软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2020,31(6):1619−1637 [doi: 10.13328/j.cnki.jos.005996]
©中国科学院软件研究所版权所有.
E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

自主机器人多智能体软件架构及伴随行为机制*

毛新军[1,2], 杨 硕[1], 黄裕泓[1], 王 硕[1]

[1](国防科技大学 计算机学院,湖南 长沙 410073)

[2](复杂系统软件工程重点实验室(国防科技大学),湖南 长沙 410073)

通讯作者: 毛新军, E-mail: xjmao@nudt.edu.cn

摘 要: 自主机器人是一类由计算机软件控制的信息物理系统,如何支持该类机器人在开放环境下的有效和协调运行,是自主机器人控制软件(control software of autonomous robot,简称 CSAR)研究与实践面临的一项重要挑战.基于组织理论的思想,采用 Structure-in-5 的组织架构模式,提出了基于多智能体的 CSAR 的软件架构 MaRSA(multi-agent robotic software architecture),通过独立抽象 CSAR 的行为规划、分发、执行等软构件,并显式加强这些构件间的交互,从而为自主机器人行为的有效规划和协调实施奠定架构基础;提出了基于 MaRSA 架构的伴随行为机制,从因果性、时序性和按需性等 3 个方面建立了机器人观察行为和任务行为间的伴随关系,并基于分步规划和动态决策的思想,设计并实现了伴随行为的自主决策算法 DAAB(decision algorithm of accompanying behaviors).分别在仿真环境和实际机器人环境下设计了对比性实验,结果表明:与主流的反应式行为决策算法和 BDI 式概率决策算法相比较,基于 MaRSA 和伴随行为机制的 DAAB 算法所生成的伴随行为规划在开放环境下具有可行性和更高效的执行效率.

关键词: 自主机器人控制软件;Structure-in-5;多智能体系统;软件架构;伴随行为

中图法分类号: TP311

中文引用格式: 毛新军,杨硕,黄裕泓,王硕.自主机器人多智能体软件架构及伴随行为机制.软件学报,2020,31(6):1619−1637. http://www.jos.org.cn/1000-9825/5996.htm

英文引用格式: Mao XJ, Yang S, Huang YH, Wang S. Towards software architecture and accompanying behavior mechanism of autonomous robotic control software based on multi-agent system. Ruan Jian Xue Bao/Journal of Software, 2020,31(6): 1619−1637 (in Chinese). http://www.jos.org.cn/1000-9825/5996.htm

Towards Software Architecture and Accompanying Behavior Mechanism of Autonomous Robotic Control Software Based on Multi-agent System

MAO Xin-Jun[1,2], YANG Shuo[1], HUANG Yu-Hong[1], WANG Shuo[1]

[1](College of Computer, National University of Defense Technology, Changsha 410073, China)

[2](Laboratory of Software Engineering for Complex Systems (National University of Defense Technology), Changsha 410073, China)

Abstract: Autonomous robot is a kind of complex cyber-physical system controlled by software. To support robots to operate in open environment in an effective and cooperative way is a great challenge for the researches and practices of control software of autonomous robot (CSAR). Adopting organization theory, this paper presents a multi-agent software architecture MaRSA (multi-agent robotic software architecture) that takes structure-in-5 organization style for CSAR. The software components of plan, dispatch, and execute behaviors of robot are independently encapsulated and explicitly separated, which lays architecture foundation for the flexible cooperation and continuous interactions among these components. The paper further proposes an accompanying behavior mechanism to enrich the

* 基金项目: 国家自然科学基金(61379051, 61532004)
Foundation item: National Natural Science Foundation of China (61379051, 61532004)
本文由“信息物理系统软件设计自动化”专题特约编辑卜磊教授、陈铭松教授、朱祺教授、刘超教授推荐.
收稿时间: 2019-08-14; 修改时间: 2019-10-23; 采用时间: 2020-01-13; jos 在线出版时间: 2020-04-18

interactions of observation actions and task actions, defines three kinds of accompanying relationships on the causality, temporal, and on-demand viewpoints, as well as designs a two-step dynamic decision algorithm DAAB (decision algorithm of accompanying behaviors) for planning accompanying behaviors. Two experiments are conducted in simulation robot environment and the real robot environment respectively, and the results show that comparing with the reactive behavior planning algorithm and BDI-based probabilistic planning algorithm, the proposed algorithm DAAB can produce plans that operate in open environment with high efficiency and low efforts to accomplish tasks.

Key words: CSAR; structure-in-5; multi-agent system; software architecture; accompanying behavior

随着计算的泛在化及计算机应用的不断拓展,计算机软件与各类物理、社会系统深度融合,形成更为复杂的人-机-物共融系统已成为必然趋势[1],由此产生新的研究问题和热点[2].机器人是一类典型和复杂的信息物理系统,软件是其核心和关键,负责与外部环境进行交互、处理传感数据、管理机器人资源、决策和控制机器人运行等等[3,4].近年来,机器人应用正从传统的工厂、车间等典型工业制造领域逐步延伸到诸如医疗、家庭、娱乐等新颖的应用领域[5].这种应用变化的关键是:机器人的运行环境正从传统的封闭、静态、确定的单一环境(如生产车间)逐步过渡到开放、动态、难控和不确定的人-机-物高度融合的环境,进而对机器人行为的自主性、安全性、适应性和灵活性等提出了新的要求.本文针对运行在开放环境下、通过与环境持续交互和自主决策以实现任务的自主机器人[3,6],研究其控制软件(control software of autonomous robot,简称 CSAR)的设计和构造问题,以支持机器人在开放环境下的有效和协调运行.

不同于纯粹的信息系统或者其他的信息物理系统,自主机器人的行为具有多样化和多目的性特点,既有物理行为(如移动和抓取)也有计算行为(如图像检测和识别等);既有达成任务实现、影响环境状态的行为(称为任务行为),也有观察外部环境、主动获取环境状态的行为(称为观察行为).机器人的多样行为通常由不同的软硬构件负责执行,并在执行时序、并发、交互等方面存在固有的复杂性.例如:机器人在实施任务行为(如通过机械臂抓取物品)时,需要同时实施观察行为,从而为任务行为提供必要的输入数据(如待抓取对象的位置信息).对于开放环境下的自主机器人而言,其控制软件需要重点解决任务行为实施过程中的行为协调性和有效性问题.具体地讲,行为协调性是指任务行为和观察行为二者间要相互配合,任务行为需要通过观察行为获得行为实施所需的外部环境信息,观察行为可获取任务行为的实际执行效果,并为行为的重规划等提供决策依据.行为有效性是指所规划的任务行为和观察行为以及它们之间的交互始终以任务为驱动,其结果最终能够促使任务的有效达成.由于自主机器人所在环境变化的动态性和不可预知性、机器人行为实施效果的难控性和不确定性、机器人对外部环境变化感知的不准确性和不完整性,如何确保机器人行为的协调性和有效性,是 CSAR 研究与实践面临的一项重大挑战.

为了迎接上述挑战,CSAR 的研究与实践需要解决两个方面的关键问题.

- 一是如何寻找有效的机制和方法来实现任务行为和观察行为的协调和有效交互,这涉及自主机器人行为的决策、规划和执行等方面的研究.现有的研究通常直接将反应式控制理论[7]、控制环路的决策理论、自适应控制模型[8]、智能体的认知模型和决策方法[9]等应用到机器人领域,未能充分考虑开放环境对机器人行为决策和实施带来的挑战,未能显式区分和抽象机器人不同类型的行为.无论是反应式决策方法还是基于知识或者认知推理的决策方法[10,11],它们或者将机器人不同行为及其内在关联性在设计阶段预先封装和实现在相关行为规则中,导致机器人的行为缺乏灵活性和自主性,难以应对环境动态变化带来的挑战;或者将机器人不同类别的行为集中封装和固化在一起加以实现,使得机器人不同类型行为之间的交互缺乏灵活性和协调性;
- 二是如何为 CSAR 的开发和运行提供软件工程技术和支撑平台.近年来,围绕机器人控制软件的研究与实践有诸多的工作和成果,涉及自适应模型[8]、程序设计语言[9]、软件架构[10,12,13]、开发框架[14]等方面.然而,如何通过 CSAR 的架构设计来实现其行为的协调性和有效性,仍是一个开放的研究问题.

本文针对自主机器人的特点及其在开放环境下行为协调、有效实施所面临的挑战,提出了基于多智能体系统的 CSAR 架构 MaRSA;在此基础上,设计了自主机器人的伴随行为机制以及伴随行为的自主决策算法

DAAB(decision algorithm of accompanying behaviors).

本文第 1 节介绍相关的研究工作.第 2 节介绍自主机器人的典型应用案例和场景,提出 CSAR 的多智能体抽象模型以及基于组织理论的软件架构 MaRSA.第 3 节设计基于 MaRSA 的自主机器人伴随行为机制及行为决策算法.第 4 节开展实验及分析,以检验本文所提出架构、机制和算法的可行性和高效性.最后,对本文工作进行总结和展望.

1 相关工作

1.1 自主机器人行为决策方法及软件架构

(1) 反应式决策方法及基于反应式的层次式架构

在该方法中,行为决策的本质是一个外部变化-响应规则的映射过程.环境变化触发机器人决策系统中相应的反应式规则执行,每个反应式规则描述了在什么情况下该执行什么样的动作.这种决策方式具有实现简单、快速响应等优点,能够满足机器人决策对实时性的要求;不足之处在于需要预先定义好规则且不具备行为的运行时规划能力.基于反应式决策方法的机器人控制软件通常采用层次式的架构,整个软件由若干个相互交互、可并发执行的反应式层次组成,每个层次对应于一个有穷状态机,可接收感知输入,控制效应器实施行为[15].各个层次的模块单元独立、平行工作,无需全局的指导和协调.典型的工作有 Brooks 的包容式体系结构[7]、AuRA[16]、iB2C[17]等.

(2) 基于 Belief-Desire-Intention(BDI)范型的决策方法

BDI 模型是 Bratman 提出的一个著名 Agent 决策模型,使用信念(belief)、愿望(desire)和意图(intention)来解决智能体的决策问题.BDI 模型的基本假设是任何动作都由 practical reasoning 的两个步骤驱动[18],其中,第 1 步为目标-商讨(goal-deliberation),即根据 Agent 的信念决定在当前状况下哪些愿望是 Agent 要追求的;第 2 步为手段-目的推理(means-end reasoning),即如何使用 Agent 可用的方法,实现上一步中的具体愿望.目前,已有各式各样的基于 BDI 模型的决策方法出现和广泛应用,如 PRS[19],JACK[20]和 JADEX[21].当前,BDI 决策模型在机器人任务决策中应用最为广泛的就是概率规划方法(probabilistic planning)[22,23],其核心是引入先验概率、后验概率等概念,根据获得的传感器数据来估计机器人的信念状态.“概率估计”的决策方法能够提高机器人规划决策的准确性和鲁棒性.

(3) 混合式的决策方法和层次式的软件架构

该方法将多种决策方法集成在一起,发挥每种方法的优势,既可支持行为的反应式实时响应,又可支持行为的持续性规划,如 Firby 提出的 RAPs 系统[24]、Volpe 的 CLARAty 系统[25].整个软件系统由多个层次构成,每个层次负责某种特定的决策方法,不同层次之间相对独立.典型的层次式模型有 SERA[12],walkman[17]等.

1.2 自主机器人控制软件的开发支持

(1) 面向对象技术

基于对象技术来实现控制软件架构及其构件,其优势是具有良好的程序结构和可重用性,有众多面向对象程序设计语言的支持(如 C++,Python 等),可有效支持自主机器人控制软件的功能层、控制层和交互层等的开发,如 Miro[26],ARIA[27],Robotics API[28]等;但对象模型存在一些固有不足,如无法主动感知环境变化、不具有自主和自发的行为、一旦实例化之后其结构和行为将无法改变等等,因而在支持 CSAR 的自主决策层、自主管理层等方面有其局限性.

(2) 基于构件技术

通过构件接口的灵活组装来满足自主机器人软件开发的不同需求,如可重用性、模块化等.例如,Saddek Bensalem 等人提出了基于形式化构件的方法以及相应的体系结构 LAAS 来开发可信的自主机器人软件[10].基于构件技术可有效支持软件重用,但其很少关心自主机器人软件决策层的行为自主性和自发性等问题以及管理层的自我管理问题,也没有为此提供相应的支持,因而在支持自主机器人软件的自主决策层、自我管理层等

方面存在局限性.

(3) 基于智能体技术

借助多智能体系统的技术来设计和实现机器人控制软件.该技术手段可充分发挥智能体技术的行为自主性、模块独立性等特点,为自主机器人软件在决策层、管理层的开发提供元层支持,如:文献[29]提出了基于主体的认知模型来支持自主机器人软件的行为决策,文献[30]提出了面向自主协同服务机器人的软件体系结构 COROS 以封装机器人软构件,文献[31]提出了 VOMAS 架构以实现自主机器人在动态环境中的任务和角色转换.然而,现有工作缺乏对于机器人软件中多智能体协同机制的研究,尚未提供有效的多智能体架构和协调机制,用于加强机器人控制软件中不同构件之间的交互以支持机器人协调地实施行为.

2 自主机器人控制软件的抽象模型和软件架构

2.1 家庭服务机器人应用案例及任务场景

家庭服务机器人(family service robot,简称 FSR)运行在开放的房间中,可根据家庭成员的要求,帮助其完成特定的任务,如拿取水杯.由于环境的开放性,在实现任务的过程中,机器人的行为决策及实施受环境变化以及机器人动作执行结果的不确定性等多种因素的影响.针对于搜寻水杯的过程中,机器人根据其与杯的相对位置规划生成运动路径,以移动到可拿取水杯的位置,其可能面临以下两方面问题.

(1) 场景 1:地面摩擦系统的差异性导致机器人的行为执行存在偏差

然而,机器人的运动环境是不可知的,且存在大量难以被其控制模型事先规约的因素,如地面的摩擦系数.这些不可预知的因素容易导致预设的控制模型在任务执行过程中出现与预期结果不一致的偏差.这些偏差可表现为运动过程中感知内容的抖动现象.当抖动现象较为严重时,会导致机器人任务目标的识别出错,从而妨碍任务的可持续执行.为此,机器人需要在执行过程中进行伴随观察,根据行为的实际执行效果来适应性地调整规划,维持机器人行为的协调性以确保实现任务.

(2) 场景 2:抓取对象的位置被移动导致机器人在行为执行中丢失目标

如图 1 所示,机器人在搜寻水杯的过程中,其所在的环境可能会发生动态变化,如水杯的位置发生变化,从而导致机器人任务目标丢失.环境的动态变化,可能导致机器人任务目标出现紊乱.当机器人的原先任务执行策略失效时,其需要通过重规划进行校正.为此,机器人需要辅助以观察行为重新确定目标物体水杯位置,以指导后续行为决策的高效实施.

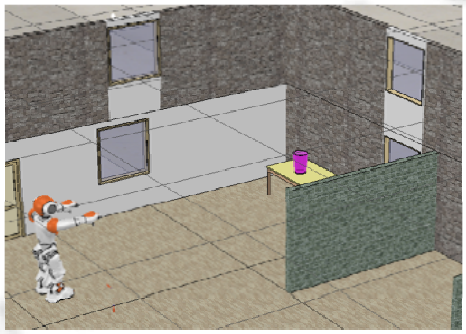


Fig.1 Object searchscenario of FSR in open environment

图 1 在开放环境下机器人搜寻物体的应用场景

2.2 自主机器人控制软件的MAS组织模型

自主机器人是一个极为复杂的信息物理系统.从结构和行为视角上看,CSAR 具有以下一组特点.

- 首先是构成分布性.CSAR 包含多个软构件,它们分别承担不同的职责和任务,分布在不同的计算节点

上运行,如任务规划构件负责决策机器人行为、传感构件负责控制传感器来感知环境信息等等;

- 其次是行为自主性.各个软构件独立运作、自主运行,如传感器构件根据其任务来自主地执行相应的行为来获取外部环境信息、分析感知数据、建立环境模型;
- 第三是管理发散性.由于软构件的构成分布性和行为自主性,CSAR 通常采用发散而非集中的方式来管理各个软构件;
- 第四是运行协同性.各个软构件间通过交互和协作来共同完成任务,应对各种预期和非预期的变化,从而实现自主机器人的有效决策和协调运行.

为了独立于机器人的物理硬件(如马达、机械臂等)和底层异构技术细节(如运行平台、编程语言等)来研究自主机器人的复杂行为及规律,本文针对 CSAR 的特点为其提供高层抽象的软件模型,以分析和揭示 CSAR 的结构和行为特征.本文针对 CSAR 的特点为其提供基于多智能体系统的高层抽象软件模型,以分析和揭示 CSAR 的结构和行为特征.多智能体系统是指由多个相对独立同时又相互作用的主体所构成的系统.在多智能体系统中:一方面,每个智能体都是自主的行为实体,封装了行为以及行为控制机制,可以在无须外部指导的情况下实施行为,因而具有相对的独立性了;另一方面,这些智能体并不是孤立的,它们之间存在各种关系,需要相互交互和协同,进而达成问题的求解.多智能体系统一个典型的特点是所有智能体是对等的,系统不存在具有全局控制能力的智能体,即可以完全控制其他智能体的行为以及环境状态的变迁,它也不能对系统的运行进行充分和完全的协调和控制.多智能体系统(multi-agent system,简称 MAS)的概念和思想可为复杂系统的研究提供抽象、分解和组织等技术手段,它用具有自主行为能力的“智能体”来抽象表示复杂系统中的行为实体,将复杂系统分解为由一组相对独立但存在交互的智能体所构成的多智能体系统,并通过诸如联盟、层次等形式来描述系统中不同智能体间的关系,进而来组织系统中的各个实体.

本质上,CSAR 可抽象为一组运行在特定环境(包括计算环境和物理环境)下相互协作的智能体所构成的多智能体系统,其抽象模型对应于一个五元偶$\langle AGENTS,ENV,TASKS,ROLES,BEHAVIORS\rangle$.

- $AGENTS=\{Agent_1,Agent_2,\ldots,Agent_n\},n\in Integer$.CSAR 由一组具有自主行为的智能体组成.每个智能体在系统中所起的作用、可执行的动作、具有的能力与智能体在系统中所承担的职责和扮演的角色密切相关;
- ENV.任何机器人及其构成的智能体都驻留在环境中并受环境的影响,CSAR 需要控制传感器来获取环境信息,通过对环境信息的分析建立起环境模型,并根据环境模型来规划和决策机器人的行为.$ENV=\{\chi_0,\ldots,\chi_k\}$表示环境状态集合,$\chi_k$指 t 时刻的环境状态,可由一组一阶谓词τ来表示,即:

$$\chi_k=\{\tau_1,\ldots,\tau_m\};$$

- $TASKS=\{T_0,\ldots,T_{p-1}\}$是自主机器人的包含 p 个子任务的任务集合,每一任务 $T_i(i\in[0,p])$对应于一个序偶$\langle\chi_k,\chi_*\rangle$,描述了任务 T_i 的目标是要从初始状态χ_k迁移至目标状态χ_*;
- $ROLES=\{R\text{-}PLANNER,R\text{-}ACTUATOR,R\text{-}SENSOR,R\text{-}DISPATCHER,R\text{-}EXECUTOR,\ldots\}$,定义了 CSAR 中的智能体可扮演的角色.例如:“*R-PLANNER*”负责规划行为;“*R-ACTUATOR*”负责执行任务行为;“*R-SENSOR*”负责执行观察行为;“*R-DISPATCHER*”负责分发和调度规划行为;“*R-EXECUTOR*”负责与机器人底层软硬件系统交互,将抽象的行为转化为底层的执行指令;
- $BAHAVIORS=BEHAVIOR_T\cup BEHAVIOR_O$,描述了自主机器人的两类行为:一类是任务行为(task-achievement behavior,简称 TB),其执行将有助于任务的达成并会改变机器人的运行环境;另一类是观察行为(observation behavior,简称 OB),其执行将获取环境信息但不会改变环境状态.

2.3 MaRSA架构模式

在基于 MAS 的 CSAR 模型中,不同智能体所承担的职责、扮演的角色、达成的任务等是不一样的,因此它们在软件架构中所处的地位和相互间的关系有所差别.在开放环境下,自主机器人的行为及其实施有其特殊性,具体表现为以下两个方面.

- 首先,任务行为的实施往往需要观察行为的支持,以帮助任务行为获得其执行所需的数据.例如在场景

一静态环境中,机器人需要以传感器信息(如水杯离机器人的距离和角度)作为输入来实现搜寻水杯的动作;

- 其次,在变化的环境中,观察行为的持续观察结果可为任务行为的在线决策提供支持.例如在场景二动态环境中,由于水杯位置可能变化,持续性的并发式视觉观察行为能够支持规划器适时做出在线决策,根据实时的感知结果规划产生新的运动轨迹.

针对上述特点,在分析和设计 CSAR 时,有必要显式地区分机器人的任务行为和观察行为,并通过加强二者之间的伴随执行,以支持行为的协调性和交互性.这就需要在软件架构层面合理地组织 CSAR 中的智能体、明确它们之间的关系以及加强不同智能体之间的协同,从而在行为层面为自主机器人的伴随行为机制设计奠定基础.

在面向智能体软件工程研究领域,人们通常将多智能体系统视为社会化的组织,借助组织理论以及社会组织模式的思想来指导软件架构的设计[32].Structure-in-5 是一类典型的组织结构模式,它是指一个组织可由 5 个承担不同职责的子组织构成.它们处于组织中的 3 个不同层次,分别承担不同的职责和义务:处于顶层的子组织称为“战略顶点(strategic apex)”,负责制定组织的总体战略;底层的子组织称为“执行核心(operational core)”,负责执行相关的活动以落实顶层制定的总体战略;中间层有 3 个不同的子组织结构,分别是“技术结构(techno-structure)”“辅助(support)”和“中间代理(middle agency)”,负责落实顶层制定的战略并为底层的执行和操作提供标准、服务和资源等,起到上传下达、监管下层的活动、辅助下层开展工作等功效.

基于 CSAR 的特点及其 MAS 组织模型,CSAR 软件架构可视为是一种特殊的 Structure-in-5 组织模式(如图 2 所示),将其称为 MaRSA.

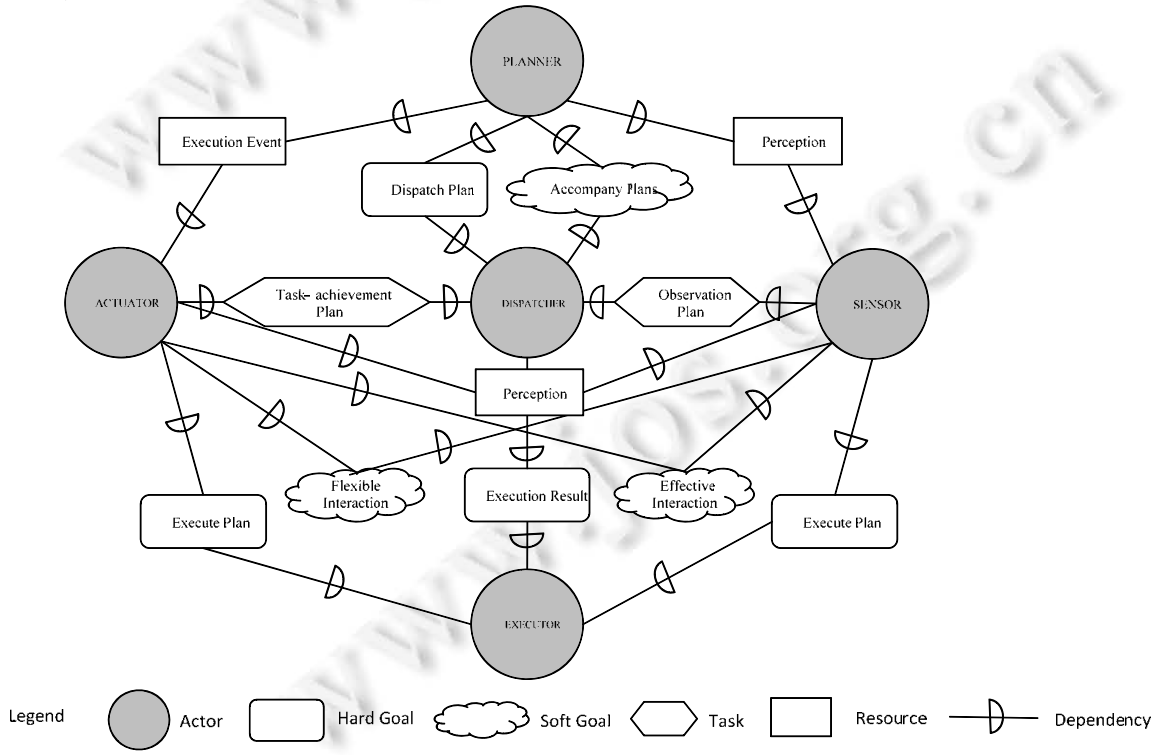


Fig.2 CSAR’s software architecture MaRSA based on structure-in-5

图 2 基于 structure-in-5 的 CSAR 架构 MaRSA

处于架构顶层的智能体为任务规划器 $Agent_{R\text{-}PLANNER}$,负责规划和决策自主机器人的行为;中间层包含 3 类智能体,分别是任务行为效应器智能体 $Agent_{R\text{-}ACTUATOR}$、传感器智能体 $Agent_{R\text{-}SENSOR}$、规划调度器智能体

$Agent_{R\text{-}DISPATCHER}$,它们负责落实顶层的行为规划,并为下层的动作执行提供支持,其中,$Agent_{R\text{-}DISPATCHER}$ 负责将顶层生成的规划分发到 $Agent_{R\text{-}PLANNER}$ 和 $Agent_{R\text{-}SENSOR}$ 去执行,$Agent_{R\text{-}ACTUATOR}$ 负责实例化和实施任务行为,$Agent_{R\text{-}SENSOR}$ 负责实例化和实施观察行为;处于底层的智能体称为执行器 $Agent_{R\text{-}EXECUTOR}$,负责执行机器人的控制程序,通过机器人的底层接口来控制机器人的运动.本文提出的基于 Structure-in-5 组织结构的多智能体系统架构,在智能体的功能抽象上具有显著的层次性,即高层抽象规划到底层感知和实施.然而从系统运行角度看,此多智能体系统架构呈发散式组织结构,即该架构没有统一的控制中心,所有智能体都是自主管理和运行的.每个智能体之间的运行不存在严格层次上的依赖关系.当某一功能组件智能体出现故障,系统的其他智能体可以通过自适应调整重新进行交互以实现系统功能,从而提高系统的健壮性.关于故障情况下智能体自适应调整,这一工作是本文的后续研究工作的重点.本文所提出的架构采用 Structure-in-5 组织的多智能体系统,通过功能抽象将一个复杂的自主机器人控制软件分解为多个独立自主运行和管理的智能体组件,每个智能体都定义了其主动交互行为,能够通过统一的交互协议和模式与其他智能体进行配合,极大降低了架构的组织复杂性,便于系统组件的组织和管理.

不同于一般的 Structure-in-5 结构,CSAR 软件架构 MaRA 在以下二方面做了进一步的增强,以支持机器人观察行为和任务行为之间的协调和有效交互.

(1) 观察行为与任务行为交互的协调性

在 MaRSA 的中间层,$Agent_{R\text{-}ACTUATOR}$ 依赖于 $Agent_{R\text{-}SENSOR}$ 所提供的感知信息,并依此来指导任务行为的实施.这意味着 $Agent_{R\text{-}ACTUATOR}$ 所执行的任务行为与 $Agent_{R\text{-}SENSOR}$ 所执行的感知行为之间存在伴随性,也即任务行为的执行需要观察行为的辅助,观察行为所获得的感知信息将为任务行为的实施提供支持.$Agent_{R\text{-}ACTUATOR}$ 行为与 $Agent_{R\text{-}SENSOR}$ 行为的伴随性意味着在机器人运行过程中,其观察行为是“按需的”和“有针对性的”,而非“盲目的”和“无所事事的”.

(2) 观察行为与任务行为交互的有效性

在 MaRSA 的顶层和中间层之间,当 $Agent_{R\text{-}SENSOR}$ 与 $Agent_{R\text{-}ACTUATOR}$ 采取持续性并发交互模式时,$Agent_{R\text{-}PLANNER}$ 可依赖 $Agent_{R\text{-}SENSOR}$ 所反馈的实时感知信息进行适应性在线决策.根据每一时刻获得的感知反馈,$Agent_{R\text{-}PLANNER}$ 做出当前环境状态下的理性规划.这种根据任务行为和观察行为之间的协调交互始终以任务为驱动,两类行为的协调交互结果最终能够促使任务目标有效达成.

概括起来,CSAR 的 MaRSA 架构具有以下特点.

- 首先,以智能体作为基本的软构件形态,每个智能体不仅封装了数据和动作,而且还具有自主的行为,从而确保软构件能够独立自主的运行,具有更好的封装性和可重用性;
- 其次,发散式管理.MaRSA 架构的智能体之间虽然存在层次性的组织关系,但是它们是对等的,不存在中心控制节点,整个系统采用发散性的管理方式,降低了系统组织和管理的复杂度;
- 第三,协调运行.MaRSA 架构的不同智能体之间基于依赖关系(包括任务、目标和资源依赖等)来进行交互和协同,如 $Agent_{R\text{-}ACTUATOR}$ 行为与 $Agent_{R\text{-}SENSOR}$ 行为之间的伴随性,从而使得整个系统能够在整体上以一种协调的方式来运行,提高了系统应对变化的能力.

3 基于 MaRSA 的伴随行为机制及决策算法

本节介绍自主机器人的伴随行为机制,以支持自主机器人的协调和有效行为实施.

3.1 伴随行为及伴随关系

3.1.1 伴随行为定义

在 CSAR 中,机器人的任务行为和观察行为之间在执行时序、数据交互、指令控制等方面存在交互,以实现相互间的协作,促进任务的完成、应对各种非预期的变化.本文将任务行为与观察行为间的上述关系称为伴随行为.在执行过程中两类行为所形成的伴随交互关系构成了本文伴随行为机制设计的基础(如图 3 所示).

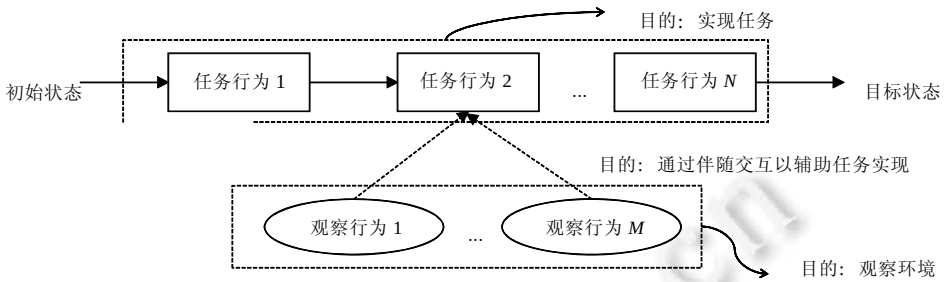


Fig.3 Accompanying interaction relationships between task behaviorsand observation behaviors

图 3 任务行为与观察行为的伴随交互关系

定义 1(任务行为). 任务行为是面向机器人任务实现的一组行为动作,其实施将使得环境状态发生变化,逐步迁移至目标状态,从而支持任务的实现.一个任务行为 p 定义为三元偶 $p=\langle\varpi,\delta,\varepsilon\rangle$,其中,$\varpi$代表该任务行为的前提条件,$\delta$代表其实施中需满足的不变条件,而$\varepsilon$则代表该任务行为成功执行后的预期效果.

例 1:以抓取水杯任务行为 *pickUp* 为例,其前提条件ϖ为:{(at cup cupboard),(at robot *roomA*)},其中,(at cup cupboard)表示当前环境状态下水杯位于橱柜中,(at robot *roomA*)表示当前机器人位于房间 *A* 中;不变条件δ定义为{collision free},预期效果ε定义为{(carry robot cup),not (at cup cupboard)}.

定义 2(观察行为). 观察行为指机器人为获得环境状态信息而实施的感知行为.一个观察行为定义为二元偶 $q=\langle\varphi,\xi\rangle$,其中,$\varphi$为触发观察的条件,$\xi$表示该观察行为所需观察的对象.

例 2:以机器人所实施的观察行为 *observe*(location?=kitchen)为例,其对应的触发条件φ=(location?=kitchen)为对应任务行为 *pickUp* 的预期效果,由此可推断出φ=not (at cup cupboard)和ξ=cupboard.

3.1.2 伴随行为关系

在开放环境下,任务行为和观察行为间的伴随交互具有以下 3 类关系:因果性、时序性和按需性.

(1) 因果性

因果伴随关系指任务行为与观察行为之间具有明显的“因-果”的特点.在开放环境中,机器人执行任务行为需要相应的环境状态作为输入,由此产生的环境观察需求构成“因”.观察行为则由于任务行为执行的需要被规划、调度和执行,其获取的感知输入作为“果”,从而为任务行为的执行和决策提供依据.

定义 3(因果伴随关系). 当任务行为 p 的前提条件 $p(\varpi)$、不变条件 $p(\delta)$或预期效果条件 $p(\varepsilon)$与观察行为 q 的待判断条件 $q(\varphi)$相一致,即 $p(\varpi)|p(\delta)|p(\varepsilon)=q(\varphi)$,则说明任务行为 p 和观察行为 q 之间存在因果伴随关系,记为

$$RELATION_{causal\text{-}effect}(p,q).$$

例 3:对于图 4 中的任务行为 p=*Move*(*roomA*,*roomB*)而言,其前提条件 $p(\varpi)$=(at robot *roomA*)表示该行为的执行前提是机器人初始位置为 *roomA*;其过程条件为(collision free),表示该行为在执行过程中需要保证机器人始终处于障碍物的安全距离;预期效果描述为(at robot *roomB*),即当该行为结束后,机器人应位于目的地 *roomB*.当对任务行为的这些条件进行分析以获知该任务行为的执行情况时,需要由相应的观察行为 q=*observe*⟨(at robot *roomA*),location⟩的支持.例如,需要判断 $p(\varpi)$=(at robot *roomA*)是否满足.为此,机器人应采取相应的观察行为予以观察和分析,如通过摄像头观察机器人当前所处环境的信息,进而来分析和判断机器人所处的房间位置.由任务行为的规约条件所产生的“因”,促使机器人采取针对性的观察行为,从而来感知相关的“果”,从而为任务行为的条件判断提供信息来源.

(2) 时序性

机器人任务行为的执行具有持续性的特点,即任务行为的实施是持续性过程而非瞬时过程.在任务行为的执行过程中可能产生不同的观察需求,观察行为需要满足一定的时序约束来辅助不同阶段的任务行为执行需求.例如:在任务行为执行过程中,观察行为需要进行持续性伴随,使得任务执行过程始终获得对环境状态的实

时感知,增强任务执行的感知性;当任务行为执行结束后,观察行为需要进行伴随观察,以获得任务行为的实际执行效果.

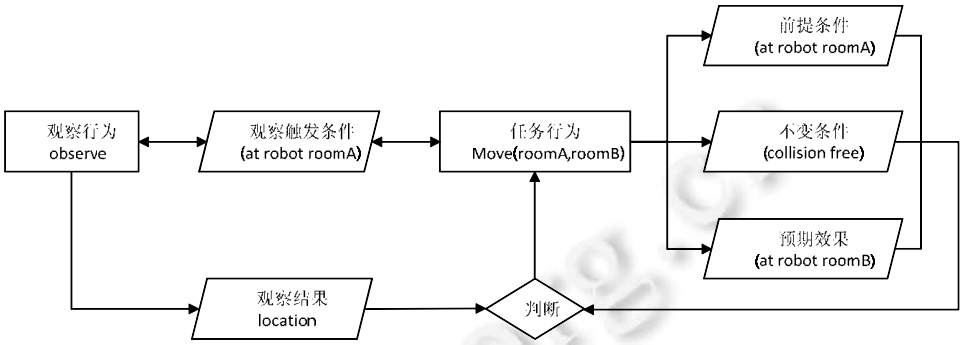


Fig.4 Causal-effect relationship between task behaviors and observation behaviors

图 4 任务行为与观察行为间的因果关系

定义 4(时序伴随关系). 对于任务行为 p 和观察行为 q 这两类持续性行为,任务行为的 3 种规约条件分别定义了该行为执行的 3 个时序特征:执行前(before)、执行中(during)和执行后(after).相应地,当观察行为 q 在不同的时间阶段与任务行为 p 在执行时序上存在相应的时序关系:执行前($p\prec q$)、执行中($p\vee q$)以及执行后($p\succ q$).则称任务行为 p 和观察行为 q 在执行时序上满足一定的时序关系,记为 $RELATION_{temporal}(p,q,\circ)$,其中,$\circ$代表$\prec$,$\succ$或$\vee$.

例 4:如图 5 所示,任务行为 $p=Move(roomA,roomB)$执行前($\prec$)以及执行后($\succ$),机器人需要执行相应观察行为 $q=observe\langle$(at robot *roomA*),location$\rangle$或者 $q=observe\langle$(at robot *roomB*),location$\rangle$来观察当前环境状态.这样的观察行为往往只执行一次,只需反馈当前环境状态,不需要进行持续性观察.相反,在该任务执行期间,机器人需要采取持续性的观察行为对环境状态进行感知,实时更新环境中相关状态的变化情况,以支持在实施过程中对任务行为的安全条件进行持续性、实时性的判断.

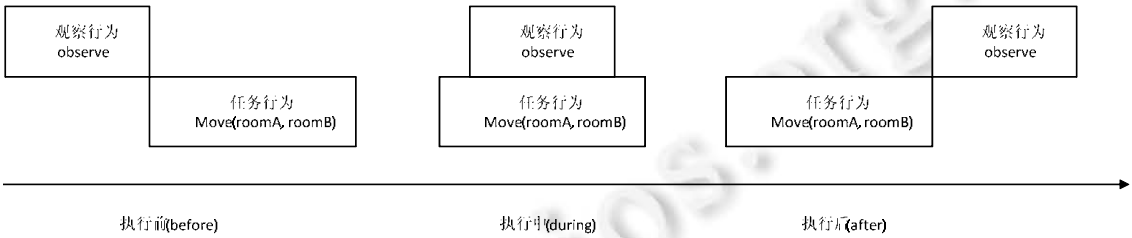


Fig.5 Temporal relationship between taskbehaviorsand observation behaviors

图 5 任务行为与观察行为的时序关系

(3) 按需性

观察行为的观察目标和伴随反馈通常是按需执行的,针对任务行为执行过程中所需的输入需求,机器人需要实施不同的观察行为.因此,任务行为的执行输入与观察行为的感知结果间存在语义方面的相关性,即观察行为所产生的语义结果将作为任务行为的输入信息,我们将这种伴随关系称之为按需伴随.

定义 5(按需伴随关系). 对于任务行为 p,当其蕴含的观察需求 $p(\varpi)_\xi$,$p(\delta)_\xi$或 $p(\varepsilon)_\xi$能够被观察行为 q 的观察结果 $q(\xi)$所满足,即 $p(\varpi)_\xi\vee p(\delta)_\xi\vee p(\varepsilon)_\xi\mapsto q(\xi)$,则称任务行为 p 与观察行为 q 间存在按需伴随关系,记为

$$RELATION_{on\text{-}demand}(p,q).$$

例 5:通过建立起与任务相关的环境语义模型,任务行为与观察行为之间可以实现有针对性、按需性的伴随,具体体现在:当任务行为中某一条件需要进行判断时,机器人会根据该条件所隐含的具体语义信息,实施相应的观察行为,从而获得相应的语义结果.当任务行为 $p=Move(roomA,roomB)$中前提条件 $p(\varpi)$=(at robot *roomA*)需要

被判断是否满足时,机器人需要去观察机器人当前所处位置,而这种对位置环境(*roomA*)的观察往往可以通过观察是否存在相应的标志性物体(obj)来实现.在此情况下,观察行为 *q*=*observe*⟨(at robot *roomA*),obj⟩与该任务行为构成按需伴随关系.

3.2 伴随行为的自主决策算法

基于自主机器人的上述软件架构和伴随行为机制,自主机器人控制软件需要解决伴随行为的自主决策及实施问题.本文介绍伴随行为的决策算法DAAB(decision algorithm of accompanying behaviors),它采用分步规划和动态决策的思想.

- 分步规划:第 1 步为全局任务规划,即根据机器人的任务,通过经典任务规划算法,规划出机器人的任务行为序列;第 2 步为伴随行为规划,针对每一个执行任务,为其行为规划出所需的观察行为以及它们之间的交互.分步规划将两类行为的规划过程交由不同的规划器智能体执行,该方法可降低规划过程的复杂度,并提高行为决策的灵活性;
- 动态决策:观察行为的规划过程与当前任务行为的执行状态相关,观察行为与任务行为通过伴随行为约束建立运行时的动态关系,而非在任务行为规划和执行之前事先制定.这种动态决策的方式实现了观察与任务行为之间的松耦合伴随,有助于提高伴随行为实施的灵活性.整个自主决策过程如图 6 所示.伴随行为决策算法包含两部分:任务行为决策和伴随行为决策.

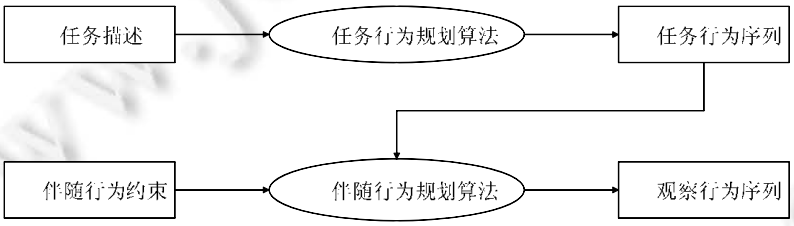


Fig.6 Two-step decision-making process of accompanying behaviors

图 6 伴随行为的二阶段决策过程

(1) 任务行为决策算法

根据任务描述,任务行为的决策算法可规划出一组有序的任务行为序列,该任务序列在理想状态下能够实现机器人由初始状态到达目标状态,从而实现任务.算法 1 中,任务行为规划算法 $T(X_0,X_*)$的输入为任务描述Π,其定义了机器人任务的问题域和规划域模型.算法首先遍历所有可用的任务行为 $p=\langle\varpi,\delta,\varepsilon\rangle$,若该任务行为的前提条件在当前初始状态下可满足,即 $X_0\mapsto p(\varpi)$,则将任务行为 p 加入可行序列集 P 中.从集合 P 中逐个遍历每一可用行为 p_i,采取一步预测方法,预测该行为在初始状态 X_0 下执行后的预期效果 X_0',即 $X_0'=X_0\cup p_i(\varepsilon_i)$.其中,机器人任务行为的预期效果可以通过其行为模型进行预测.

算法 1. 任务行为规划算法 $T(X_0,X_*)$.

输入:任务描述$\Pi=(X_0,X_*,P)$;

输出:一组有序任务行为序列 $P=[p_1,\ldots,p_n]$.

1. 当初始状态与目标状态一致,即 $X_0=X_*$,任务行为序列为空集 $P=\varnothing$;
2. 当某一任务行为 $p=\langle\varpi,\delta,\varepsilon\rangle$在初始状态 X_0 可执行,即 $X_0\mapsto p(\varpi)$,则 $p\in P$;
3. 遍历可行序列集 P 中每一个任务行为 $p_i=\langle\varpi_i,\delta_i,\varepsilon_i\rangle$,预测其在初始状态 X_0 下执行后的预期效果,即得:
$$X_0'=X_0\cup p_i(\varepsilon_i);$$
4. 取 $X_0=X_0'$ 继续执行第 3 步,添加新的任务行为至可行序列集 P 中,直至 $X_0=X_*$;
5. 输出一组有序任务行为序列 $P=[p_1,\ldots,p_n]$.

图 7 举例展示了机器人的任务行为 *Move*(*roomA*,*roomB*)的行为模型,该模型定义了行为执行的前提条件、

不变条件以及预期效果(at robot *roomB*).在每一步的任务行为决策时,算法通过轮询当前每个可用任务行为模型中的预期效果 X_0',同时依次取 $X_0 = X_0'$,并继续在当前所有任务行为中选取当前状态 X_0 下可行任务行为 p_j,并将其加入可行序列集 P 中.重复上述过程,直至达成目标状态,即 $X_0=X_*$,此时得到的可行序列集 P 即为一组有序任务行为.

```
# 任务行为Move(roomA,roomB)的行为模型
 (: durative action Move
          :parameters (roomA roomB)
          :duration (= ?duration 10)
         :condition (and
                  (at start (robot_at robot roomA)))
          :guard (and (free collision))
         :effect (and # 行为实施的预期效果
                 (at end (visited roomB))
                  (at start (not (robot_at robot roomA)))
                 (at end (robot_at roomB)))
)
```

Fig.7 Action model of task behavior *Move*(*roomA*,*roomB*)

图 7 机器人任务行为 *Move*(*roomA*,*roomB*)的行为模型

(2) 伴随行为决策算法

针对任务规划所决策产生的每一个任务行为,伴随行为规划算法根据伴随关系约束,规划产生对应的观察行为以及观察行为与任务行为间的交互.算法 2 以算法 1 所产生的任务行为序列 $P=[p_1,\ldots,p_n]$以及前述伴随关系约束为输入,输出一组符合伴随交互关系的任务-观察伴随行为.针对每一任务行为 p 的执行,该决策算法会依据伴随关系的约束条件产生相应的观察行为,以辅助当前任务行为的执行.当任务行为 p_i 执行之前,算法会依据其前提条件 $p(\varpi_i)$在观察行为库寻找合适的观察行为 q,使得该观察行为与任务行为符合伴随因果关系 $p(\varpi_i)=q(\varphi)$;同时,观察行为的观察结果符合该任务行为的观察需求 $p(\varpi_i)\mapsto q(\xi)$.此时,观察行为 q 与该任务行为构成前序时序关系 $p_i\succ q$,观察行为在任务行为执行前进行环境观察.当任务行为 p_i 执行过程中,若存在观察行为 q 能够对任务行为的过程条件进行观察,并且观察结果满足其观察需求,即 $p(\delta_i)=q(\varphi),p(\delta_i)\mapsto q(\xi)$,则观察行为 q 与任务行为 p_i 构成并发时序关系 $p_i\vee q$,观察行为在任务行为执行过程中进行观察活动.同样,在任务行为 p_i 执行结束后,若观察行为 q 满足 $p(\varepsilon_i)=q(\varphi),p(\varepsilon_i)\mapsto q(\xi)$,则观察行为 q 与任务行为 p_i 构成后序时序关系 $p_i\prec q$,观察行为在任务行为执行结束后观察.

算法 2. 伴随行为决策算法 $A(P,RELATION_{causal\text{-}effect},RELATION_{temporal},RELATION_{on\text{-}demand})$.

输入:任务行为序列 $P=[p_1,\ldots,p_n]$,$RELATION_{causal\text{-}effect}$,$RELATION_{temporal}$ 和 $RELATION_{on\text{-}demand}$;

输出:一组伴随执行的任务/观察行为序列 $O=[p_1,q_1,p_2,q_2,\ldots]$.

1. 任务规划器基于任务描述 $\Pi=(X_0,X_*,P)$由算法 1 的 $T(X_0,X_*)$产生一组任务行为:$X_0+X_*\rightarrow P=[p_1,\ldots,p_n]$;
2. 对于每一子任务行为 $p_i=\langle\varpi_i,\delta_i,\varepsilon_i\rangle\in P$ 的执行,在执行前,当存在观察行为 $q=\langle\varphi,\xi\rangle$使得 $p(\varpi_i)=q(\varphi)$, $p(\varpi_i)\mapsto q(\xi)$且 $p_i\succ q$,则调度观察行为 q 在任务行为 p_i 前执行;
3. 在执行过程中,当存在观察行为 $q=\langle\varphi,\xi\rangle$使得 $p(\delta_i)=q(\varphi),p(\delta_i)\mapsto q(\xi)$且 $p_i\vee q$,则调度观察行为 q 在任务行为 p_i 执行过程中并发执行;
4. 在执行结束后,当存在观察行为 $q=\langle\varphi,\xi\rangle$使得 $p(\varepsilon_i)=q(\varphi),p(\varepsilon_i)\mapsto q(\xi)$且 $p_i\prec q$,则调度观察行为 q 在任务行为 p_i 执行结束后执行;
5. 当无满足伴随关系的观察行为时,继续执行下一任务行为 p_{i+1},直至任务结束.

4 实验分析

本节介绍论文所开展的实验,分别针对环境变化和行为执行结果不确定两种应用场景,对比分析基于MaRSA 架构和伴随行为机制的 DAAB 算法所生成的行为规划在达成任务实现方面的可行性和高效性.

4.1 实验设计

针对第 2.1 节所描述的开放环境下自主机器人任务实现所面临的二类典型挑战,分别在基于仿真的机器人实验环境和基于真实机器人的实验环境中开展实验.基于仿真环境的实验主要用于模拟机器人行为执行结果的不确定性,通过设置地板的摩擦系数,使得机器人在移动过程中产生运动偏差,以此来表征机器人行为执行的不确定性.基于真实机器人的实验主要研究当机器人的外部环境发生变化时,如人为地移动水杯的位置,机器人如何通过伴随行为来快速地定位目标并进行重规划,以高效率地完成任务.

4.1.1 基于固定目标搜索的仿真实验

为了验证所提出的方案对机器人任务执行过程中协调性的支持,本文于仿真环境下进行目标物体位置不变、运动环境存在大量干扰因素的目标跟踪实验.基于 V-REP 仿真器,使用 NAO 机器人模型、水杯模型等构筑仿真案例场景.在仿真场景中(如图 8 所示),机器人需要寻找固定位置的目标水杯.然而,由于地面摩擦系数较小以及机器人运动系统的固有偏差,机器人在该环境下的移动存在较大的方向和距离偏差,从而导致感知及结果的抖动现象出现.这种行为执行上的不确定性导致机器人在运动过程中容易偏离规划轨迹,视觉系统容易丢失目标水杯,从而导致已决策的任务规划不可行.在此仿真实验中,根据机器人的视野调整与目标水杯之间的对应关系,我们通过机器人最终能否顺利到达目标水杯的位置,来评估算法所决策生成规划对机器人动作的协调性.仿真实验中地面摩擦参数设置见表 1.

Fig.8 Simulated scenario for uncertainties in robot plan execution

图 8 针对行为执行结果不确定的仿真实验场景

Table 1 Parameter settings of contacting floor in the simulator

表 1 仿真环境地面摩擦参数设置

Bullet 属性	Friction		
	1.0000e+00		
ODE 属性	Friction	Maximum contacts	Soft ERP
	1.0000e+00	64	2.0000e+00
Vortex 属性	Restitution threshold	Compliance	Damping
	5.0000e-01	1.0000e-08	1.0000e+07
Newton 属性	Static friction	Kinetic friction	
	1.0000e+00	1.0000e+00	

4.1.2 基于动态目标搜索的真实机器人实验

为了验证所提出的方案对机器人生成任务决策的高效性,本文于实际环境中进行目标物体位置变化、无明

显运动环境干扰因素的目标跟踪实验.基于真实机器人的实验建立在双足人形机器人NAO硬件平台(操作系统版本 1.14)之上,在室内环境中设置真实的水杯来构造真实的实验环境(如图 9 所示).在此实验中,NAO 机器人运行在开放的室内环境中,在实现任务的过程中,待抓取机器人的位置会发生变化.在此情况下,机器人所决策生成的规划变得不可行,机器人需要通过观察行为来搜寻并确定目标对象的位置,依此来指导机器人的行为决策及执行,并进行高效的重规划过程.

Fig.9 Indoor scenario based on real robot to take the target with changed position

图 9 基于真实机器人、针对拿取对象位置动态变化的实验场景

4.2 实验结果及其分析

上述仿真和真实场景通过以下数据评估本文 DAAB 算法所生成的行为规划在达成任务实现方面的可行性和高效性.

(1) 世界坐标系中机器人距目标物体的距离 CoordinateDistance 与机器人视野中目标物体与视野中心的距离 ObjectVisionDistance.通过这两种距离的变化情况,得以判断所决策的行为能否以及多大程度上有助于任务的实现,评估决策算法所生成的规划在达成任务实现方面的可行性;

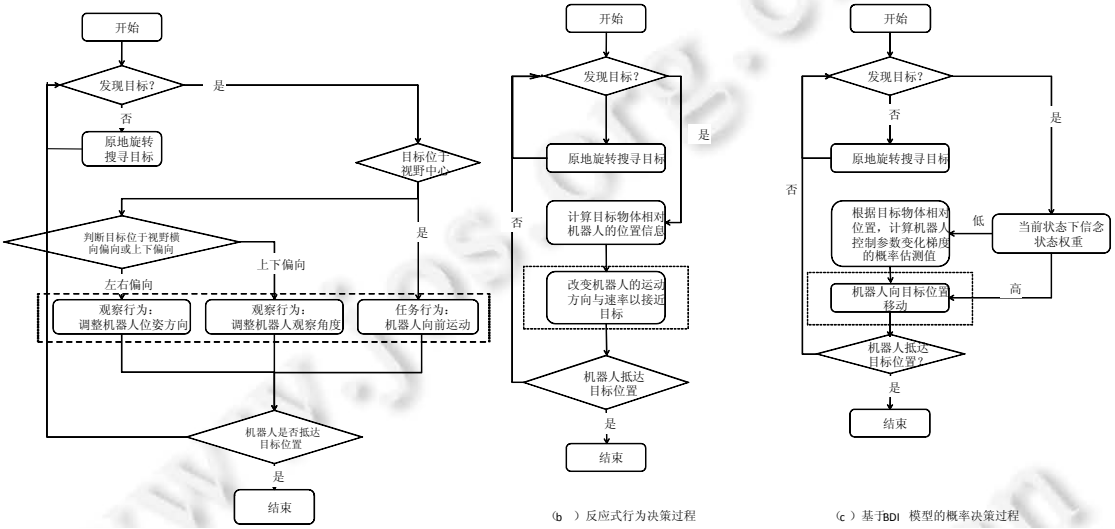
(2) 机器人从初始位置到目标位置所用时间,即完成任务所需的时间 TaskTime,以及任务实现过程中从任务重规划到重新执行阶段所用时间 AdjustmentTime.评估决策算法所生成的规划在任务实现方面的高效性.

4.2.1 实验对比算法的选择

实验选取了在自主机器人行为决策领域的二类主流算法进行对比和分析:反应式行为决策算法[7]和基于 BDI 模型的概率决策算法[23].这两种决策算法中,对机器人的观察行为和执行行为没有进行显式区分和建模,在决策时也没有考虑这二类行为之间的伴随.实验通过与这两种决策算法进行对比,评估 DAAB 算法所生成的规划在支持任务实现方面的可行性和高效性.本文基于文献[23]复现了基于 BDI 模型的概率决策方法,其主要算法思想(文献[23]中算法 4)是:通过对机器人观察结果的概率估计来更新机器人信念状态,并根据当前信念状态选择下一步行为.基于此算法思想,本文复现了如图 10(c)所示的 BDI 式机器人决策算法.通过反馈信息,计算机器人当前行为执行的信念值.结合这一信念值与意图行为序列,生成一组行为以缩短机器人与目标物体之间的距离.同样,基于文献[7]中提出的基于反应式方法,本文在复现算法中考虑直接根据反馈的感知信息输出相应的控制行为,无更高层次的规划和推理过程.如图 10(b)所示,机器人的控制系统通过上层决定的控制结果来驱动下层控制行为的执行,以条件触发的形式进行反应式的行为调整.

图 10 展示了这 3 种决策方法的规划过程及产生的规划结果.伴随行为决策过程中,机器人根据目标在视野中相对位置进行决策,使其位于视野中心,DAAB 算法最终产生的为观察行为和任务行为伴随执行的伴随行为

规划.反应式决策过程中,机器人根据目标与机器人的相对位置直接生成控制参数来驱动机器人运动,以快速应对目标与机器人的相对位置变化.基于 BDI 模型的概率决策过程根据目标与机器人的相对位置来获取机器人对任务执行的信念值,并通过概率模型来表征机器人的控制参数变化期望与任务实现的意图,从而在保证任务执行效率的同时提高任务执行的稳定性.反应式行为决策算法和基于 BDI 模型的概率决策算法由于没有显式区分机器人的观察行为和任务行为,因此产生的规划为传统的任务规划.本文提供 3 种对比算法的实现源码,以供读者进一步理解上述算法思想(https://github.com/yangshuo11/Accompanying-Behavior-Planning-Algorithms-for-Structure-in-5-Architecture.git).



(a) 伴随行为决策过程 (b) 反应式行为决策过程 (c) 基于 BDI 模型的决策过程

Fig.10 Accompanying behavior, reactive behavior and BDI-based probabilistic planning processes

图 10 伴随行为决策算法、反应式行为决策算法和基于 BDI 模型的概率决策算法的决策过程和规划结果

4.2.2 仿真场景下的实验数据及分析

仿真场景下,机器人距离目标物体的距离以及机器人对目标物体的观察状态如图 11 和图 12 所示.

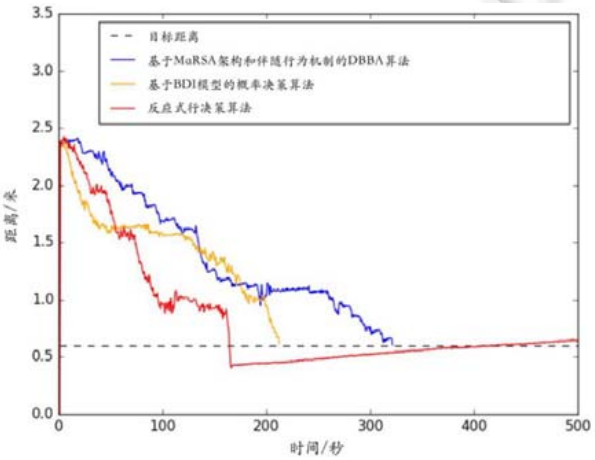


Fig.11 Global distance between robot and target in the simulated scenario

图 11 仿真实验中在世界坐标系下机器人与目标物体间的距离

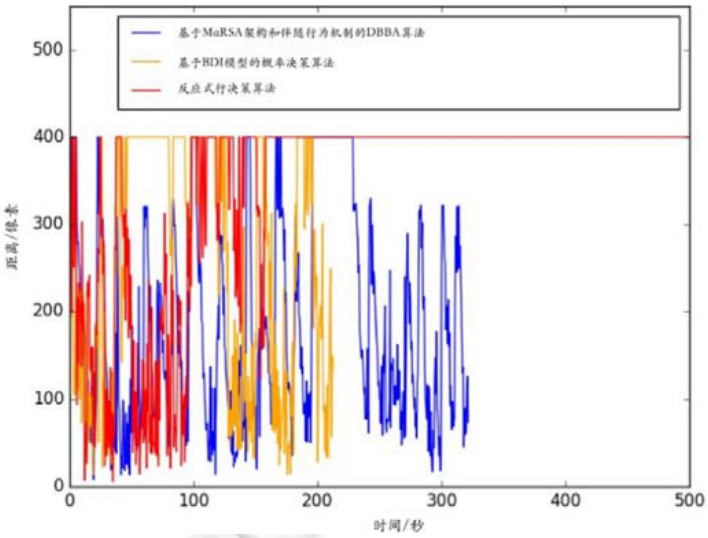


Fig.12 Distance between target and robot vision center in the simulated scenario

图 12 仿真实验中,在机器人视野中目标物体与视野中心的距离

从图 11 中可知:在反应式决策方法中,机器人在 180s 摔倒从而任务执行失败;在基于 BDI 的概率规划方法中,机器人在 210s 完成给定任务抵达水杯附近;基于 MaSRA 的 DBBA 算法的完成任务时间为 320s.因此,本文提出的 DAAB 算法以及 BDI 概率决策算法所产生的规划在实现任务执行过程中具有较高的可行性,而反应式决策算法所产生规划可行性相对较差.下面具体分析其原因.

反应式的决策过程中,机器人与目标物体的距离快速缩小,但最终出现断崖式下跌,即机器人摔倒.基于 BDI 的方法使得机器人与目标物体的距离先是迅速缩小到 1.5m 左右的位置处,之后呈平缓变化并出现快速下跌过程.而本文提出的 DAAB 算法使得机器人与目标物体的距离长期呈现一个平稳变化的过程.结合图 12 可以发现:当机器人能够持续发现目标物体时(0s~100s 内),由于反应快速,反应式决策方法能够根据目标物体与机器人的相对位置支撑机器人直接转向并持续运动.BDI 决策方法与反应式决策方法类似.而本文提出的 DAAB 算法则由于其需要长期保持目标物体位于视野中心,因此会花费大量时间开销在伴随行为间的切换过程,效率较反应式的决策过程相对较低.就反应式与 BDI 方法而言,当机器人接近目标物体时,目标物体频繁脱离机器人视野(在图 12 中表现为距离值达到 400 pixel),而基于伴随行为的 DAAB 算法仍能长期保持观察到目标物体.因此出现了图 11 中,反应式方法距离的断崖式下跌以及 BDI 方法的距离变化明显减缓而基于伴随行为的方法距离变化仍较为稳定.为了分析反应式决策算法所产生的规划的低可行性原因,本文将结合图 13 反应式决策方法过程进一步阐述.

如图 13 所示,对于反应式决策方法,由于机器人运动过程中的抖动,目标物体在机器人视野中总是出现于视野边缘,机器人容易丢失目标物体.特别当机器人抵达目标物体附近时,运动抖动对机器人观察视野的影响将被放大.因此出现了如图 12 所示的现象:在 100s 后,机器人抵达目标物体附近时,目标物体在机器人视野中出现频繁丢失,机器人需要不断原地旋转来搜寻目标物体.由于极大的转动惯量及光滑地面的影响,导致机器人摔倒和任务失败.同时,我们进一步分析采用伴随行为的 DAAB 算法与基于 BDI 的概率决策方法所产生规划的高可行性的原因.对于 DAAB 算法,由于机器人需要通过频繁的观察行为与任务行为间的调度,其能保证目标物体处于机器人视野中心.对于 BDI 的概率规划方法,机器人总是能够根据前段时间的物体对机器人的相对信息赋予任务执行以信念值.当信念值较高时,机器人将以较高的速率运动.当机器人运动受阻时,该模型将暂时降低任务执行的信念值,以任务实现意图为驱动保证机器人任务的顺利执行,如 50s~90s 期间的平缓运动.

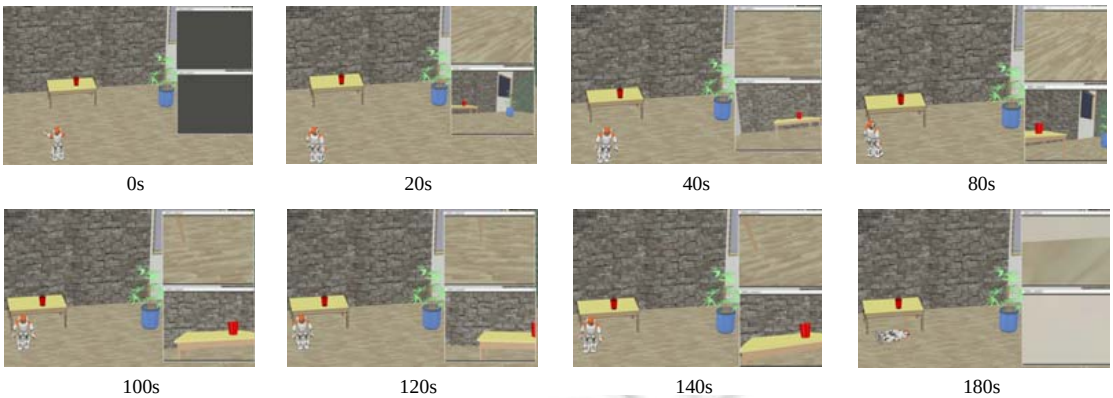


Fig.13 Robot movement and observation results at different time in reactive behavior planning process

图 13 反应式决策过程中,不同时刻的 NAO 机器人的运动及观察结果

4.2.3 真实场景下的实验数据及分析

图 14 展示了机器人在 DAAB 算法所产生的伴随行为规划下的机器人运动状态.

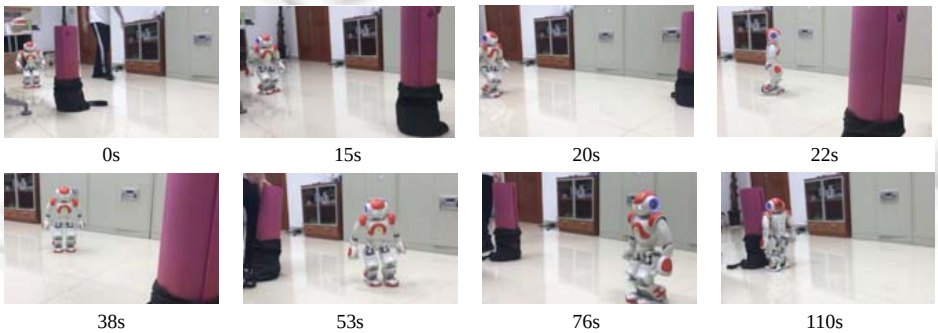


Fig.14 NAO robot movement results of different time in accompanying behavior planning process

图 14 伴随行为决策过程中,不同时刻 NAO 机器人的运动状态

整个运动过程分为 4 个阶段.

- 阶段 1:当机器人运动一段距离后,在机器视野范围内直线后移目标物体(15s 处);
- 阶段 2:在机器做出进一步运动后,快速将目标物体往某一方向进行偏移(20s 处);
- 阶段 3:当机器人找到目标物体并进一步运动一段时间后,快速将目标物体往另一个方向进行偏移(22s 处);
- 阶段 4:当机器人再次找到目标物体并进一步运动一段时间后,快速将目标物体移置到机器人背后(53s 处).

在阶段 1 与阶段 2 的过程中,DAAB 算法的规划总是能够保持目标物在机器人视野中,并持续面向目标物体运动.对于阶段 3,由于目标物体的移动速率较快且偏移幅度相对于阶段 2 而言是其偏移量的两倍,机器人不能快速地跟踪目标物体的变化,因此出现了第 1 次较长的目标物体丢失状态,在此情况下,机器人并未丢失目标物体信息,而是通过记录的历史信息按目标物体的偏移方向持续转动,最终能尽可能快地发现目标物体所在.对于阶段 4,由于人为的控制导致了目标物体位置相对于机器人而言出现了剧烈的变化,因此出现了第 2 次较长的目标物体丢失状态.这一次的目标物体丢失状态由于变化较为明显,并且机器人完全丢失了目标物体信息,从而触发了搜寻目标物体状态.在重新发现目标物体后,机器人快速调整自身状态并向目标物体移动.由此,相比其他两种决策算法,DAAB 算法所决策的行为所生成的规划在根据变动的目标调整机器人观察行为过程中具有

更好的可行性.

表 2 展示了真实场景下机器人在任务执行各阶段的时间开销及任务执行的总时间开销.首先,从任务执行的总时间开销来看,基于 MaSRA 的 DAAB 算法的时间开销远低于其余两种方法.其原因在于:反应式决策算法中,机器人的视野调整会出现大量扰动,目标物体长期处于机器人的视野边缘.在这一情况下,当目标物体出现即使如阶段 1 的较小幅度变化,机器人也会由于对环境观察过程的延时导致机器人完全丢失目标物体信息,从而触发大量的搜寻过程.因此当环境动态变化时,反应式的决策方法由于花费大量时间用于目标物体搜寻过程,因此其时间开销远大于基于伴随行为的决策方法.基于 BDI 的概率决策方法同样由于短期记忆的失效附加以目标物体处于视野边缘,因此机器人多次丢失目标物体信息而触发大量搜寻过程.而基于 MaSRA 的 DAAB 算法受益于其灵活的观察行为与任务行为的调度,机器人总是能快速发现目标物体,从而避免了大量的搜寻过程.特别的,就 BDI 而言,当目标物体丢失时,由于其固有运动期望的影响,BDI 的搜寻过程时间开销较反应式更长.基于此分析,虽然 BDI 在静态环境中,任务执行效率最高,但是当环境复杂多变时,BDI 模型难以适应环境变化.而基于 MaSRA 的 DAAB 算法由于其灵活的行为调度,在静态环境中运动效率不高,但当机器人处于动态变化的环境中时,其就任务执行效率的优势将得以展现.反应式的方法就任务执行效率介于其二者之间.由此说明,基于 MaSRA 的 DAAB 算法所产生的规划在机器人任务动态变化时将具有更高的高效性.

Table 2 Time costs of 4 execution phases and the total time of task execution in real-world environment

表 2 真实机器人 4 个任务执行阶段的时间开销及任务执行总时间开销

任务执行阶段 / 决策方法	阶段 1(s)	阶段 2(s)	阶段 3(s)	阶段 4(s)	任务执行总时间开销(s)
基于 MaSRA 的伴随行为决策方法	18	20	16	56	110
基于 BDI 的概率决策方法	23	47	33s	57	160
基于反应式的决策方法	26	30	31	60	147

5 结束语和未来工作

本文针对开放环境下自主机器人任务执行所面临的行为决策和实施的挑战,根据自主机器人控制软件的构成分布性、行为自主性、管理发散性、运行协同性等特点,开展了自主机器人控制软件的架构、伴随行为机制和决策算法的研究,取得了 3 个方面的研究成果和贡献.

(1) 采用组织理论的思想,提出了基于 Structure-in-5 组织架构的自主机器人控制软件架构 MaRSA.该架构显式抽象和封装了自主机器人的任务行为和观察行为,实现了自主机器人控制软件的感知和执行功能相分离,并能够促进任务行为实施构件与观察行为实施构件间的灵活交互和协同;

(2) 提出了自主机器人的伴随行为机制,从时序、按需、因果这 3 个方面描述和规约了任务行为和观察行为间的伴随交互关系,从而为自主机器人的有效和协调运行提供了核心运行机制支持;

(3) 设计了基于二阶段、动态规划思想的伴随行为自主决策算法,将伴随行为的决策延后到任务行为实施过程中进行,降低了伴随行为决策的复杂度,提高了自主机器人伴随行为决策和实施的灵活性.

本文分别在仿真环境和实际机器人环境下设计了对比性实验,结果表明:与主流的反应式行为决策算法和 BDI 式概率决策算法相比较,基于 MaRSA 和伴随行为机制的 DAAB 算法所生成的伴随行为规划在开放环境下具有可行性和更高效的执行效率.

未来将围绕以下几个方面开展研究工作:(1) 面向多任务多目标的行为决策机制及算法的设计,支持多目标任务决策的优化;(2) 自主机器人的支撑软件平台,支持自主机器人控制软件的开发、部署和运行;(3) 结合具体的应用场景,开展基于实际机器人的应用开发和演示.

References:

[1] Ma XX, Liu XZ, Xie B, Yu P, Zhang T, Bu L, Li XD. Software development methods: review and outlook. Ruan Jian Xue Bao/Journal of Software, 2019,30(1):3−21 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5650.htm

[2] Bu L. Formal verification: From hybrid system to CPS. Communications of the CCF, 2013,9(7):23−30 (in Chinese).

[3] Mao XJ. Autonomous robot software technology. Communications of the CCF, 2016,12(7):60−65 (in Chinese).

[4] Skrzypczynski P. Multi-Agent software architecture for autonomous robots: A practical approach. Management and Production Engineering Review, 2010,1(4):55−66.

[5] Hajduk M, Sukop M, Haun M. Cognitive Multi-Agent Systems: Structures, Strategies and Applications to Mobile Robotics and Robosoccer. Springer-Verlag, 2019.

[6] Bensalem S, Ingrand F, Sifakis J. Autonomous robot software design challenge. In: Proc. of the 6th IARPIEEE/RASEURON Joint Workshop on Technical Challenge for Dependable Robots in Human Environments. 2008.

[7] Brooks R. A robust layered control system for a mobile robot. IEEE Journal on Robotics and Automation, 1986,2(1):14−23.

[8] Aldrich J, Garlan D, Kästner C, Le Goues C, Mohseni-Kabir A, Ruchkin I, Samuel S, Schmerl B, Steven Timperley C, Veloso M, Voysey I, Biswas J, Guha A, Holtz J, Camara J, Pooyan J. Model-Based adaptation for robotics software. IEEE Software, 2019, 36(2):83−90.

[9] Ziafati P. Programming autonomous robots using agent programming languages. In: Proc. of the 2013 Int'l Conf. on Autonomous Agents and Multi-Agent Systems. 2013. 1463−1464.

[10] Benaissa S, Moutaouakkil F, Medromi H. New multi-agent's control architecture for the autonomous mobile robots. Int'l Review on Computers and Software, 2011,6(4):477−480.

[11] Tzafestas SG. Mobile robot control and navigation: A global overview. Journal of Intelligent & Robotic Systems, 2018,91(1): 35−58.

[12] García S, Menghi C, Pelliccione P, Berger T, Wohlrab R. An architecture for decentralized, collaborative, and autonomous robots. In: Proc. of the 2018 IEEE Int'l Conf. on Software Architecture (ICSA). IEEE, 2018..

[13] Ferrati M, Settimi A, Muratore L, Alberto C, Rocchi A, Mingo Hoffman E, Pavan C, Kanoulas D, Tsagarakis N, Natale L, Pallottino L. The Walk-Man robot software architecture. Frontiers in Robotics and AI, 2016,3:25.

[14] Hong H, Jung H, Park K, Yoon Y, Ha S. Service-Oriented robot software development framework for distributed heterogeneous platforms. In: Proc. of the 17th Int'l Conf. on Autonomous Agents and Multi-Agent Systems, Int'l Foundation for Autonomous Agents and Multi-Agent Systems. 2018. 1959−1961.

[15] Penin B, Giordano PR, Chaumette F. Vision-Based reactive planning for aggressive target tracking while avoiding collisions and occlusions. IEEE Robotics and Automation Letters, 2018,3(4):3725−3732.

[16] Arkin RC, Balch T. AuRA: Principles and practice in review. Journal of Experimental & Theoretical Artificial Intelligence, 1997, 9(2-3):175−189.

[17] Proetzsch M, Luksch T, Berns K. Development of complex robotic systems using the behavior-based control architecture iB2C. Robotics and Autonomous Systems, 2010,58(1):46−67.

[18] Georgeff MP, Lansky AL. Reactive reasoning and planning. In: Proc. of the 6th National Conf. on Artificial Intelligence, 1987. 677−682.

[19] Howden N, Rönnquist R, Hodgson A, Lucas A. JACK intelligent agents-summary of an agent infrastructure. In: Proc. of the 5th Int'l Conf. on Autonomous Agents. 2001.

[20] Braubach L, Lamersdorf W, Pokahr A. Jadex: Implementing a BDI-Infrastructure for JADE Agents. EXP–in Search of Innovation 2003,3(3):76−85.

[21] Smith T, Simmons R. Probabilistic planning for robotic exploration [Ph.D Thesis]. Carnegie Mellon University, the Robotics Institute, 2007.

[22] Meneguzzi F, De Silva L. Planning in BDI agents: A survey of the integration of planning algorithms and agent reasoning. The Knowledge Engineering Review, 2015,30(1):1−44.

[23] Vahrenkamp N, Wachter M, Krohnert M, Welke K, Asfour T. The robot software framework armarx. IT-Information Technology, 2015,57(2):99−111.

[24] Volpe R, Nesnas I, Estlin T, Mutz D, Petras R, Das H. The CLARAty architecture for robotic autonomy. In: Proc. of the 2001 IEEE Aerospace Conf., 2001. 1–121.

[25] Kraetzschmar GK, Utz H, Sablatnog S, Enderle S, Palm G. Miro-Middleware for cooperative robotics. In: Proc. of the RoboCup. 2001. 411–416.

[26] Kramer J, Scheutz M. Development environments for autonomous mobile robots: A survey. Autonomous Robots, 2007,22(2): 101–132.

[27] Angerer A, Hoffmann A, Schierl A, Vistein M, Reif W. The Robotics API: An object-oriented framework for modeling industrial robotics applications. In: Proc. of the 2010 IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems. IEEE, 2010. 4036–4041.

[28] Wei C, Hindriks KV. An Agent-Based Cognitive Robot Architecture. Berlin, Heidelberg: Springer-Verlag, 2012. 54–71.

[29] Kouba A, Sriti MF, Bennaceur H, Ammar A, Javed Y, Alajlan M, Al-Elaiwi N, Tounsi M, Shakshuki E. COROS: A multi-agent software architecture for cooperative and autonomous service robots. In: Proc. of the Cooperative Robots and Sensor Networks. 2015. 3–30.

[30] Hsu HCH, Liu A. A flexible architecture for navigation control of a mobile robot. IEEE Trans. on Systems, Man, and Cybernetics —Part A: Systems and Humans, 2007,37(3):310–318.

[31] Kolp M, Giorgini P, Mylopoulos J. Multi-Agent architectures as organizational structures. Autonomous Agents and Multi-Agent Systems, 2006,13(1):3–25.

附中文参考文献:

[1] 马晓星,刘譞哲,谢冰,余萍,张天,卜磊,李宣东.软件开发方法发展回顾与展望.软件学报,2019,30(1):3–21. http://www.jos.org.cn/1000-9825/5650.htm

[2] 卜磊.形式化验证:从混成系统到 CPS.中国计算机学会通讯,2013,9(7):23–30.

[3] 毛新军.自主机器人软件技术.计算机学会通讯,2016,12(7):60–65.

毛新军(1970－),男,浙江江山人,博士,教授,博士生导师,CCF 杰出会员,主要研究领域为智能软件技术,群智软件工程,多智能体系统.

黄裕泓(1995－),男,硕士生,CCF 学生会员,主要研究领域为自主机器人软件,机器人控制.

杨硕(1992－),男,博士生,CCF 学生会员,主要研究领域为自主机器人软件,机器人任务规划,多智能体机器人软件架构.

王硕(1996－),男,硕士生,CCF 学生会员,主要研究领域为自主机器人软件,机器人任务规划,多智能体机器人软件架构.