaunay 空洞的性质。Bowyer/Watson 算法的有点是与 E^d 空间的位数 d 无关，并且算法在实现上比局部变换算法简单。

4.2.2 局部变换法

局部变换法又称为换边/换面法。利用局部变换法实现增量式点集的 Delaunay 三角化时，首先定位新加入点的所在三角形，然后在网格中加入三个新的连接该三角形顶点与新顶点的边(若该新点位于某条边上，则该边被删除，加入 4 条连接该新点的边)，最后在通过换边方法对该新点的局部区域内的边进行检测和变换，重新维护网格的 Delaunay 性质。

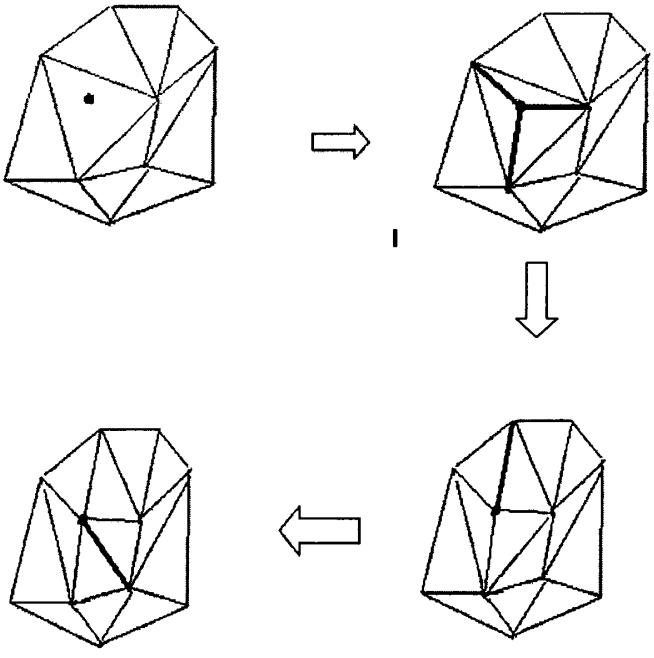


图 7 局部变换法的示意图

4.3 Delaunay 三角化后的阵型

根据上面介绍的算法，我们可以将球场进行 Delaunay 三角化，再根据控球方分为防守阵型(对方控球)、进攻阵型(我方控球)和任意球阵型(也分为对方发任意球阵型和我方发任意球阵型) ，不同阵型三角化的不同。每个三角形的顶点作为球所在的位置，根据球所在的位置球员再进行跑位，球到达下一个顶点时，球员则相应地跑到下一个顶点对应的位置。由于球员有体力限制和速度限制，所以在定义每个顶点球员所在位置时，两个顶点间相差距离不能太大，否则球员不能跑到相应位置或者体力消耗过快。

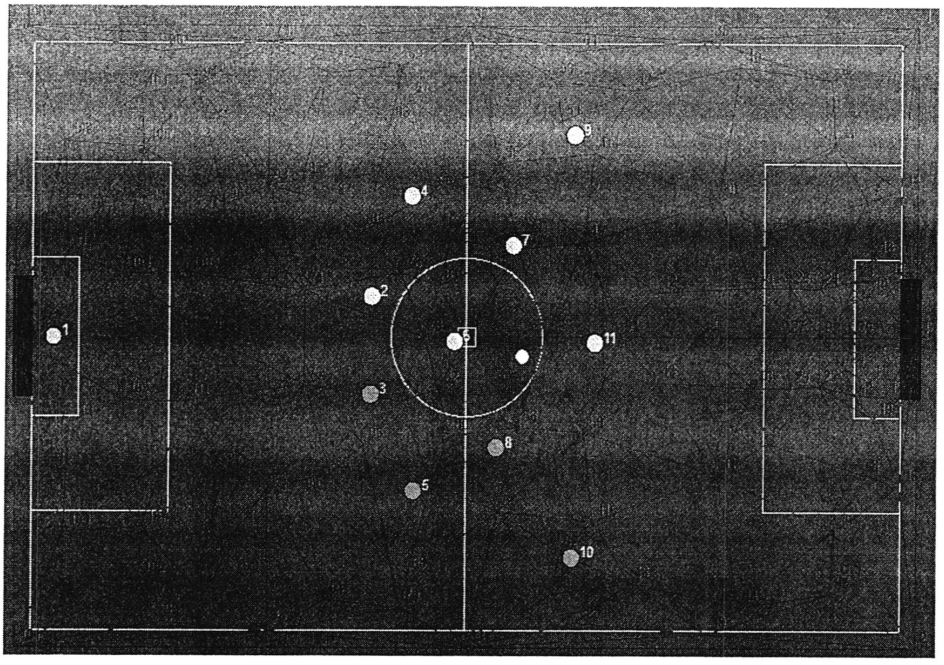


图 8 球场三角化后阵型表示

在上述的阵型中，球员根据角色和球所在的位置，得到自己的基本跑位点进行跑位，还可以在当前球场形式和球员策略等方式下，在基本跑位点的基础上计算出其他跑位点来更好的进行球员间的合作。

第五章 RoboCup 中的学习算法

5.1 马尔可夫决策过程

马尔可夫决策过程（Markov Decision Process, MDP）是指智能体周期性的或者连续的观察具有马尔可夫性的随机动态系统，然后做出相应的动作决策。即通过对环境的描述，agent 可以在每个离散的周期中观察到一些状态信息，同时从动作集中选出一个动作作为相应的动作决策，动作的产生影响了未来的系统状态，系统状态的改变随机的，并且状态转移的概率具有马尔可夫性。Agent 根据新观察到的状态再做出新的动作决策，依次反复进行直到 agent 达到某种目标。马尔可夫性是指一个环境状态的变化规律与历史无关的性质。马尔可夫性也可以叙述为状态转移概率的无关性。状态转移概率具有马尔可夫性即为马尔可夫过程。

马尔可夫过程可以用一个四元组<S,A,T,R>表示，

其中，S 是状态空间；

A 是行动空间，表明每个状态下智能体可以执行的动作，状态 s 下可执行的动作记为 A(s);

T: S×A×A→[0, 1]是状态转换模型，T(s, a, s’)表示在状态 s 下执行动作 a 后到达状态 s’的概率，满足 $\sum_{s'} T(s,a,s')=1$；

R: S×A×A→R 是对动作 A 的奖赏值，其中，R(s, a, s’)表示根据对环境的描述，agent 在状态 s 下执行动作 a 到达状态 s’的评价。

从转换模型和即时回报函数可以看到，它们只与当前状态和动作有关，不需要历史信息。

5.2 动态规划

动态规划(dynamic programming)作为运筹学的一个分支，于 20 世纪 50 年代初，由美国数学家 R.E.Bellman 等人提出，是求解决策过程最优化的方法。

动态规划主要应用于经济管理、工程技术、生成调度和最优控制等方面。其

通过组合子问题的解而解决整个问题的方法。动态规划的子问题不是独立的情况，也就是各子问题包含公共的子子问题。为了避免在计算各个子问题时会出现重复计算的情况，动态规划方法只对每个子子问题进行一次求解，并将计算结构存在一张表中，供下次计算是查找。

动态规划通常应用于最优化问题，此类问题可能有很多种可行解。而动态规划的目的就是在众多可行解中找到最优的解，即具有最大值或者最小值的解。该算法的设计可以分为四个步骤：

(1) 描述最优解的结构

(2) 递归定义最优解的值

(3) 按自底向上的方式计算最优解的值

(4) 由计算出的结果构造一个最优解

5.3 人工神经网络

人工神经网络是近年来发展起来了一门学科，通过对人脑的研究和模拟，来实现对某些方面的学习和判断。其模拟了大脑的神经元细胞和神经元之间的连接以及连接上的信号，通过相应的训练，使得神经元和连接分别有合适的阈值和权值，再根据相应的运算规则来进行运算和判断。

5.3.1 单层神经网络

单层神经网络即只有一个神经元的神经网络，其可以接受一组输入信号，每个输入对应一个权值，所有输入的加权加和后通过激活函数的计算值决定该神经元的激活状态。

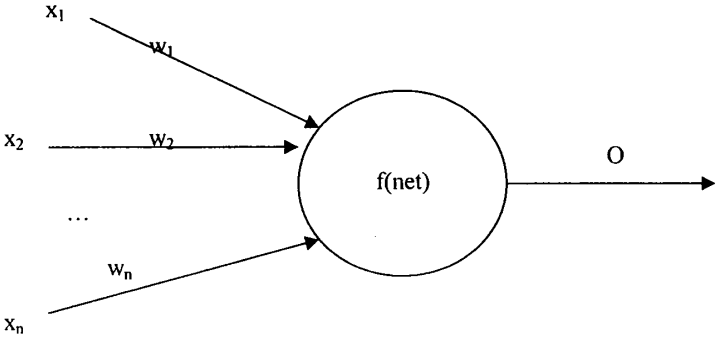


图 9 单层神经网络示意图

典型的四种激活函数有：线性激活函数、非线性斜面激活函数、阶跃激活函数、s 型激活函数。

5.3.2 多层神经网络

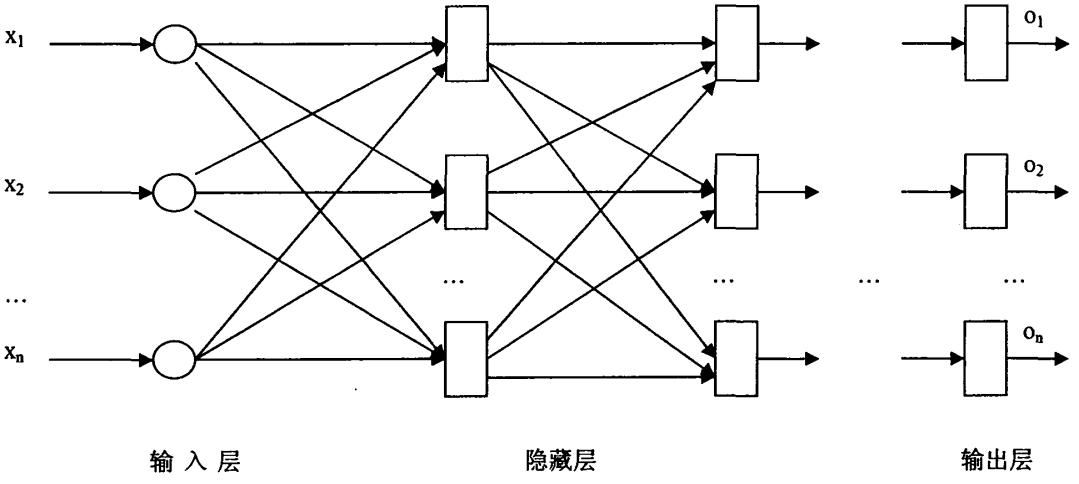


图 10 多层前馈网

上图是一个典型的多级前馈网，又叫做非循环多级网络，分为输入层、中间层(即隐藏层)和输出层。在这种网络中，采用的是非线性激活函数，用以解决人工神经网络所面临的线性不可分问题。增加网络的层数在于提高网络的计算能力。**X** 是其输入向量，$\mathbf{W}^{(1)}$、$\mathbf{W}^{(2)}$、…、$\mathbf{W}^{(n)}$是各级联接矩阵，$\mathbf{NET}_1$、$\mathbf{NET}_2$、…、$\mathbf{NET}_1$ 分别是各级的网络输入向量，$\mathbf{F}_1$、$\mathbf{F}_2$、…、$\mathbf{F}_n$为各级神经元的激活函数，如果它们是线性的，我们可以用下面公式计算(如果为其他类型的，就必须通过其他公式计算)：

$$\mathbf{F}_i(\mathbf{NET}_i) = \mathbf{K}_i\mathbf{NET}_i + \mathbf{A}_i \quad 1 \leqq i \leqq n$$

其中，$\mathbf{F}_i$，$\mathbf{A}_i$是常数向量。

5.3.3 反传神经网络

反传神经网络及其学习算法由 Rumelhart 等于 1986 年提出，很快就成功应用于多个领域。其学习算法由很强的数学基础，有成熟的权值修改方法。

反传神经网络结构符合多层神经网络模型，有一层或多层的隐藏层神经元，

同时要求相邻两层神经元间才有连接，同层神经元间和不相邻的神经元之间没有连接，每个神经元的激活函数是处处可导的，一般用 s 型激活函数。

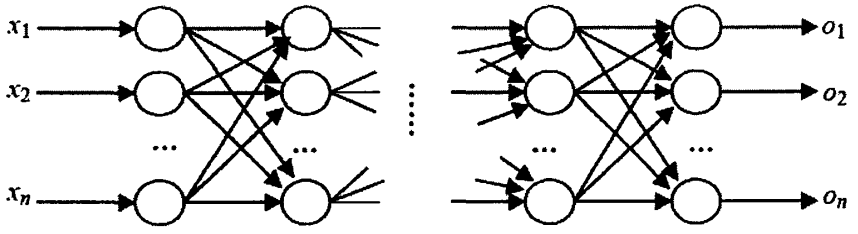


图 11 BP 网络结构示意图

反传神经网络的学习过程主要分为两个阶段：先从样本集中取出一个样本 (X_p,Y_p)，将 X_p 输入网络，信息从输入层逐级的变换，最终得到网络的实际输出 Op；然后根据实际输出与理想输出 Y_p 的误差，逐级地从输出层到输入层方向有梯队下降的方法调整权值。因为调整权值是从后向前反向进行，且根据反向传播过来的误差来调节权值，所以才称为反传网络。

梯队下降的方法要求权值的改变满足下面公式：

$$\Delta_p w_{ij} \propto -\frac{\partial E_p}{\partial w_{ij}}$$

其中，w_{ij} 是从节点 i 到节点 j 的权值，Ep 是样本 p 的误差，用下面公式计算：

$$E_p^{\backslash\backslash}=\frac{1}{2}\|Y_p-O_p\|=\frac{1}{2}\sum_k(y_{pk}-o_{pk})^2$$

反向传播学习算法如下：

(1)选取比率参数 r;

(2)进行下列过程直至性能满足要求为止;

① 对于每一训练(采样)输入

a.计算所得输出;

b.按下式计算输出接点的值

$\beta_z=d_z-O_z$，其中 d_z 为期望输出，O_z 为实际输出

c.按下式计算全部其他节点

$$\beta_j=\sum_k W_{i\to k}O_k(1-O_k)\beta_k$$

d.按下式计算全部权值变化

$$\Delta W_{i\to j} = r O_i O_j (1 - O_j) \beta_j$$

②根据所有训练(或采样)的输入值，对权值进行求和，并根据算法修正各权值。

权值的变化与输入误差成正比，训练目标输出必须且只能无限逼近零和一两个值，而绝不能等于它们。因此，当采用 1 作为目标值进行训练时，所有输出世界上呈现出大于 0.9 的值；然而训练的目标值是零时，所有的实际输出都是小于 0.1 的；这样的性能就被认为是满意的。

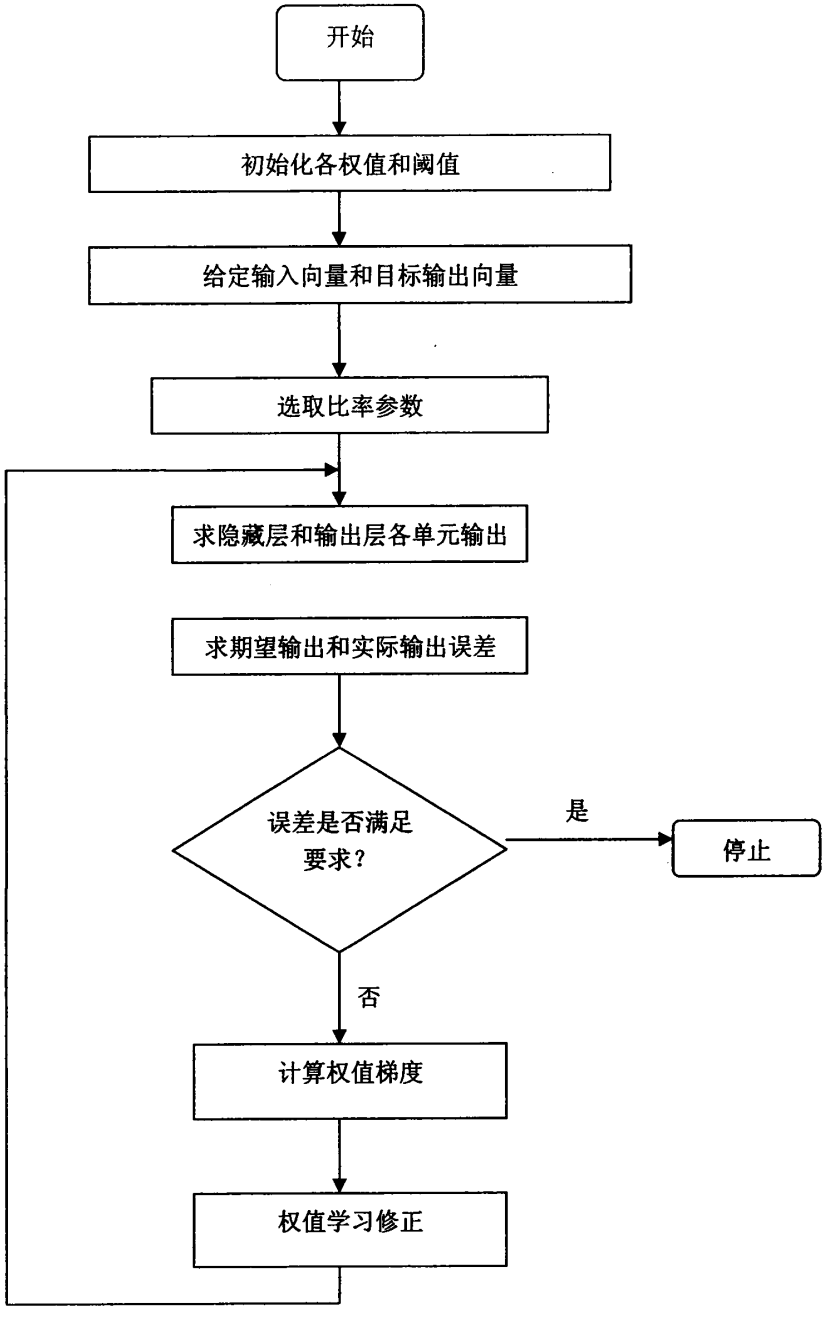


图 12 反向传播网络算法框图

5.4 启发式搜索

5.4.1 搜索概述

知识的搜索和推理是人工智能所要研究的核心问题之一，其所要解决的问题大多是由于结构不良或者是非结构化等特征而不存在相应的成熟的算法可求解，因此智能理由现有的知识和规则进行一步步地搜索或者推理前进。

对知识的搜索分为盲目搜索和启发式搜索。盲目搜索是按照既定的搜索策略进行穷举搜索，不考虑问题本身的特性，也不利用搜索时产生的中间结构，更不更改搜索的策略，因此这样的搜索通常效率不高，对复杂问题的求解便不那么方便，，但是总能找到全局最优的解，因此对结果要求比较高的可以使用盲目搜索，常见的盲目搜索有宽度优先搜索、深度优先搜索和等代价搜索等。启发式搜索即是在搜索中利用产生的中间结果来改变搜索策略，从而改变了搜索方向，加速了问题的求解，常见的启发式搜索有有序搜索、最优搜索 A*算法和 AO*算法等。

5.4.2 A*算法

在介绍 A*算法之间，先介绍两个关键的数据结构：

(1) OPEN 表

用于存放刚生成的节点的表

(2) CLOSED 表

用于存放未曾在 OPEN 表和 CLOSED 表中出现过的节点，这些节点也就是即将要扩展的节点或者已经扩展过的节点。

在启发式搜索过程中，根据不同的搜索策略，会得到不同的将要考察的节点，这也是启发式搜索过程的最关键部分。用于估价节点重要性的函数称为估价函数，其一般形式为：

$f(x) = g(x) + h(x)$

其中 g(x)为从开始节点 S_0 到节点 x 实际已经付出的代价；h(x)是从节点 x 到目标节点 S_g 的估计代价，根据不同的启发搜索策略，得到不同的启发信息和启发信息的形式。

5.4.2.1 局部择优搜索

局部择优搜索算法是将启发式搜索方法应用于深度优先搜索方法，对其进行的改进。其基本思想是：在扩展节点 x 时，用启发式搜索函数 f(x)对此节点的每

一个子节点进行估价值计算，并选择估价值最小的节点作为下一个节点进行考察，该算法的特点是它每次都只是在某节点的子节点的范围内选择下一个要考察的节点，所以其搜索范围比较窄，被称为局部择优搜索。该算法的具体步骤如下：

(1) 把初始节点 S_0 放入 OPEN 表，计算 $f(S_0)$.

(2) 如果 OPEN 表为空，则问题无解，推出。

(3) 从 OPEN 表中取出第一个节点(记为节点 n，默认为估价值最小)放入 CLOSED 表中。

(4) 考察节点 n 是否为目标节点，若是，则求得了问题的解，退出。

(5) 若节点 n 不可扩展，则转第(2)步。

(6) 对节点 n 进行扩展：用估价函数 f(x)计算节点 n 的每个子节点的估价值，并将估价值的大小进行排列后依次放到 OPEN 表中，同时将每个子节点的父节点指针指向节点 n，然后转向第(2)步。

5.4.2.2 全局择优搜索

全局择优搜索算法是对局部择优搜索算法的一个补充，因为局部择优算法在选择扩展节点时只在刚生成的节点的子节点中选择，选择范围窄，容易错过最优的路径，因而又提出了全局择优搜索方法。按这种方法进行搜索时，总是从全局考虑，选择 OPEN 表中所有节点的估价值最小的节点进行考察、扩展。其搜索过程如下：

(1) 把初始节点 S_0 放入 OPEN 表，计算 $f(S_0)$。

(2) 如果 OPEN 表为空，则搜索失败，退出。

(3) 把 OPEN 表中的第一节点(记为节点 n)从表中移除放入 CLOSED 表。

(4) 考察节点 n 是否为目标节点，若是，则求得了问题的解，退出。

(5) 若节点 n 不可扩展，则转向第(2)步。

(6) 对节点 n 进行扩展：首先用估价函数 f(x)计算节点 n 的每个子节点的估价值，同时将其子节点的父节点指针指向节点 n，然后把其子节点都送入 OPEN 表中，最后将 OPEN 表中的所有节点根据其估价值进行从大到小的排序。转向第(2)步。

5.4.2.3 A*算法

在搜索算法中为了能让搜索更有效率，都希望获得一条从初始节点经过节点 n 的最小代价，但是这通常是不可能的，因此引入了 h*(n),表示从节点 n 到目标节点的最小代价，估价函数 f*(n) = g*(n) + h*(n)，其中 g*(n)表示从初始节点 S_0 到节点 n 的最小代价。

其搜索过程如下：

(1) 把初始节点 S_0 放入 OPEN 表，并建立目前只包含 S_0 的图 G。

(2) 检查 OPEN 表是否为空，若空则问题无解，退出。

(3) 把 OPEN 表的第一个节点取出放入 CLOSED 表，并记该节点为节点 n。

(4) 考察节点 n 是否为目标节点，若是则求得问题的解，退出。

(5) 扩展节点 n，生成一组子节点。将不属于节点 n 的父节点的所有节点都加入节点集合 M 中，并把这些子节点作为节点 n 的子节点(其父节点指针指向节点 n)插入图 G 中。

(6) 对节点集合 M 中出现的各种情形，进行如下处理：

① 节点集 M 中的节点没有出现在图 G 中时，设置一个指向其父节点(即节点 n)的指针，并把它们放入 OPEN 表。

② 如果某些 M 集合中的节点已经存在于图 G 中但是还没扩展的，就要确定其父节点指针的指向是否需要修改。

③ 对于不仅出现在图 G 中还已经扩展了的 M 成员，还要确定其子节点的父节点指针是否需要修改。

(7) 把 OPEN 表中的所有节点按估计函数 f* (n) = g*(n) + h*(n)的值从小到大进行排序。

其中 g(n)是对 g*(n)的估计，g(n)>0;对所有的节点 n 均有：$h(x) \leq h^*(x)$，即 h(n)是 h*(n)的下界，。

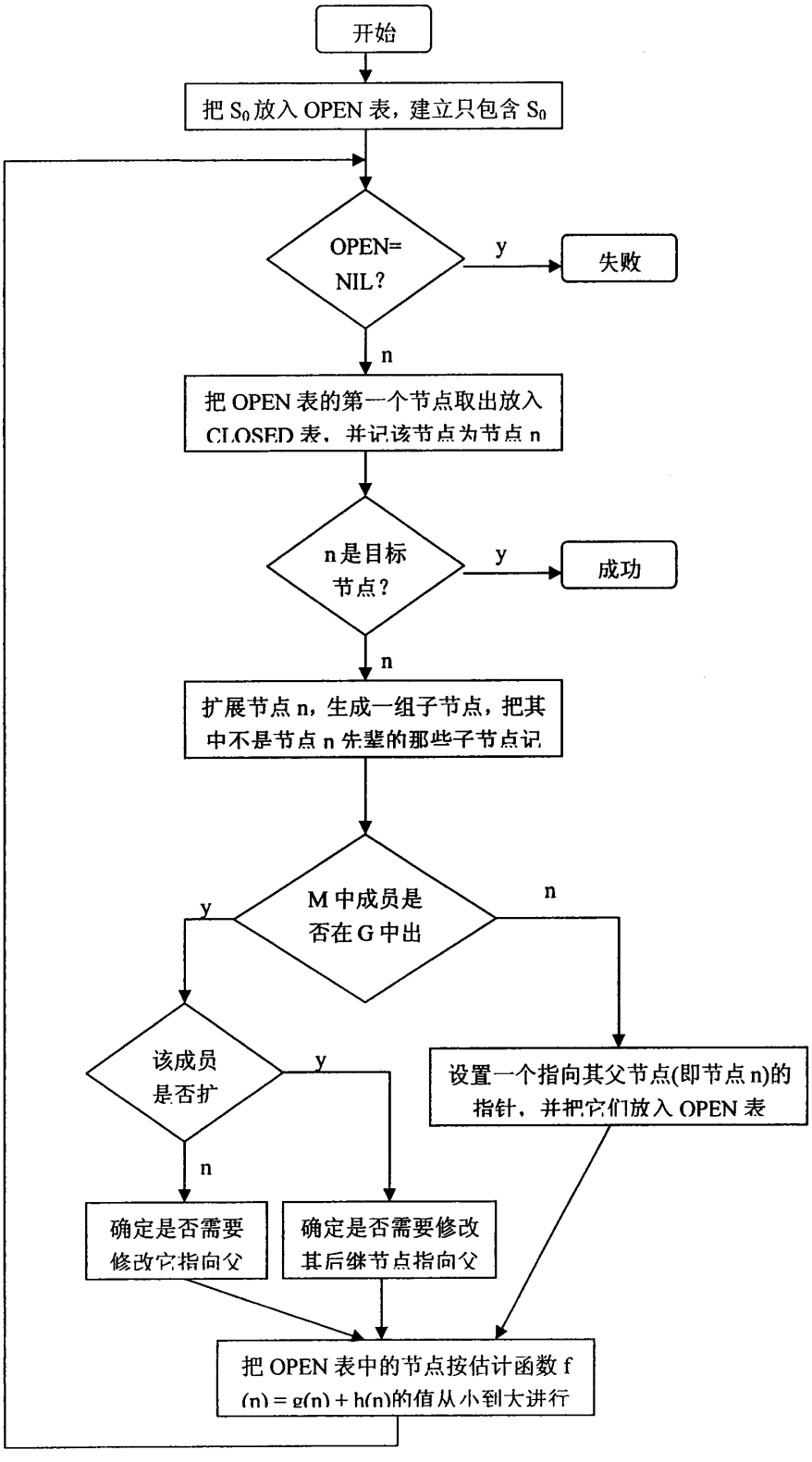

开始
把 S_0 放入 OPEN 表，建立只包含 S_0
OPEN= NIL?
y
失败
n
把 OPEN 表的第一个节点取出放入 CLOSED 表，并记该节点为节点 n
n是目标节点？
y
成功
n
扩展节点 n，生成一组子节点，把其中不是节点 n 先辈的那些子节点记
M 中成员是否在 G 中出
y
n
该成员是否扩
y
设置一个指向其父节点(即节点 n)的指针，并把它们放入 OPEN 表
n
确定是否需要修改它指向父
确定是否需要修改其后继节点指向父
把 OPEN 表中的节点按估计函数 f(n) = g(n) + h(n)的值从小到大进行

图 13 A*算法的流程图

5.5 强化学习

强化学习(reinforcement learning, RL)是从动物学习和参数扰动自适应控制理论中发展出来的，其是现代机器学习的重要组成部分。RL 不同于传统的连接主义学习中，RL 是智能系统将环境映射到行为中去的机器学习方法。通过对环境的描述，环境会给 RL 提供强化信号，针对给出的强化信号值来作为动作的评价，并判断动作的好坏(如，好动作评价则积极，坏动作评价则消极)。RL 就是通过这种不断经历环境的学习，从中获得知识，从而改进动作选择方案来更好地适应环境。RL 系统学习的目标是通过动态地调整参数，来达到强化信号最大化，从而只能体能对环境做出正确且及时的反应。

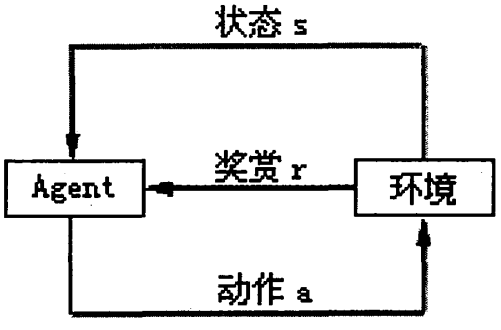


图 14 强化学习模型

为了能达到这个目的，RL 必须不断的从环境到动作的映射中学习。通过不断的试错(trial and error)再根据环境给出的评价值来寻找最优的动作选择策略。现行的对强化学习研究的学习技术大致分成两类：①通过动态规划或者统计技术计算在某一特定环境下某动作的评估值；②通过搜寻 agent 的动作空间来寻找 agent 的最优动作或者最优动作链，这类技术通常由遗传算法或者蚁群算法等来实现。

目前主流的 RL 算法由 TD 算法，蒙特卡洛算法和 Q 学习算法等。

5.5.1 Q 学习

Q 学习(Q-Learning)是强化学习的主要算法之一，其在 1989 年由 watkins 提出。Q 学习是一种与模型无关的强化学习算法，该算法通过马尔可夫决策过程对

环境进行建模，再采用迭代方法逼近最优解，并且以状态-动作对的奖赏值 Q(s, a)作为衡量标准。其中 Q(s, a)表示在状态 s 下执行动作 a 后获得的奖励值，该值越大就说明采取的动作越好。

在学习之前，该算法并不要求了解具体的环境模型，而是可以利用反复执行动作获得奖赏值进行学习。因此常被用于机器人控制领域。

在马尔可夫决策过程中，agent 可以感知周围环境在不同时期的状态变化，并且可以执行动作库中的任何一个动作。在 t 周期里，环境状态为 S_t，agent 执行动作 a_t 后，状态变为 S_{t+1}，同时返回一个回报值 $r(s_t, a_t)$,此时利用如下函数计算 $Q(s_t, a_t)$的值：

$$Q(s_t, a_t) = r(s_t, a_t) + \gamma \max Q(s_{t+1}, a_t)$$

其中γ是折算因子($0 \leqq \gamma \leqq 1$).agent 在每周期时总会选择 Q 值最大的动作执行，然后继续计算 Q 值并更新，反复迭代，知道满足一定条件(比赛结束或者 Q 值收敛)。

为了让 Q 学习的收敛速度平稳，可以在公式中加入学习率 α 来控制学习速率，避免陷入局部最优点而得不到全局最优点。此时 Q 值可以根据如下公司更新：

$$Q(s_t, a_t) = (1-\alpha)Q(s_t, a_t) + \alpha(r(s_t, a_t) + \gamma \max Q(s_{t+1}, a_t))$$

5.5.2 蒙特卡洛算法

蒙特卡洛算法在二十世纪四十年代中期提出的，是一种使用随机数(常见的是伪随机数)来解决计算问题的方法，因此蒙特卡洛算法是一种以概率统计理论为指导的非常重要的数值计算方法，也称为统计模拟方法。其解决的问题定义如下：在一个任意形状的封闭图形（该图形的面积已知）的内部画出一个不规则的形状（也是封闭的），若要求出该不规则形状的面积，可以使用如下方法，均匀地向大封闭图形内部撒 N 个小绿豆，随后统计不贵子几何形状内部绿豆的数量，如有 M 个，则不规则形状的面积接近于 M/N，N 越大，算出来的值越精确。这就是蒙特卡洛算法最初要解决的问题。

传统方法很难解决的问题中大部分是用概率模型描述的。由于概率模型含有

随机因素，因此分析起来要比确定性的模型困难，不是因为难以作定量分析而得不到分析的结果就是会因为计算量太大而无法使用。此时，可以使用蒙特卡洛算法：首先利用该算法建立一个概率模型，该模型的参数或者其他特征量即是所要求的问题的解，然后通过模拟和统计的试验，通过多次随机抽样，统计出某事件发生的百分比。只要试验次数很多，就能非常近似地模拟出事件发生的概率。

5.6 学习算法在 RoboCup 中的应用

由于仿真 2D 机器人足球的复杂性，很难将机器学习的方法直接应用。通常要先把 agent 的决策分成不同的层次，在每个层次中又分成一些不同的任务，对应每个任务再使用相应的学习方法来完成学习任务。结合人类足球的特点，可以把仿真机器人足球中 agent 的决策分为三个层次，分别是个人技术决策、局部战术决策和全局战术决策。

个人技术决策是最底层的决策，其任务是完成一些跟人必备的技术，如跑位、带球、射门和传球等，它们的决策结果是由基本动作组成的动作链；局部战术决策是较高层次的决策，其任务主要是实现几个队友之间的一些战术配合，比如二过一等，决策结果是个人技术层的任务组成动作链；全局战术决策层是球员最高层的决策，是对球队阵型和攻防战术风格等方面的，决策后产生的是战术配合组成的序列。

目前主要是在个人技术层和局部战术层进行相关的学习，如神经网络和机器学习等算法的应用，而对于全局战术层，由于考虑的因素很多，涉及的知识很复杂，所以还没有较成熟的学习算法应用，只能依靠传统的人力手工编码来进行决策。例如，在个人技术层，可以通过蒙特卡洛算法进行球场定位，首先获得自身精确的位置，然后所有相对自身的计算误差才会很小，做出的反应才会最有效；对于射门和传球等技术，可以通过强化学习技术，不断的训练，获得最优的射门角度、射门力度和射门点等参数，才会有精准的射门和传球等动作。

本文主要是在个人技术层完善的基础上，将多 agent 学习算法应用与局部战术层，从而使球队在局部范围内增强攻防能力。

第六章 基于多 agent Q 学习的 RoboCup 局部配合策略

6.1 随机策略(Stochastic Game，SG)

马尔可夫决策过程假定的是 Agent 所处的环境是固定并且不存在其它自适应 Agent 的，因此它不满足多 Agent 环境；Michael L. Littman 等人提出了随机对策(SG) 来作为多 Agent 强化学习的框架，随机对策是在 MDP 环境中引入了对策论，是对 MDP 环境的泛化，也是在多状态下对矩阵对策进行相应的延伸。

随机对策可定义为五元组<N, S, {A_1, A_2, ...A_n}, T, {R_1, R_2, ...R_n}>,其中：N 为 n 个 agent 的集合，N={1, 2, ...， n}，S 是对环境离散表示后的环境状态集，Ai 为 $agent_i$ 的可选的动作集，T:S×A×S→[0,1]为状态转移函数。T(s_i, a, s_{i+1})表示从状态 s_i 经过 agent 的联合行动 a={a_1, a_2,...a_n}到达状态 s_{i+1} 的概率；R_i: S×A×S→R 为 $agent_i$ 的回报函数。在随机对策的框架下，Q 值可定义为：

$$Q^i_{t+1}(s',\alpha_t)=(1-\alpha)Q^i_t(s',\alpha_t)+\alpha[r^i_t+\beta V^i(s_{t+1})]$$

其中，α (0< α <1)为控制收敛的学习率，$V^i(s_{t+1})$是一个状态值函数，

$$V^i(s_{t+1})=\max f^i(Q^i_t(s_{t+1},\alpha_t))$$

Q 值是通过上述公式的反复迭代而收敛的，上述公式的关键因素是学习策略，即行为α_t的选择方式和函数$V^i(s_{t+1})$的定义[3,6]。不同的选择方式会产生不同的多 Agent 学习算法。

6.2 状态-动作对的确定

球场上情况复杂，需要考虑的因素太多。但是要达到局部的配合，首先要知道自己所处的位置 S_A 和球的位置 S_B，这样才能确定球员本身是否出在配合的范围内；其次是否自己控球 L_A 还有是否是我方控球 L_B，用来确定配合的策略是防守还是进攻。于是将<S_A, S_B, L_A, L_B>作为球场上局部范围内环境状态的描述。

为了减少因为 S_A, S_B 连续的坐标信息而增加的环境状态的数量，笔者对球场上的位置进行了离散。本论文中的位置都是在这个离散方式的基础上讨论的。具体的离散方法如下：笔者将球场划分为 60*10 个小的区域，其中 X 轴方向分为 60 等份，Y 轴方向分为 10 等份，这样就可以用一对离散化的(i , j)来描述球场上的位置信息。L_A, L_B 的取值为 0 或 1，0 表示不控球，1 表示控球，当球处于自由状态时(即任何一方都不控球)，L_A, L_B 取 0。

因此环境的状态信息描述为<S_A, S_B, L_A, L_B>=<(iA , jA), (i_B , j_B),{0,1},{0,1}>(0≤i≤59, 0≤j≤9)

笔者根据状态信息来确定相应动作集：当 Agent 为控球球员，动作集为上述决策树中控球时的动作，即{shoot, pass, cross, dribble, selfpass}；当 Agent 不控球时，但是控球方为我方时，动作集为{gotopoint, turntoball}；当 Agent 不控球时且对方控球时，动作集为{gotopoint, turntoball, tackle}。

6.3 reward 值的确定

本文中的 reward 值的确定比较复杂，分为：Agent 控球，Agent 不控球但我方有控球权，对方控球三种情况，下面将分别讨论这三种情况下的 reward 值的确定。

6.3.1 agent 控球

① 我方进球，r=1

② 球到达射门点， r=0.9

③ 球出界，r=0

④ 变为对方控球，r=-0.9

⑤ 对方进球，r=-1

⑥ 否则，r=区域基础回报+区域内部回报+f(x)

上述的射门点是球队根据球队特点和禁区内的各种复杂情况事先计算出来的位置，在这些位置射门时进球的概率很高，因此到达这些点时回报值应仅次于

进球时的回报值。

在情况⑥中，笔者继续沿用以前的[章惠龙等提出][6]算法，但是区域的划分不同，划分的具体情形如下：

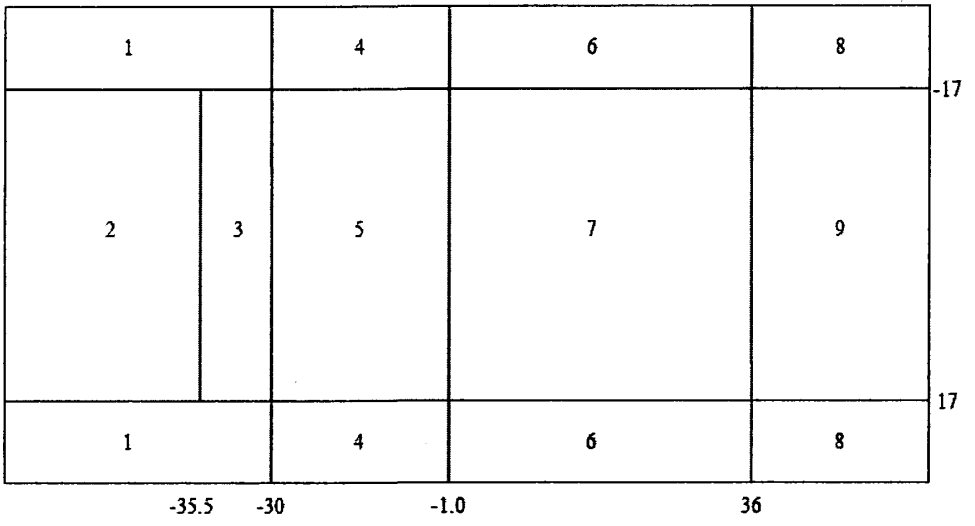


图 15 球场划分策略

如图 15，将球场划分为关于 X 轴对称的 9 个区域，笔者将这 9 个区分的基础回报分别设定为-0.4，-0.6，-0.5，-0.3，-0.3，0.3，0.3，0.5，0.7。

划分之后，球在不同区域时区域内部回报不同。如果采用此方法离散球场信息会是的 r 值过于死板，难以体现在区域内部随着球位置变化时 r 的变化，因此还要加上一个函数 f(x)， f(x)的值域应控制在(0,0.2)，这里笔者采用 sigmoid 函数，即 $f(x)=\frac{1}{5(1+e^{-x})}$。

上面两个参数的设定不仅保证了区域之间的差异性，也保证了区域值得连续性，使得两个区域的交界处的回报值差距不至于太大。

区域内部回报是由球员之间的位置关系确定的，在球队中笔者事先用一个函数确定在这个范围中的最适合配合球员 P_1 和最危险的对方球员 P_2，因此区域内部回报：

当 $d_1>5.0$

区域内部回报=(X_A+（d_1-5.0）*2.0*X_A-d_3)/100;

当 $3.0<d_1\leqslant 5.0$

区域内部回报=(X_A+（d_1-d_2）*2)/100;

当 $d_1 \leqslant 3.0$

区域内部回报=(X_A-d_3)/100-本区域基础回报;

公式中 X_A 为球的 X 坐标，d_1 为 P_2 与球之间的距离，d_2 为 P_1 与球之间的距离，d_3 为球与对方球门之间的距离。公式中通过控制 X_A 的倍数来控制带球速度；通过控制 d_3 的大小来控制是否将球推向对方球门；通过控制 d_1 的大小来在一定程度上控制是否摆脱 P_2；这里 P_1、P_2 是根据球员是否适合铲球，有没有球员盯住等许多复杂因素确定的，而不是单纯的使用最近的球员，确定方法不是本文重点，这里将不再讨论。

6.3.2 agent 不控球但我方有控球权

① 如果自身是最佳配合球员，r=区域基础回报+f(x)+(X_B+d_4-d_5)/100

② 如果自身不是最佳配合球员，r= (X_B+d_4)/100

其中区域基础回报和 f(x)就是前面讨论过的值，d_4 为自身与球之间的距离，d_5 为经过一定方法得到的对自己最有威胁的对方球员(一般情况下为离自己最近的对方球员，除非此队员有人盯防)与自己的距离，通过控制 X_B 的倍数来控制跑动速度。这样确定的目的是因为，如果自身为最佳配合球员，控球球员可能将球传给自己，因此要跑向球，并且要保证周围没有对方球员盯防，如果自身不是最佳配合球员，则跑向对方球员进行阻挡对方跑位，此外，由于情况①加入了区域基础回报，则会考虑到区域之间的差异，尽量向对方球门方向带球。

6.3.3 对方控球

① 如果得到球，r=1.0

② 如果自己是离球最近的球员，r=(X_C-d_6)/100

③ 否则，r=(X_C +d_7*2+d_6-d_8)/100

其中，d_6 为与球的距离，d_7 为自身与最近对方球员 P_3 的距离，d_8 为 P_3 与球的距离，X_C 与目标的距离有关，用来控制球员的跑动速度。由此可见，当对方控球时，己方球员首先是争取获得控球权，如果不能取得控球权，则先主动上前

去拦截，否则盯防距离自己最近的进攻球员，挡在球与该球员 P_3 之间，防止控球球员传球。

6.4 状态-动作表中的 Q 值更新

本文实验的 Q 值是按照下面公式进行更新的：

$$Q_{t+1}(s,\alpha_t)=(1-\alpha)Q_t(s,\alpha_t)+\alpha[r_t+\beta V(s_{t+1})] \quad (3)$$

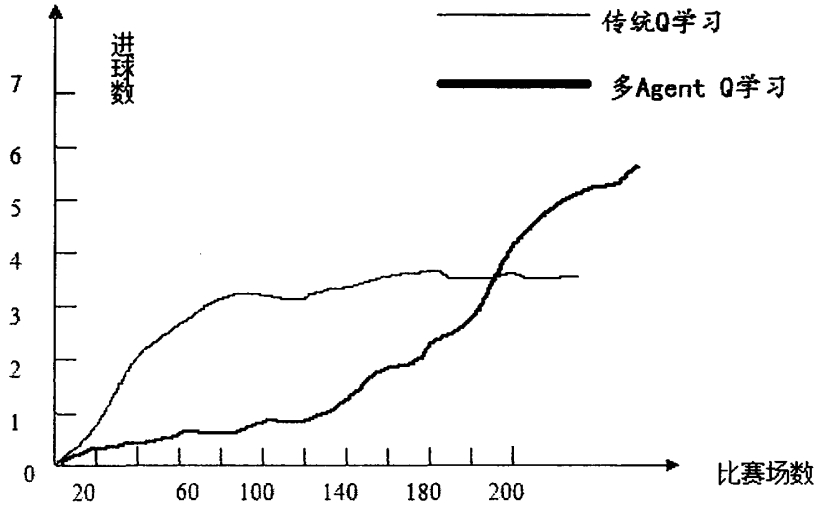
其中,当在训练的时候 α =0.35，当 Q 值趋于稳定的时候 α =0.1，β=0.8,

$$V(s_{t+1})=\max Q_t(s_{t+1},\alpha_t)-Q(s_t,\alpha_t)$$

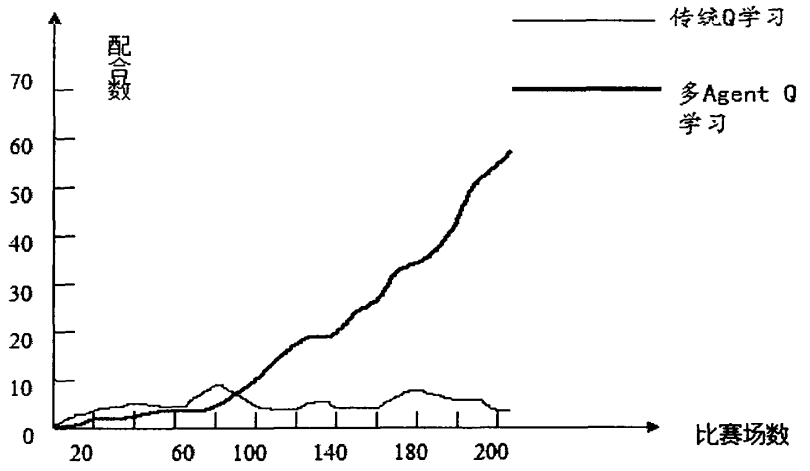
每个周期 Agent 都会从状态-动作表中找到对应状态中 Q 值最大的动作执行，执行过后再根据上述公式更新对应的 Q 值。更新时都会找到 α_t 中的所有动作，这时就要选择一个范围 d，在这个范围内的所有球员都将考虑进去，用来更新 Q 值，笔者根据球的最大速度和衰减率得出 d=30。

6.5 实验结果

将上述算法植入到校队代码中并在 RoboCup 仿真 2D 的平台上进行实验。通过大量的反复学习，使得 Q 值收敛于一个稳定的值，以进球和配合数作为统计数据，通过实验发现，配合数和进球数较采用传统 Q 学习的方法有明显上升。



a. 改进算法与传统算法进球数的比较



b．改进算法与传统算法配合数的比较

图 16 进球数和配合数的统计数据

如图 16.a.所示，改进的多 Agent 方法进攻能力有所提升，图 5.b.所示，改进的多 Agent Q 学习的配合数明显增多，另外还可以发现传统的多 Agent Q 学习的配合数一直不稳定，这说明传统多 Agent Q 学习不存在配合，即使有配合也只是偶然出现的巧合，因为它设计时即没有考虑到 Agent 配合的情形。当 Q 值趋于稳定时，再进行 200 次防守实验，实验结果如下表：

表 1 防守实验统计结果

平均被进球数	传统 Q 学习代码	多 Agent Q 学习代码
传统 Q 学习代码	2.03	0.35
国内八强代码	4.12	1.59
多 Agent Q 学习代码	5.81	3.44

从表 1 可以看出，多 Agent Q 学习的代码相对于传统多 Agent Q 学习代码，在平均被进球方面都明显减少，说明防守能力得到增强；相反，传统多 Agent Q 学习的代码相对改进的代码被进球数显著增加，这说明改进后的代码整体进攻实力得到增强，与实验的初始设计目标相符。

第七章 总结与展望

7.1 总结

本文主要是针对RoboCup仿真2D机器人足球项目所要研究的重点进行了研究，并在此基础上改进了安徽大学仿真 2D 球队——Dreamwing2D。本文所涉及到的研究重点分为：

（1） 仿真 2D 球队的阵型表示与跑位

本文延用 Helios 球队对阵型的表示方法，首先采用 Delaunay 三角化方法将球场进行划分，用这种方法划分的球场具有唯一性，可根据算法无差别的构建。在此基础上将球员相对于球的位置来进行阵型表示，在比赛时通过载入这样的阵型来进行基本的跑位，再在基本跑位点的基础上进行其他决策来选出最佳的跑位点。

（2） Agent 学习策略

本文通过对 Agent 决策层划分为三个层次：个人技术决策、局部战术决策和全局战术决策，很好地将学习算法应用与各层，使得球员具有一定的智能。在个人技术决策层上运用强化学习进行训练，以达到良好的个人技术效果，如，射门精准，传球精确，跑位及时等。在局部战术层上运用多 agent Q 学习进行学习，达到局部团队配合良好。

（3） 团队合作

目前，由于受到硬件和软件的限制，计算机的计算能力受限，在保证实时性的基础上很难实现全局团队合作，因此，本文中采用局部团队合作来达到期望的效果。首先，通过对球球的归属对球场状态进行划分，并对在每个状态下 agent 可以采取的动作进行归纳。其次，对球场进行细分，并赋予相应的数值，根据球所在的位置和归属对 agent 相应动作集的动作进行评估，采用 Q 值最大的动作执行，并获得相应的奖赏值，根据奖赏值的数值再对 Q 值进行更新，并作为下一周期的动作选择参考。最后，在局部范围内(球周围 30m 的范围)，所有 agent 都采用此算法进行决策。

通过实验证明，本文中采用的方法可以使球队的攻防能力得到明显的加强，球队配合数也有显著增加。

7.2 展望

在本文的基础上，我们对安徽大学仿真 2D 队 Dreamwing2D 进行了相关的改进，球队的整体攻防能力也有一定的提高，但是球队还有很多不足的地方需要改进，在今后的研究工作中的重点将放在以下几点：

(1) 对通信信道的利用

由于目前的通信信道有很大的噪音，获得的也是不可靠的消息，所以 Dreamwing2D 中是取消了球员之间的通信，这是不符合现实情况的，今后会对通信信道进行降噪处理，使获得的信息相对准确，并且根据信息进行相关的决策，如跑位、接球或者协防。

(2) 对离线教练的完善

由于技术和理论的不足，Dreamwing2D 的离线教练一直不够完善，只能进行一些简单的规划，而不能进行高层的抽象而获得更为准确的信息。

(3) 对个人技术的优化

球员的个人技术还是存在可发展的空间，可以利用强化学习算法进行强化训练，来获得更优质的动作，确保比赛时不出现失误，例如传球时对接球队员的选择，可以进行局部优化而获得最佳的传球队员，从而在比赛时减少丢球。

参考文献

[1] Luiz A.Celiberto Jr., Carlos H.C.Ribeiro, et al. Heuristic Reinforcement Learning applied to RoboCup Simulation Agents [J]. Springer Berlin Heidelberg, 2008, 5001: 220- 227.

[2] Luis Mota, Nuno Lau, Luis Paulo Reis.Co-ordination in RoboCup's 2D Simulation League: Setplays as flesible, Multi-Robot plans[J]. IEEE,2010:362-367

[3] BAI aijun, WU feng, CHEN xiaoping. Online Planning for Large MDPs with MAXQ Decomposition: Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems- Volume 3, 2012[C]. Richland: International Foundation for Autonomous Agents and Multiagent Systems. 2012,3:1215-1216.

[4] ZHANG zhongzhang, CHEN Xiaoping. A Factored Hybrid Heuristic Online Planning Algorithm for Large POMDPs: Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence, 2012[C]. Artificial Intelligence.2012:934-943.

[5] 向中凡. Q 学习角色值法在机器人足球比赛中的应用[J].电子科技大学学报,2007,36(4):809-812

[6] 孟祥萍, 王欣欣, 王圣镔. 多 Agent Q 学习几点问题的研究及改进[J]. 计算机工程与设计, 2009,30(9):2274-2276.

[7] 刘亮,李龙澍. 基于局部合作的 RoboCup 多智能体 Q-学习[J]. 计算机工程, 2009,35(9):11-16

[8] 柯文德,彭志平, 蔡则苏等. 基于 π 演算的足球机器人协作 Q 学习方法[J]. 计算机应用, 2011,31(3):654-656.

[9] S.Kalyanakrishnan, Y. Liu, and P. Stone. Half field offense in RoboCup soccer: A multiagent reinforcement learning case study[J]. Computer Science,2007,4434:72-85

[10] 章惠龙,李龙澍. Q 学习在 RoboCup 前场进攻动作决策中的应用[J]. 计算机工程与应用,2011,15(2):311-316.

[11] 顾晓锋, 张代远.机器人足球比赛接球策略设计[J].计算机应用, 2005,25(8):1858-1859.

[12] 李实，陈江.清华机器人足球的结构设计与实现[J].清华大学学报，2001,41(7)：94-97

[13] 张波，蔡庆生，陈小平等.基于智能团队的 RoboCup 仿真球队.Proceeding of the 3rd World Congress on In telligebt Control and Automation, Jun28-July2, 2000, Hefei, P. R. China.

[14] 王永庆.人工智能原理与方法[M].西安：西安交通大学出版社,1998:263-276,420-431.

[15] 蒋宗礼.人工神经网络导论[M].北京:高等教育出版社,2008:30-52.

[16] 蔡自兴，徐光祐.人工智能及其应用[M].北京：清华大学出版社,2004:126-134,242-263.

[17] 蔡强.限定 Voronoi 网格剖分的理论及应用研究[M].北京:北京邮电大学出版社,2010:14-26,54-62.

[18] Thomas H.Cormen, Clifford Stein.算法导论[M].北京：机械工业出版社,2009:192-212.

[19] Simon Haykin.神经网络与机器学习[M].北京:机械工业出版社,2010:627-668.

[20] 尚丽.RoboCup2D 中的多 Agent 协作技术研究[D].合肥：合肥工业大学计算机学院,2010.

[21] 杨志雄.机器人仿真 2D 足球比赛的研究[D].合肥:安徽大学计算机学院,2006.

[22] 张明开.仿真 2D 机器人球员训练算法的研究与设计[D].合肥：安徽大学计算机学院,2007.

[23] 陈小平.仿真机器人去求：设计与实现[EB/OL]. http://ai.ustc.edu.cn/en/robocup/2D/training.php, 2008.

[24] 申迅，刘国栋.基于 Q 学习 RoboCup 前锋的射门训练[J].计算机工程与应用,2011,47（18）：53-55.

[25] 周勇，刘峰. 基于改进的 Q 学习的 RoboCup 传球策略研究[J].计算机技术与发展，2008,18（4）：63-66.

[26] 陈玉明，张广明，赵英凯.基于混合 Q 学习的多 Agent 系统[J].制造业自动化，2010,32（9）：61-63.

[27] 祝宇虹，毛俊鑫.基于人工情感与 Q 学习的机器人行为策略[J].机械与电子，

2011，(7):61-65

[28] 曹卫华，吴敏.模糊 Q 学习的足球机器人双层协作模型[J].智能系统学报，2008,3（3）：234-238

[29] 章惠龙.RoboCup 仿真 2D 中的 Agent 只能决策系统[D].合肥：安徽大学计算机学院，2012.

[30] Xiong Li, Wei Chen, Jie Wang.The Application of Hybrid Distributed Reinforcement Learning Algorithm in RoboCup 2D Soccer Simulation System[J].Springer,2008:651-658.

[31] Tong-liang. A Speedup Convergent Method for Multi-agent Reinforcement Learning[J].IEEE,2009:653-656.

[32] 徐美清，刘国荣，周桂珍，等. 基于近似 Voronoi 图的移动机器人实时路径规划[J].微计算机信息，2010,26（5）:157-159.

[33] 瞿彬.Voronoi 图在足球机器人规划中的应用研究[J].科技信息,2008,(4):90-91.

[34] 冯洪奎，鲍劲松，金烨.广义 Voronoi 图求解多机器人运动规划[J].计算机工程与应用，2010,46(22):1-3.

[35] 张志军，李峰，曹布阳.一种基于 Voronoi 图求解车辆路径问题的混合启发式算法[J].计算机应用研究，2010,27(2):515-518.

[36] Jing Huang, Bo Yang, Dayou Liu.A Distributed Q-Learning Algorithm For Multi-Agent Team Coordination[J].IEEE, 2005,1(5):108-113.

[37] Choonghyeon Lee, Kyungeun Cho, Kyhyun Um.A stratigy for improving performance of Q-learning with prediction information[J].IEEE, 2006,8(6):117-121.

[38] Zheng Qin, Jason Gu.Neural Q-learning in Motion Planning for Mobile Robot[J].IEEE,2009,4(9):1024-1028.

[39] 张恒，刘艳丽，孙晋.不完整地图中移动机器人蒙特卡洛定位研究[J].计算机应用研究, 2010,27(2):509-511.

攻读硕士期间发表的论文

（1）赵发君，李龙澍.基于多 agent Q 学习的 RoboCup 局部配合策略[J].计算机工程与应用，已录用。

致 谢

三年的研究生时光即将结束，在这三年中无论在学习还是生活上，我都收获颇丰，我之所以能获得今天的成绩，这与我的家人导师对我的指导，对我的支持和同学对我的帮助是密不可分的。

首先，我要感谢我的导师李龙澍老师，要感谢他这三年来对我的谆谆教诲和鼎立的帮助。在论文研究期间，李龙澍老师不仅给我提供了良好的研究环境和充足的资金支持，还给本文的研究提供了实验平台，即带领我们参加每届的 RoboCup 中国公开赛，使得理论的研究有了实践的机会。在本文撰写期间，李老师还专门抽出时间指导我的写作，给我拓宽了研究和写作的思路，并在论文即将完成时给我提出了非常宝贵的修改意见。另外，我还要感谢杨为民老师和罗罹对我的指导，尤其是杨老师，每当一次比赛结束，杨老师就会帮助我仔细分析当前代码的优缺点以及理论的不足。

其次，我还要感谢我的同学们，尤其是我们同实验室的同学：张青、张生、王孝贵、韩丽丽、王芳、华骁飞和朱超，实验室之所以有了他们才会有良好温馨的学习环境。同时，还非常感谢给我莫大的指导和帮助的师兄们：章惠龙、朱国龙和王唯翔等，正是因为有你们在我的身边，才使得我的论文研究进展顺利。此外我还要感谢我可爱的 2515 全体室友，有了你们我的生活才如此丰富多彩，谢谢你们对我的帮助。

最后，我还要特别感谢我的家人，正因为有你们的理解和支持作为我强大的后盾，才让我能安心地在学校学习和研究。从今天起，我还会在你们的注视下逐渐成长，变的独立、成熟。也祝愿你们身体健康，生活幸福。