软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software, [doi: 10.13328/j.cnki.jos.007131]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于 DH 标定的机器人正向运动学形式化验证

谢果君[1]，杨焕焕[2]，石正璞[1]，陈钢[1]

[1](南京航空航天大学 计算机科学与技术学院,江苏 南京 211106)

[2](国防科技大学 计算机学院,湖南 长沙 410073)

通讯作者: 陈钢, E-mail: gangchensh@nuaa.edu.cn

摘 要： DH 坐标系在机器人运动学分析中发挥着重要的作用。在基于 DH 坐标系构建的机器人控制系统中，机器人结构的复杂性使得构建安全的控制系统成为一个难题，仅仅依靠人工方法可能导致系统漏洞和安全风险，从而危及机器人的安全。形式化方法通过演绎推理与代码抽取实现了对软硬件系统的设计、开发及验证。基于此，本文设计了基于DH标定的机器人正向运动学的形式化验证框架。在Coq中构建了机器人运动理论的形式化证明，并验证控制算法的正确性以确保机器人的运动安全。首先，对 DH 坐标系进行形式化建模，构建相邻坐标系间转换矩阵的形式化定义，并验证了该转换矩阵与复合螺旋运动的等价性；其次，构建了机械臂正向运动学的形式化定义，并对机械臂运动的可分解性进行形式化验证；再次，本文对工业机器人中常见连杆结构及机器人进行形式化建模，并完成了正向运动学的形式化验证；最后，本文实现了 Coq 到 OCaml 的代码抽取，并对抽取的代码进行分析与验证。

关键词： 机器人运动学;形式化验证;DH 坐标系;代码自动生成

中图法分类号： TP311

中文引用格式:谢果君,杨焕焕,石正璞,陈钢.基于 DH 坐标系的机器人正向运动学形式化验证.软件学报. http://www.jos.org.cn/1000-9825/7131.htm

英文引用格式:Xie GJ,Yang HH, Shi ZP, Chen G. Formal verification of robot forward kinematics based on DH coordinate system. Ruan Jian Xue Bao/Journal of Software (in Chinese). http://www.jos.org.cn/1000-9825/7131.htm

Formal Verification of Robot Forward Kinematics Based on DH Coordinate system

XIE Guo-Jun[1], YANG Huan-Huan[2], SHI Zheng-Pu[1], CHEN Gang[1]

[1](College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China)

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

Abstract: The DH coordinate system is significantly important when analyzing robots' kinematics. In the robot control system built upon the DH coordinate system, the complexity of the robot structure poses challenges in developing a secure control system. Depending solely on manual methods could introduce system vulnerabilities and security hazards, thereby endangering the overall safety of the robot. The formal method becomes a promising direction to design, develop, and verify hardware and software systems through deductive reasoning and code extraction. Based on this, we design a formal verification framework for the forward kinematics of a robot based on the DH calibration, during which we rigorously prove the robot kinematics theory and verify the correctness of the control algorithm in Coq to ensure the safety of the robot's motion. To begin with, we formally model the DH coordinate system, define the transformation matrix between adjacent coordinate systems, and verify the equivalence of this transformation matrix with the composite helical motion. Then, we formally define the forward kinematics of the robotic arm and verify it's motion detachability. Subsequently, we formally model the common connecting rod structures and robots in industrial robots and verify their forward kinematics. Finally, we implement the code extraction from Coq to OCaml, analyze, and verify the extracted code.

Key words: Robot kinematics; Formal verification; DH coordinate system; Automatic code generation

近年来，机器人广泛应用于生产、医疗、交通、家庭等各个领域。机器人应用范围的不断扩大，使机器人运动安全性问题日益引人关注。对机器人进行控制时，不安全的行为不仅会导致意外事故发生，给人们带来生命财

收稿时间: 2023-09-10; 修改时间: 2023-10-30; 采用时间: 2023-12-13; jos 在线出版时间: 2024-01-05

产上的损失，而且会影响人们对机器人技术的信任。因此如何保障机器人运动的安全性成为近期研究的热点[1]。机器人控制系统与机器人运动安全息息相关，由于控制系统的设计与实现涉及到复杂的模型与算法，单纯采用人工设计容易出现漏洞从而引起安全隐患，这对机器人的安全构成了潜在的威胁。基于数学模型和理论的形式化技术为控制系统的建模和分析提供了新的手段[2]。形式化技术有助于识别设计中的漏洞和安全隐患，为机器人运动安全性问题提供有效的解决方案。采用形式化技术可以更精确地描述机器人的运动行为，并允许在设计阶段进行测试和验证[3]，进而有效降低了机器人运动相关的安全风险。

机器人运动学是几何学在控制理论中的应用，重点关注机器人运动状态的几何属性，包含位置和方向[4]，以实现对机器人的精确控制。本文中，机器人被描述为一个由多连杆和关节组成的刚体，机器人的每个关节都有自己的坐标系，称为刚体坐标系 $B(B_0\ B_1 \cdots B_n)$。机器人所处环境的坐标系被称为全局坐标系 G。坐标变换是机器人运动学的基础，在实现精确控制方面起着关键作用。机器人运动控制系统的形式化验证围绕坐标变换这一问题展开，旨在通过形式化验证机器人在各坐标系中的变换来提高机器人控制的安全性和稳定性。

基于数学建模和演绎推理的形式化技术为保障机器人控制算法设计的安全性提供了更加严格和可靠的方法[5]。尽管机器人运动学包含正向运动学和逆向运动学等多个方面，但本文重点对机器人的正向运动学进行形式化验证。本文形式化验证框架如图 1 所示。具体而言，在矩阵形式化的基础之上，本文对机器人的平动及旋转进行形式化定义，并对机器人运动的齐次矩阵进行形式化的推理与验证；之后，对 DH 坐标系进行形式化定义，并对连杆两端（相邻坐标系）转换与复合螺旋运动的等价性进行形式化证明；对特定类型的机器人($R \parallel R$ 机器人、$R \perp R \parallel R$ 机器人、空间机械手等)的结构进行形式化定义，并验证了对应的转换矩阵和正向运动学公式；最后对机器人控制系统中部分算法进行代码抽取。需要注意的是，Coq 代码抽取到 OCaml 的过程是递归进行的，这意味着在 Coq 中当一个目标函数被抽取为 OCaml 代码时，它所依赖的辅助函数也同时被抽取出来[6][7]。递归抽取的优势在于它能够保持代码的完整性和一致性。当目标函数需调用辅助函数时，抽取机制会自动识别并抽取这些辅助函数，从而消除手工处理依赖关系的需要。递归抽取功能提高了将 Coq 代码转换为 OCaml 代码的便利性和效率，同时确保所产生代码的准确性和可靠性。

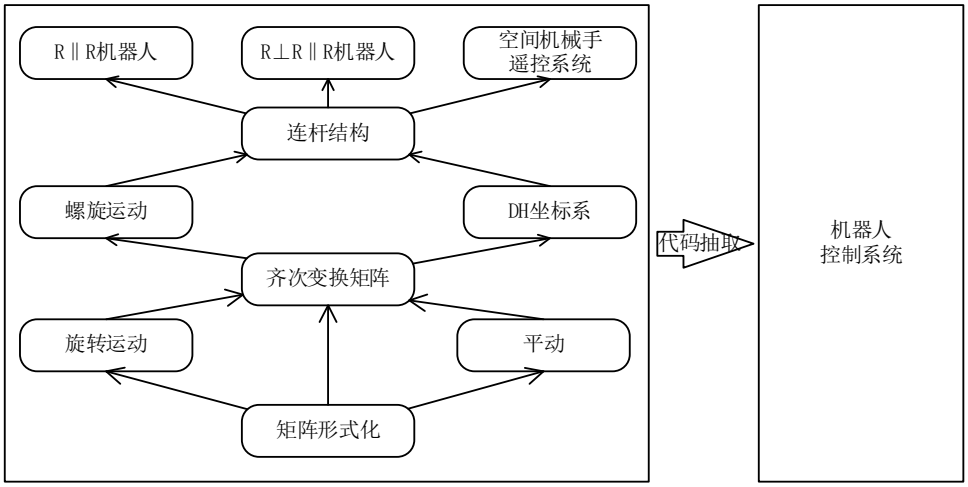

图 1　机器人正向运动学形式化框架图

本文的贡献可以概括为以下几点：首先，扩展了基于 *Record* 类型的形式化矩阵库，为机器人运动学的形式化验证奠定了数学基础。其次，实现了机器人运动学中齐次变换矩阵的形式化定义，并完成了基于 DH 标定的坐标系的形式化定义。再次，完成了基于 DH 坐标系的连杆结构的定义，并验证了连杆两端坐标系转换矩阵与复合

螺旋运动的等价性。最后，对常见结构机器人的坐标转化算法进行定义与验证，实现相关算法代码的抽取。

本文的组织结构如下：第 1 节介绍机器人形式化的相关方法和研究现状。第 2 节介绍本文所需的基础知识，包括矩阵形式化和机器人运动学。第 3 节介绍 DH 标定的形式化。第 4 节介绍机器人正向运动学的形式化定义，并完成相关理论的验证。第 5 节介绍代码的抽取，并对抽取后的 OCaml 代码进行理论分析。第 6 节总结全文。

1 相关工作

在机器人领域，形式化技术被广泛应用于人机协作、机器人系统开发与仿真、机器人基础理论验证、工业应用、自动驾驶等多个领域，下面进行详细的介绍。

在人机协作领域，Vicentini 等人[8]利用时序逻辑对任务执行方式进行描述，并采用形式化验证技术对状态空间进行自动探索与验证以便及早发现和修正危险情况，从而确保在人与机器人之间保持安全的共同工作空间。Isobe 等人[9]对协作运输机器人进行研究，使用模型检查工具 FDR 对有限状态机进行形式化建模与验证，从而证明了形式化方法在提高协作机器人可靠性的有效性。Lesting 等人[10]构建了一个基于形式建模、验证的框架，用于开发和分析在不确定的人类行为下，与人类高度互动的机器人应用，为人类的健康保护和任务完成提供了可靠的数学保证。Askarpour 等人[11]使用一个非确定性形式化模型来捕捉人类操作者的不确定行为，并分析协作机器人应用的安全性，使安全工程师能够在设计过程中考虑和减轻所有可能的错误行为，为未来工厂中人类和机器人在共同工作空间紧密合作的安全性提供了可靠的评估。

在机器人系统开发与仿真领域，Praveen 等人[12]针对分布式无人机在城市空域的运动问题，提供了一个在更高抽象层中集成设计和形式化验证的框架，加速从形式化验证系统过渡到仿真的开发周期。Martin 等人[13]使用不同的可执行规范语言对 Ros 中的 Navigation Stack 库进行定义，采用差分测试技术验证了规范与原始系统之间的等价性，并形式化验证了规划算法的若干性质。Dal 等人[14]通过自动生成机器人软件组件的形式化模型，实现了强大的监控和运行时验证功能，能够预防碰撞、触发紧急降落等，同时对可调度性、最坏遍历时间和互斥性等问题进行了形式化的验证。Bohrer 等人[15]提出了一种可重复使用的经过形式化验证安全网，为具有公差和加速度的 Dubins 型地面机器人的二维航点跟随提供端到端的安全性和有效性保证。Foughali 等人[16]提出一种搜索启发式算法，使用动态优先级调度器对时间自动机模型进行扩展，在同一模型检测框架下对扩展模型进行验证，实现对机器人在实时约束下行为的形式化验证，并实现了从机器人框架到 UPPAAL 的自动转换。Paul 等人[17]设计了基于形式化验证的运行时评估框架，用于航空航天系统的安全检查，支持对分布式和集中式航空应用的确定性和非确定性进行验证，包括可验证算法、软件、形式推理模型、形式化证明库和数据驱动的运行时验证方法。

在机器人基础理论验证领域，Xie 等人[18]使用 Coq 对机器人刚体运动中的坐标转换问题进行形式化验证，确保理论设计的正确性，并提供了一个通用的机器人运动学分析框架，有助于机器人控制系统的形式化验证。Lopez 等人[19]针对机器人系统中非结构化环境的异步交互问题，使用 Petri 网对任务级控制模型进行描述，使用 Petri 标记语言将任务描述语言（TDL, Task Description Language）翻译到 Petri 网，解决了不同 Petri 网工具之间的相互操作的难题。Sangnier 等人[20]研究了群体自组织机器人在离散空间中的验证问题，并表明在同步情况和特殊算法的异步情况下安全性问题可通过对 Presburger 算术公式的编码使用 SMT 求解器进行判定。Evangelidis 等人[21]对 Kalman 滤波器的效能进行严格的定量分析，建立了适用于线性离散时间随机系统的 Kalman 滤波器验证框架，评估和验证了在两个不同的概率运动模型和四个 Kalman 滤波器实现中的可扩展性和准确性。Abd 等人[22]提出了一种基于形式化建模和推理的可重构多智能体系统的方法，在该形式化方法中，智能体以不同的模式相互作用和通信，以完成共同的任务；智能体可以动态同步、交换数据、调整行为和重新配置其通信接口，扩展了线性时态逻辑（Linear Temporal Logic, LTL）验证智能体之间的交互意图和通信协议，并对这种扩展的可满足性和模型检测

复杂性进行了研究。Rashid 等人[23]提出了一种使用高阶逻辑定理证明对复杂物理系统进行 Laplace 和 Fourier 变换的方法，确保在信息物理融合系统（Cyber-physical Systems, CPS）的连续动态行为中的绝对准确性，从而实现对智能系统的形式化动态分析，为提高系统安全性提供了新的解决方案。Wang 等人[24]使用形式化工具开发了一种基于螺旋理论的方法，解决了传统 DH 参数方法在逆运动学问题建模和求解中的奇异性问题，保证了机器人系统的安全性。

在工业应用领域，Murray 等人[25]设计了一种新颖且通用的协同验证框架，通过硬件和软件模型之间的平台映射，实现了 ABB 工业喷漆机器人中的高压静电控制系统（High-Voltage electrostatic Control system, HVC）的形式化验证。Mkaouar 等人[26]将实时系统的形式化验证与架构分析和设计语言（Architecture Analysis and Design Language, AADL）模型开发流程相结合，通过使用 LNT 语言对实时任务模型进行形式化映射，并提供完整的自动化工具链，实现了对实时系统的事件驱动任务、异步通信和抢占式固定优先级调度的形式化验证。Sakata 等[27]提出一种工业控制系统（Industrial control systems, ICS）建模及其分析方法，利用 UPPAAL 和定时计算树逻辑（Timed Computation Tree Logic, TCTL）对控制系统进行形式化验证，确保在网络攻击下，监控器能够有效实现回退控制以保证系统的安全运行。Menghi 等人[28]解决了机器人任务的可靠性、性能、资源使用等关键属性的精确定义问题，并通过与概率性奖励计算树逻辑相结合的工具支持领域特定语言，实现了该问题形式化验证和机器人任务的自动规划。

在自动驾驶领域，Arcile 等人[29]提出了一种被称为 VERIFCAR 的形式化框架，用于验证自动驾驶车辆的决策策略，该方法通过使用时间自动机对通讯自组织车辆（Communicating Autonomous Vehicles, CAVs）进行形式化建模，实现对车辆行为的形式化分析与验证。Pek 等人[30]提出了一种形式化验证方案，用于确保自主车辆在任意城市交通场景中的安全性验证，并可在安全关键情况下提供备用方案，以确保自主车辆的安全性。Kabra 等人[31]形式化验证了包括轨道坡度和曲率、空气制动传播和戴维斯方程计算的阻力等在内的货运列车动力学模型的安全性，为改善铁路运营提供了新的控制方案。

2 预备知识

2.1 基于Coq的矩阵形式化

Coq、Isabelle 和 HOL Light 都是广泛应用于形式化方法和定理证明领域的工具，各自具有独特的特点。Coq 的底层逻辑基于依赖类型论（Dependent Type Theory），允许精确规范和验证软件及数学定理。它提供了交互式证明助手，支持自动化策略，拥有庞大的生态系统，适用于多个应用领域。然而，对于复杂证明，Coq 的证明脚本可能变得冗长和难以理解。Isabelle 提供了灵活的逻辑框架，便于定义新的逻辑和规则，特别适合形式化方法研究。它不仅用于定理证明，还可用于程序验证和形式化方法开发，支持多种逻辑系统，并提供了直观的 Isar 语言，使证明更清晰易懂。HOL Light 是一款轻量级工具，具有较快的证明速度。然而，其逻辑系统相对简单，可能无法表达某些复杂的数学属性与概念。

依赖类型论的主要特征是类型可以依赖于值，这允许更精确地表达程序行为和数学定理。在依赖类型论中，类型可以动态地依赖程序的输入和运行时信息，这与传统的静态类型不同。此外，依赖类型允许在类型中包含详细的规范信息，从而可以明确规定变量的取值范围、性质和约束条件，提供更严格的安全性和正确性保证。类型本身也是一等公民，可以像其他数据一样进行计算和构造。最重要的是，依赖类型论将编程和定理证明更紧密地集成在一起，使得程序和证明可以共享相同的表示方式，促进了形式化验证的实现。

矩阵形式化[32][33]在诸如机器人运动学的形式化工程数学中至关重要。Coq 社区中不同的组织和学者开发了多种矩阵形式化方案，每个方案都有其独特的特点和应用。Coquelicot[34]专注于实数分析和数值计算，使用迭代乘

积（Iterated Product）来表示向量和矩阵，包括矩阵群、矩阵环等，其中所有的“等于”关系都基于莱布尼茨等式。CoLoR[35]矩阵库是基于 Coq 标准库中向量库来开发的矩阵形式化库，具有良好的重写关系；该库在终止证书（Termination Certificates）的自动化验证中被广泛使用；该库实现了矩阵的各种运算和属性，如获取矩阵的元素、行或列、零矩阵、转置、加法和乘法；该矩阵库要求元素类型为半环（Semiring），这意味着它具有类似于环结构 $<R,+,\times>$；因此，N、Z、Q 和 R 等数据类型都可以作为元素类型来构成矩阵。MathComp[36]库使用了一种结合了归纳类型、乘积类型和有限集的复杂矩阵形式化，该库中的形式化矩阵理论极为丰富，几乎涵盖了前面讨论过的所有方案，并增加了块矩阵、行列式、邻接矩阵、逆矩阵和 LU 分解等矩阵运算。

然而，这些方案的形式化范围是有限的，由于底层矩阵模型的定义不同，这使得开发者难以在不同库之间切换。为了解决这些问题，Shi 等人[37]为矩阵理论提出了一套统一的分层接口，并遵照这些接口实现了一个名为 CoqMatrix 的开源多模型形式化矩阵库，这种实现有助于解耦底层库与上层应用之间的紧密联系。CoqMatrix 还构建了不同模型之间的双射转换函数，建立了不同模型之间的联系，为 Coq 中形式化矩阵理论的多模型和多应用适应性提供了一种新颖而有效的解决方案。

2.2 机器人运动学

机器人运动学是研究机器人在空间中运动的学科，主要关注机器人的位置和方向。机器人的运动状态通常通过关节角和末端执行器的位置和方向来描述。机器人的坐标系包括全局坐标系和局部（刚体）坐标系两种。在全局坐标系中，刚体的运动可分为旋转和平移两种方式。图 2 展示了刚体坐标系 B 在全局坐标系 G 中的状态，即经过旋转和平移后的状态。初始时刻，局部坐标系与全局坐标系重合，随后局部坐标系旋转至 B'，最后平移至 B。

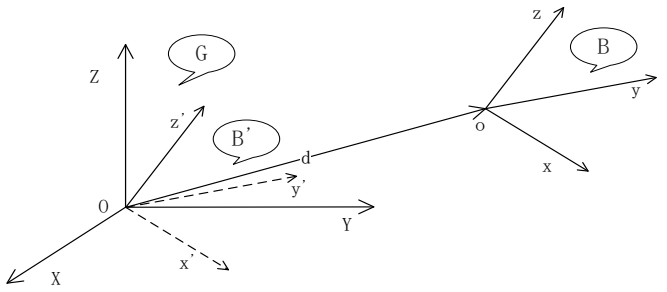

图 2　局部坐标系在全局坐标系中的运动

在刚体平动中，刚体内所有点的运动状态相同，因此可以用刚体坐标系中原点 o 的位置来描述刚体在全局坐标系 G 中的位移 d^G。刚体在全局坐标系中的运动如公式 (1) 所示，其中：r_P^G 表示刚体中任意一点 P 在全局坐标系 G 中的位置，r_P^B 表示点 P 在局部坐标系 B 中的位置，R_B^G 为局部坐标系到全局坐标系的旋转矩阵。

$$r_P^G = R_B^G * r_P^B + d^G \quad s.t. \begin{cases} r_P^G = \begin{bmatrix} X_P & Y_P & Z_P \end{bmatrix}^T \\ r_P^B = \begin{bmatrix} x_P & y_P & z_P \end{bmatrix}^T \\ d^G = \begin{bmatrix} x_0 & y_0 & z_0 \end{bmatrix}^T \end{cases} \tag{1}$$

在机器人运动学中，齐次变换技术被广泛用于描述刚体在全局坐标系中的位置与方向。齐次变换矩阵是 $4*4$ 的矩阵，矩阵的最后一行为 $[0,0,0,1]$。矩阵前 3 行被用来表示刚体的旋转和平移。齐次变换将多次的矩阵运算合并为一次运算，从而降低在求解运动学问题时的运算次数。机器人运动学中转换矩阵的齐次表示如公式 (2) 所示。

$$G_r = T_B^G * B_r \quad s.t. \begin{cases} T_B^G = \begin{pmatrix} R_B^G & d^G \\ 0 & 1 \end{pmatrix} \\ G_r = \begin{bmatrix} X_P & Y_P & Z_P & 1 \end{bmatrix}^T \\ B_r = \begin{bmatrix} x_P & y_P & z_P & 1 \end{bmatrix}^T \\ d^G = \begin{bmatrix} X_0 & Y_0 & Z_0 & 1 \end{bmatrix}^T \end{cases} \tag{2}$$

3 DH 标定

DH 标定是机器人运动学中用于描述机械臂运动常用的一种方法，DH 参数由机械臂各连杆在运动中的几何参数确立。DH 参数可以构建出机械臂的运动学模型，从而实现机械臂的精确运动与定位。机械臂之间通过关节串联，在本文中，平动关节符号记为 P，转动关节符号记为 R。

3.1 DH标定规则

在具有 n 个关节的串联机器人中存在 $n+1$ 个连杆（基体为编号为 0 的连杆），关节的编号从 1 开始。图 3 给出了具有关节 i、$i+1$ 的串联机器人结构图。对于关节 i，使用 DH 标定可以构建与之对应的局部坐标系 B_{i-1}，其中：关节 i 的最小取值为 1，B_0 为全局坐标系。在 DH 标定中坐标轴的选取规则如下：Z_{i-1} 轴与关节 i 的关节轴对齐，正方向可任意选取；X_{i-1} 轴沿 Z_{i-1} 轴与 Z_i 轴的公共法线，方向遵从右手螺旋定则从 Z_{i-1} 轴指向 Z_i 轴；Y_{i-1} 轴由 Z_{i-1} 轴与 X_{i-1} 轴的叉乘确定。在本文中，将通过 DH 标定规则确定的坐标系称为 DH 坐标系。

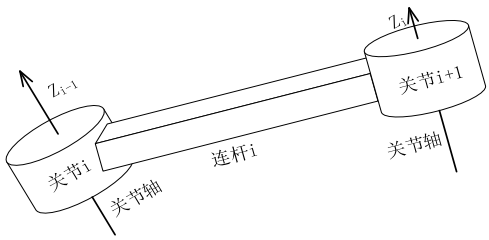

图 3 串联机器人结构图

坐标系 B_i 的 DH 参数包括连杆长度 (l_i)、连杆扭转角 (α_i)、关节距离 (d_i)、关节角 (θ_i) 等 4 个参数。其中：连杆长度 (l_i) 是 Z_{i-1} 轴与 Z_i 轴之间沿 X_i 轴平动的距离；连杆扭转角 (α_i) 是 Z_{i-1} 轴与 Z_i 轴之间绕 X_i 轴转动的夹角；关节距离 (d_i) 是 X_{i-1} 轴与 X_i 轴之间沿 Z_{i-1} 轴平动的距离；关节角 (θ_i) 是 X_{i-1} 轴与 X_i 轴之间绕 Z_{i-1} 轴转动的夹角。

图 4 显示了 2R 平面机器人的 DH 坐标系及参数。该模型共有 3 个坐标系 $(B_0 \quad B_1 \quad B_2)$。$B_0$ 被称为全局坐标系，它的选取是任意的（为方便描述，本文将 B_0 的原点固定在编号为 1 的关节中）。局部坐标系 B_1 和 B_2 的 DH 参数为：连杆长度：l_1 和 l_2、连杆扭转角：0 和 0、关节距离：0 和 0、关节角：θ_1 和 θ_2。

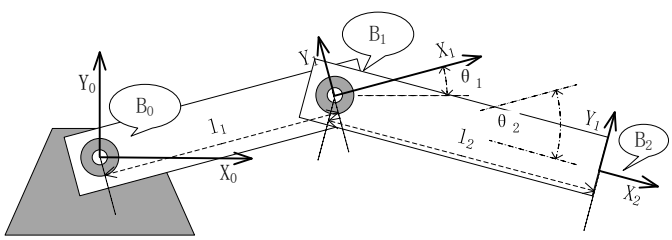

图 4 2R 平面机器人 DH 坐标系图

3.2 DH坐标系的形式化定义

在 DH 坐标系中，坐标系 B_i 固定在关节 $i+1$ 上，坐标系 B_{i-1} 固定在关节 i 上，坐标系 B_i 至坐标系 B_{i-1} 的变换如公式 (3) 所示，其中：α_i 为连杆扭转角、l_i 为连杆长度、θ_i 为关节角、d_i 为关节距离。

$$T_i^{i-1} = D_{(Z_{i-1},d_i)} * R_{(Z_{i-1},\theta_i)} * D_{(X_i,l_i)} * R_{(X_i,\alpha_i)} \tag{3}$$

在 Coq 中，Class 机制定义类型类，用于描述一组相关的类型和操作的抽象接口。引入 Class 机制，DH 参数的定义如 *DH_PRM* 所示，其中：l 对应连杆长度，α 对应连杆的扭转角，d 对应关节距离，θ 对应关节角。

$$Class\ DH\_PRM : Type := \{ l : R;\quad \alpha : R;\quad d : R;\quad \theta : R \}.$$

根据公式(3)，坐标系 B_i 至坐标系 B_{i-1} 的形式化定义如定义 1 中的 *Tnear* 所示。该函数接受一个 *DH_PRM* 类型的参数，返回对应的齐次转换矩阵。因此 *Tnear* 的类型为：$DH\_PRM \to mat\ 4\ 4$ 。

定义 1：在 DH 坐标系中，设 *prm* 为坐标系 B_i 对应的 DH 参数，则将坐标系 B_i 转换至坐标系 B_{i-1} 的变换矩阵形式化定义如下：

$$Definition\quad Tnear\ (prm : DH\_PRM) := (DZ\ prm.(d)) * (RZ\ prm.(\theta)) * (DX\ prm.(l)) * (RX\ prm.(\alpha))$$

对于同一个 DH 坐标系的变换矩阵，该运动可等价描述为：首先沿着 x_i 进行平动与转动，然后沿着 z_{i-1} 进行平动与转动。对于任意一次运动其平动与旋转沿着相同坐标轴，因此变换矩阵与运动次序无关。该性质的形式化描述与证明如下：

引理 1：在 DH 坐标系中，假设 *prm* 为坐标系 B_i 的任意 DH 参数，则坐标系 B_i 转换至坐标系 B_{i-1} 的变换矩阵与 $D_{(z_{i-1},d_1)} * R_{(z_{i-1},\theta_1)} * R_{(x_i,\alpha_i)} * D_{(x_i,l_i)}$ 等价。

$$Lemma\ Tnear\_eq1 : forall\ (prm : DH\_PRM),$$
$$Tnear\ prm = (DZ\ prm.(d)) * (RZ\ prm.(\theta)) * (RX\ prm.(\alpha)) * (DX\ prm.(l)).$$

引理 2：在 DH 坐标系中，假设 *prm* 为坐标系 B_i 的任意 DH 参数，则坐标系 B_i 转换至坐标系 B_{i-1} 的变换矩阵与 $R_{(z_{i-1},\theta_1)} * D_{(z_{i-1},d_1)} * D_{(x_i,l_i)} * R_{(x_i,\alpha_i)}$ 等价。

$$Lemma\ Tnear\_eq2 : forall\ (prm : DH\_PRM),$$
$$Tnear\ prm = (RZ\ prm.(\theta)) * (DZ\ prm.(d)) * (DX\ prm.(l)) * (RX\ prm.(\alpha)).$$

引理 3：在 DH 坐标系中，假设 *prm* 为坐标系 B_i 的任意 DH 参数，则坐标系 B_i 转换至坐标系 B_{i-1} 的变换矩阵与 $R_{(z_{i-1},\theta_1)} * D_{(z_{i-1},d_1)} * R_{(x_i,\alpha_i)} * D_{(x_i,l_i)}$ 等价。

$$Lemma\ Tnear\_eq3 : forall\ (prm : DH\_PRM),$$
$$Tnear\ prm = (RZ\ prm.(\theta)) * (DZ\ prm.(d)) * (RX\ prm.(\alpha)) * (DX\ prm.(l)).$$

3.3 相邻坐标系的螺旋表示

螺旋运动是同时具有平动和旋转的运动形式。在螺旋运动中，物体绕着一个中心轴旋转，并沿着该中心轴平移。中心螺旋运动是螺旋运动的一种特殊情况，即中心轴经过坐标系原点。中心螺旋运动如公式(4)所示，其中 h 表示沿中心轴的平动距离，φ 表示绕中心轴的旋转弧度，$\vec{u}$ 表示中心轴的方向向量。

$$S_{(h,\varphi,\vec{u})} = D_{(\vec{u},h)} * R_{(\vec{u},\varphi)} \tag{4}$$

由公式 (3) 与公式 (4)，可推出公式 (5) 所示的 DH 坐标系中坐标系 B_i 到坐标系 B_{i-1} 的变换矩阵，其中：$S_{(l_i,\alpha_i,X_i)}$ 表示平动距离为 l_i，旋转弧度为 α_i，中心轴为 X_i 轴的中心螺旋运动；$S_{(d_i,\theta_i,Z_{i-1})}$ 表示平动距离为 d_i，旋转

弧度为 θ_i，中心轴为 Z_{i-1} 轴的中心螺旋运动。

$$T_i^{i-1} = S_{(d_i,\theta_i,Z_{i-1})} * S_{(l_i,\alpha_i,X_i)} \tag{5}$$

在中心螺旋运动中，坐标转换矩阵形式化定义由如下的 *fun_GTB_center* 给出。该函数接收 2 个实数类型和 1 个 3∗1 矩阵类型的变量，返回该运动对应的齐次转换矩阵。因此，该函数类型为 $R \to R \to mat\ 3\ 1 \to mat\ 4\ 4$。

```
Definition  fun_GTB_center (h φ : R)(u : mat 3 1) :=
  let u' := (1/| u |) c* u in
  (D4 (h c* u')) * (R4 φ u')
```

Coq 中的 *Notation* 策略用于自定义符号和语法，以增强 Coq 的可读性和易用性。它允许用户定义自己的运算符、记法和语法规则，进而使代码更接近于人类的数学规约。基于 *Notation* 策略，给出 *fun_GTB_center* 的中缀符号定义如下：

```
Reserved Notation " S( h, φ, u )" (at level 34).
Notation " S( h, φ, u )" := (fun_GTB_center h φ u) : mat_scope.
```

在上述中缀符号 $S(h,\varphi,u)$ 的基础上，公式 (5) 的形式化证明如引理 4 所示。

引理 4：在 DH 坐标系中，假设 *prm* 为坐标系 B_i 的任意 DH 参数，则坐标系 B_i 到坐标系 B_{i-1} 的变换矩阵与 $S_{(d_i,\theta_i,Z_{i-1})} * S_{(l_i,\alpha_i,X_i)}$ 等价。

```
Lemma Tnear_S_eq : forall prm : DH_PRM, Tnear prm = S(d,θ,Z) * S(l,α,X).
```

4 正向运动学的形式化定义

4.1 正向运动学基础理论的形式化定义

正向运动学（Forward Kinematics，FK）是指通过关节变量计算机器人末端执行器在全局坐标系中的位置和姿态等。在机器人控制系统中，正向运动学是机器人控制中最基本的问题之一。对于任意结构的机器人，其 DH 标定的正向运动学方程如公式 (6) 所示，其中：r_P^n 表示末端执行器在局部坐标系 B_n 中的位置，r_P^0 表示末端执行器在全局坐标系中的位置，T_n^0 为综合变换矩阵。

$$\begin{aligned} r_P^0 &= T_n^0 * r_P^n \\ T_n^0 &= T_1^0 * T_2^1 * \cdots * T_n^{n-1} \end{aligned} \tag{6}$$

对于任意的 B_j 到 B_i 的综合变换矩阵，其形式化定义如定义 2 中的 T 所示。在定义 2 中，使用 *Fixpoint* 策略定义了一个递归函数，该函数接受 3 个参数，其中：i 表示目的坐标系编号，j 表示源坐标系编号，*prms* 表示该结构中所有 DH 参数的列表，默认参数 *Default_prm* 为所有变量都为 0 的 DH 参数。

定义 2：在 DH 坐标系中对于任意结构的机器人，设 *prms* 为该机器人的 DH 参数列表，则坐标系 B_j 到坐标系 B_i 的综合变换矩阵形式化定义如下：

```
Fixpoint  T (i j : nat)(prms : list DH_PRM) { struct prms } :=
  match prms with
    | [] ⇒ Tnear Default _ prm
    | h :: t ⇒ match  i, j with
                 | 0,0 ⇒ Tnear Default_prm        | 0,(S j') ⇒ (Tnear h) * (T 0 j' t)
                 | (S i'),0 ⇒ Tnear Default_prm   | (S i'),(S j') ⇒ T i' j' t
               end
  end.
```

在定义 2 的基础上，基于 DH 标定的正向运动学方程的形式化定义如定义 3 中的 *fun_FK_DH* 所示。在该定义中，使用 *let* 策略定义临时变量，*length* 函数获取一个列表长度；因此，该函数的类型为 $list\ DH\_PRM \to mat\ 3\ 1 \to mat\ 4\ 1$。

定义 3：给定 DH 坐标系中任意结构的机器人，若 *prms* 为该机器人对应的 DH 参数列表，*r* 为末端执行器 *P* 在局部坐标 B_n 中的坐标，则 *P* 在全局坐标系 B_0 中坐标的形式化定义如下：

```
Definition fun_FK_DH (prms : list DH_PRM) (r : mat 3 1) :=
  let r0 := mnth r 0 0 in
  let r1 := mnth r 1 0 in
  let r2 := mnth r 2 0 in
  let r' := {{r0}, {r1}, {r2}, {1}} in
  let len := length prms in
  T 0 len prms * r'.
```

对于任意的综合变换矩阵 T_j^i，综合变换矩阵的可分解性可以被形式化验证。

引理 5：在 DH 标定的机器人结构中，对于任意的综合变换矩阵 T_j^i，可以分解为 $T_k^i * T_j^k$，其中 $i \le k \le j$。

```
Lemma T_split : forall (i j k : nat) (p : list DH_PRM), i <= k <= j → (T i j p) = (T i k p) * (T k j p).
```

4.2 连杆的形式化定义

在工业机器人中根据连杆两端关节类型及 Z_i 轴与 Z_{i-1} 轴的夹角关系，本文将任意连杆 *i* 分为如表 1 所示的 8 种类型，其中：$\|$ 表示 Z_i 轴与 Z_{i-1} 轴平行，$\perp$ 表示 Z_i 轴与 Z_{i-1} 轴垂直，~代表任意类型关节。

表 1：连杆类型

连接符号	类型说明	连接符号	类型说明
$R \|\|\sim (0)$	关节 *i* 为旋转关节，Z_i 轴与 Z_{i-1} 轴同向	$P \|\|\sim (0)$	关节 *i* 为平动关节，Z_i 轴与 Z_{i-1} 轴同向
$R \|\|\sim (\pi)$	关节 *i* 为旋转关节，Z_i 轴与 Z_{i-1} 轴反向	$P \|\|\sim (\pi)$	关节 *i* 为平动关节，Z_i 轴与 Z_{i-1} 轴反向
$R \perp\sim (\pi/2)$	关节 *i* 为旋转关节，Z_i 轴与 Z_{i-1} 轴夹角为 $\pi/2$	$P \perp\sim (\pi/2)$	关节 *i* 为平动关节，Z_i 轴与 Z_{i-1} 轴夹角为 $\pi/2$
$R \perp\sim (-\pi/2)$	关节 *i* 为旋转关节，Z_i 轴与 Z_{i-1} 轴夹角为 $-\pi/2$	$P \perp\sim (-\pi/2)$	关节 *i* 为平动关节，Z_i 轴与 Z_{i-1} 轴夹角为 $-\pi/2$

因此，在任意基于 DH 标定的机器人结构的形式化定义中，本文将连杆结构归纳定义为 8 种不同的类型，如定义 4 中的 *Pole* 所示。其中：构造子 *R_0*、*R_pi*、*R_pi2*、*R_pi2'* 的类型为 $R \to R \to R \to Pole$ 类型，代表连杆近端为螺旋关节；*P_0*、*P_pi*、*P_pi2*、*P_pi2'* 的类型为 $R \to R \to Pole$ 类型，代表连杆近端为平动关节。

定义 4：在 DH 标定的机器人结构中，机器人连杆类型被归纳定义为 8 种，其中：*R_0* 表示近端为旋转关节且远端与近端关节轴平行且同向；*R_pi* 表示近端为旋转关节且远端与近端关节轴平行且反向；*R_pi2* 表示近端为旋转关节且远端与近端关节轴夹角为 $\pi/2$；*R_pi2'* 表示近端为旋转关节且远端与近端关节轴夹角为 $-\pi/2$；*P_0* 表示近端为平动关节且远端与近端关节轴平行且同向；*P_pi* 表示近端为平动关节且远端与近端关节轴平行且反向；*P_pi2* 表示近端为平动关节且远端与近端关节轴夹角为 $\pi/2$；*P_pi2'* 表示近端为平动关节且远端与近端关节轴夹角为 $-\pi/2$。

$$
\begin{aligned}
&Inductive\quad Pole:Type:=\\
&\quad |R\_0:R\to R\to R\to Pole \quad |R\_pi:R\to R\to R\to Pole\\
&\quad |R\_pi2:R\to R\to R\to Pole \quad |R\_pi2':R\to R\to R\to Pole\\
&\quad |P\_0:R\to R\to Pole \quad |P\_pi:R\to R\to Pole\\
&\quad |P\_pi2:R\to R\to Pole \quad |P\_pi2':R\to R\to Pole.
\end{aligned}
$$

对于不同连杆类型，其转换矩阵(从远端到近端)的形式化定义如定义 5 中的 *fun_poleT* 所示。

定义 5：在 DH 标定的机器人结构中，若 *pole* 为连杆类型，且该连杆的连杆长度为 l，关节距离为 d，关节角为 θ，则该连杆转换矩阵的形式化定义如下：

$$
\begin{aligned}
&Definition\ fun\_poleT\ (pole:Pole)(l\ d\ \theta:R):=\\
&\quad let\ prm:=match\ pole\ with\\
&\quad |R\_0\ l\ d\ \theta\Rightarrow Build\_DH\_PRM\ l\ 0\ d\ \theta \quad |R\_pi\ l\ d\ \theta\Rightarrow Build\_DH\_PRM\ l\ PI\ d\ \theta\\
&\quad |R\_pi2\ l\ d\ \theta\Rightarrow Build\_DH\_PRM\ l\ (PI/2)\ d\ \theta \quad |R\_pi2'\ l\ d\ \theta\Rightarrow Build\_DH\_PRM\ l\ (-PI/2)\ d\ \theta\\
&\quad |P\_0\ l\ d\Rightarrow Build\_DH\_PRM\ l\ 0\ d\ 0 \quad |P\_pi\ l\ d\Rightarrow Build\_DH\_PRM\ l\ PI\ d\ 0\\
&\quad |P\_pi2\ l\ d\Rightarrow Build\_DH\_PRM\ l\ (PI/2)\ d\ 0 \quad |P\_po2'\ l\ d\Rightarrow Build\_DH\_PRM\ l\ (-PI/2)\ d\ 0\\
&\quad end\ in\\
&\quad Tnear\ prm
\end{aligned}
$$

对于近端为R类型关节，图 5(a)与图 5(b)分别显示了 $R\|R$ 和 $R\|P$ 类型的连杆结构图。在该结构中，连杆长度 (l_i)、关节距离 (d_i)为常量，关节角 (θ_i) 为唯一可变参数。因此 $R\|\sim$ 类型连杆转换矩阵的形式化验证如下。

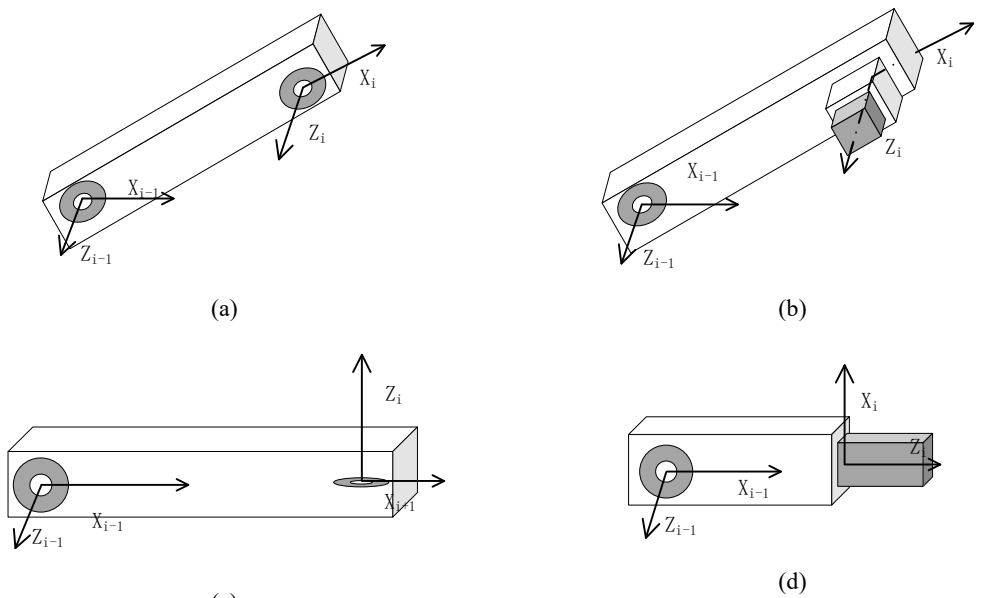

图 5 $R\sim$连杆结构图

引理 6：在 DH 坐标系中，设连杆 i 的类型为 $R\|\sim(0)$，则 B_i 到 B_{i-1} 的变换矩阵为：$[[cos\ \theta, -sin\ \theta, 0, l*cos\ \theta];[sin\ \theta, cos\ \theta, 0, l*sin\ \theta];[0, 0, 1, d];[0, 0, 0, 1]]$。

$$
\begin{aligned}
&Lemma\ R0\_check:\ forall\ \{po:Pole\}(l\ d\ \theta:R)\to po=R\_0\ l\ d\ \theta\to fun\_poleT\ po\ l\ d\ \theta=\\
&\quad \{\{cos\ \theta, -sin\ \theta, 0, l*cos\ \theta\},\ \{sin\ \theta, cos\ \theta, 0, l*sin\ \theta\},\ \{0, 0, 1, d\},\ \{0, 0, 0, 1\}\}.
\end{aligned}
$$

引理 7：在 DH 坐标系中，设连杆 i 的类型为 $R\|\sim(\pi)$，则 B_i 到 B_{i-1} 的变换矩阵为：$[[cos\ \theta, sin\ \theta, 0, l*cos\ \theta];[sin\ \theta, -cos\ \theta, 0, l*sin\ \theta];[0, 0, -1, d];[0, 0, 0, 1]]$。

$$Lemma\ Rpi\_check:\ forall\ \{po:Pole\}(l\ d\ \theta:R) \to po = R\_pi\ l\ d\ \theta \to fun\_poleT\ po\ l\ d\ \theta = \{\{cos\ \theta, sin\ \theta, 0, l*cos\ \theta\},\ \{sin\ \theta, -cos\ \theta, 0, l*sin\ \theta\},\ \{0, 0, -1, d\},\ \{0, 0, 0, 1\}\}.$$

图 5(c)、图 5(d)分别显示了 $R\perp R$ 类型、$R\perp P$ 类型的连杆结构图。根据 Z_{i-1} 与 Z_i 的方向，连杆扭转角可以划分为 $\alpha_i=\pi/2$ 和 $\alpha_i=-\pi/2$ 两种情况。引理 8 与引理 9 对这两种情况下 B_i 到 B_{i-1} 的变换矩阵进行形式化验证。

引理 8：在 DH 坐标系中，若连杆 i 的类型为 $R\perp\sim(\pi/2)$ 类型，则 B_i 到 B_{i-1} 的变换矩阵为：$[[cos\ \theta, 0, sin\ \theta, l*cos\ \theta];[sin\ \theta, 0, -cos\ \theta, l*sin\ \theta];[0, 1, 0, d];[0, 0, 0, 1]]$。

$$Lemma\ Rpi2\_check:\ forall\ \{po:Pole\}(l\ d\ \theta:R) \to po = R\_pi2\ l\ d\ \theta \to fun\_poleT\ po\ l\ d\ \theta = \{\{cos\ \theta, 0, sin\ \theta, l*cos\ \theta\},\ \{sin\ \theta, 0, -cos\ \theta, l*sin\ \theta\},\ \{0, 1, 0, d\},\ \{0, 0, 0, 1\}\}.$$

引理 9：在 DH 坐标系中，若连杆 i 的类型为 $R\perp\sim(-\pi/2)$ 类型，则 B_i 到 B_{i-1} 的变换矩阵为：$[[cos\ \theta, 0, -sin\ \theta, l*cos\ \theta];[sin\ \theta, 0, cos\ \theta, l*sin\ \theta];[0, -1, 0, d];[0, 0, 0, 1]]$。

$$Lemma\ Ropppi\_check:\ forall\ \{po:Pole\}(l\ d\ \theta:R) \to po = R\_pi2'\ l\ d\ \theta \to fun\_poleT\ po\ l\ d\ \theta = \{\{cos\ \theta, 0, -sin\ \theta, l*cos\ \theta\},\ \{sin\ \theta, 0, cos\ \theta, l*sin\ \theta\},\ \{0, -1, 0, d\},\ \{0, 0, 0, 1\}\}.$$

同样地，对于近端为 P 类型关节，图 6(a)与图 6(b)分别显示了 $P\|R$ 和 $P\|P$ 类型的连杆结构图。在该结构中，连杆长度(l_i)、连杆扭转角(α_i)及关节角(θ_i)为常量(关节角为 0)，关节距离(d_i)为唯一可变参数。$P\|\sim$ 类型连杆转换矩阵的形式化验证如下。

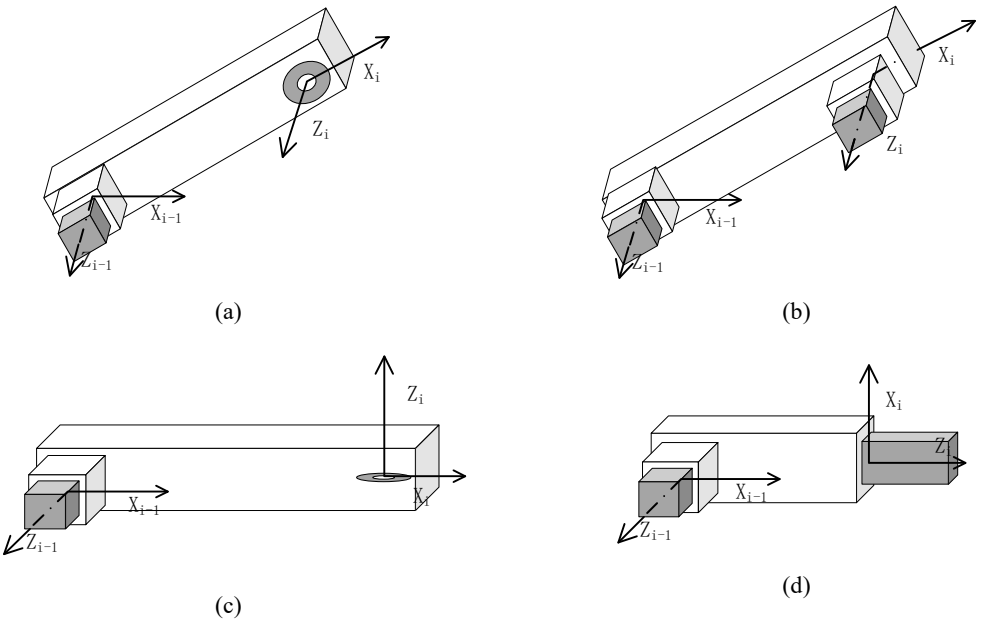

图 6 $P\sim$连杆结构图

引理 10：在 DH 坐标系中，设连杆 i 的类型为 $P\|\sim(0)$，则 B_i 到 B_{i-1} 的变换矩阵为：$[[1, 0, 0, l];[0, 1, 0, 0];[0, 0, 1, d];[0, 0, 0, 1]]$。

$$Lemma\ P0\_check:\ forall\ \{po:Pole\}(l\ d:R) \to po = P\_0\ l\ d \to fun\_poleT\ po\ l\ d\ 0 = \{\{1, 0, 0, l\},\ \{0, 1, 0, 0\},\ \{0, 0, 1, d\},\ \{0, 0, 0, 1\}\}.$$

引理 11：在 DH 坐标系中，设连杆 i 的类型为 $P\|\sim(\pi)$，则 B_i 到 B_{i-1} 的变换矩阵为：

$[[1,0,0,l];[0,-1,0,0];[0,0,-1,d];[0,0,0,1]]$。

$$Lemma\ Ppi\_check:\ forall\ \{po:Pole\}(l\ d:R)\rightarrow po = P\_pi\ l\ d \rightarrow fun\_poleT\ po\ l\ d\ 0 = \{\{1,0,0,l\},\ \{0,-1,0,0\},\ \{0,0,-1,d\},\ \{0,0,0,1\}\}.$$

图 6(c)、图 6(d)分别显示了 $P \perp R$ 和 $P \perp P$ 类型的连杆结构图。根据 Z_{i-1} 与 Z_i 的方向，连杆扭转角可以划分为 $\alpha_i = \pi/2$ 和 $\alpha_i = -\pi/2$ 两种情况。引理 12 与引理 13 对这两种情况下 B_i 到 B_{i-1} 的变换矩阵进行形式化验证。

引理 12：在 DH 坐标系中，若连杆 i 的类型为 $P \perp\sim (\pi/2)$ 类型，则 B_i 到 B_{i-1} 的变换矩阵为：$[[1,0,0,l];[0,0,-1,0];[0,1,0,d];[0,0,0,1]]$。

$$Lemma\ Ppi2\_check:\ forall\ \{po:Pole\}(l\ d:R)\rightarrow po = P\_pi2\ l\ d \rightarrow fun\_poleT\ po\ l\ d\ 0 = \{\{1,0,0,l\},\ \{0,0,-1,0\},\ \{0,1,0,d\},\ \{0,0,0,1\}\}$$

引理 13：在 DH 坐标系中，若连杆 i 的类型为 $P \perp\sim (-\pi/2)$，则 B_i 到 B_{i-1} 的变换矩阵为：$[[1,0,0,l];[0,0,1,0];[0,-1,0,d];[0,0,0,1]]$。

$$Lemma\ Popppi\_check:\ forall\ \{po:Pole\}(l\ d:R)\rightarrow po = P\_pi2'\ l\ d \rightarrow fun\_poleT\ po\ l\ d\ 0 = \{\{1,0,0,l\},\ \{0,0,1,0\},\ \{0,-1,0,d\},\ \{0,0,0,1\}\}.$$

4.3 机器人实例的形式化验证

作为 Coq 中用于定义多态的一种机制，类型类允许在不同类型上定义相同名字的操作，使得这些操作可以在不同类型的对象上具有类似的行为。本文关注机器人正向运动学的形式化定义与验证，针对不同类型的机器人（本文选取了 $R \| R$ 平面机器人、$R \perp R \| R$ 机器人、空间站机械手），其转换矩阵求解函数、坐标转换函数被封装在 *fun_Get* 中，由 *fun_Get* 对外提供统一的接口。其中：*getT* 表示转换矩阵求解函数，其类型为 $A \rightarrow mat\ 4\ 4$；*getGr* 为坐标转换函数，其类型为 $A \rightarrow mat\ 3\ 1 \rightarrow mat\ 4\ 1$。

$$Class\quad fun\_Get\ A:Type := \{$$
$$getT : A \rightarrow mat\ 4\ 4;$$
$$getGr : A \rightarrow mat\ 3\ 1 \rightarrow mat\ 4\ 1;\}$$

4.3.1 $R \| R$ 平面机器人

$R \| R$ 平面机器人是一种用于简单装配和操作的机器人。它由两个旋转关节连接，在二维平面内移动和执行任务。图 4 显示了 $R \| R$ 平面机器人的 DH 坐标系模型，其中 Z_0、Z_1、Z_2 的方向垂直于平面向上。因此，$R \| R$ 平面机器人的 DH 参数如表 2 所示。

表 2：$R \| R$ 平面机器人 DH 参数表

坐标系编号	连杆长度	连杆扭转	关节距离	关节角
1	l_1	0	0	θ_1
2	l_2	0	0	θ_2

$R \| R$ 结构的形式化定义如下述 *RR_Type* 所示。

$$Class\ RR\_type : Type := \{l1:R; l2:R; \theta1:R; \theta2:R\}.$$

定义 6 中的 *fun_RR_T* 与 *fun_RR* 分别为 $R \| R$ 平面机器人的转换矩阵与坐标转换函数的形式化定义。它们被定义在一个 *Section* 结构中。

定义 6：在 DH 坐标系中，若机器人 *rob* 的类型为 *RR_type*，则该机器人的转换矩阵及坐标转换算法的形式化定义如下：

```
Section RR.
  Variable  rob : RR_type.
  Let  prm := [(Build_DH_PRM l1 0 0 θ1); (Build_DH_PRM l2 0 0 θ2)].
  Definition  fun_RR_T := T 0 (length prm) prm.
  Definition  fun_RR (Pb : mat 3 1) :=
    let b0 := mnth Pb 0 0 in  let b1 := mnth Pb 1 0 in
    let b2 := mnth Pb 2 0 in  let b := {{b0}, {b1}, {b2}, {l}} in
    fun_RR_T * b.
End RR.
```

在 Coq 中，*Instance* 策略用于定义类型类的实例，$R \| R$ 机器人在类型类 *fun_Get* 的实例化如 *get_RR* 所示。

```
#[export] Instance  get_RR : fun_Get RR_type := {
  getT := fun_RR_T;
  getGr := fun_RR;}
```

在 $R \| R$ 结构的机器人中，以下引理可以被形式化证明。

引理 14：在 DH 坐标系中，若机器人 *rob* 的类型为 *RR_type*，则该机器人转换矩阵具有如下等价关系。

```
Lemma  RR_T_check : forall (rob : RR_type), getT rob =
  {{cos (θ1 + θ2), - sin (θ1 + θ2),  0,  l1 * cos θ1 + l2 * cos (θ1 + θ2)},
   {sin (θ1 + θ2),  cos (θ1 + θ2),  0,  l1 * sin θ1 + l2 * sin (θ1 + θ2)},
   {  0,            0,              1,  0                                 },
   {  0,            0,              0,  1                                 }}.
```

引理 15：在 DH 坐标系中，若机器人 *rob* 的类型为 *RR_type*，则该机器人的末端执行器在全局坐标系中的位置为 $[[l1 * cos\ \theta1 + l2 * cos\ (\theta1 + \theta2)]; [l1 * sin\ \theta1 + l2 * sin\ (\theta1 + \theta2)]; [0]; [1]]$。

```
Let  Br := {{0}, {0}, {0}}.
Lemma  RR_Gr_check: forall (rob:RR_type), getGr rob Br =
  {{l1 * cos θ1 + l2 * cos (θ1 + θ2)}, {l1 * sin θ1 + l2 * sin (θ1 + θ2)}, {0}, {1}}.
```

4.3.2 $R \perp R \| R$ 机器人

$R \perp R \| R$ 机器人由 3 个旋转关节连接，可以在 3 维空间内移动和执行任务。图 7 给出了 $R \perp R \| R$ 机器人的 DH 坐标系模型，其中：Z_0 与 Z_1 垂直相交，Z_1 与 Z_2 同向平行。因此，$R \perp R \| R$ 机器人的 DH 参数如表 3 所示。

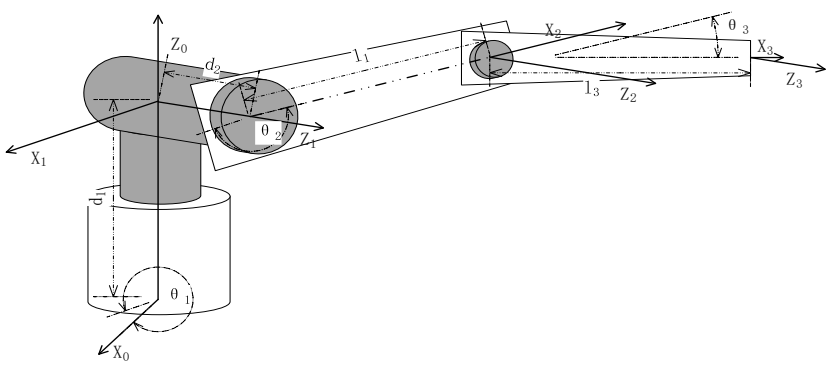

图 7　$R \perp R \| R$ 机器人 DH 坐标系图

表 3： $R\perp R\|R$ 机器人 DH 参数表

坐标系编号	连杆长度	连杆扭转	关节距离	关节角
1	0	$-\pi/2$	d_1	θ_1
2	l_2	0	d_2	θ_2
3	l_3	0	0	θ_3

$R\perp R\|R$ 结构的形式化定义如下述 *RRR_Type* 所示。。

$$Class\ RRR\_type:Type:=\{l2:R;l3:R;d1:R;d2:R;\theta1:R;\theta2:R;\theta3:R\}.$$

定义 7 中的 *fun_RRR_T* 与 *fun_RRR* 分别为 $R\perp R\|R$ 器人的转换矩阵与坐标转换求解函数的形式化定义。

定义 7：在 DH 坐标系中，若机器人 *rob* 的类型为 *RRR_type*，则该机器人的转换矩阵及坐标转换算法的形式化定义如下：

Section RRR.

Variable rob : RRR_type.

Let prms: = [Build_DH_PRM 0 (– PI/2) d1 θ1;Build_DH_PRM l2 0 d2 θ2;Build_DH_PRM l3 0 0 θ3].

Definition fun_RRR_T: = T 0 (length prms) prms.

Definition fun_RRR (Pb:mat31) : =

let b0 : = mnth Pb 0 0 in let b1 : = mnth Pb 1 0 in let b2: = mnth Pb 2 0 in let b: = {{b0}, {b1}, {b2}, {1}} in

*fun_RRR_T * b.*

End RRR.

因此， $R\perp R\|R$ 机器人在类型类 *fun_Get* 的实例化如 *get_RRR* 所示。

$$\begin{aligned}&\#[export]\ Instance\quad get\_RRR:fun\_Get\quad RRR\_type:=\{\\&\quad getT:=fun\_RRR\_T;\\&\quad getGr:=fun\_RRR;\}\end{aligned}$$

在 $R\perp R\|R$ 结构的机器人中，以下引理可以被形式化证明：

引理 16：在 DH 坐标系中，若机器人 *rob* 的类型为 *RRR_type*，则该类型机器人转换矩阵具有如下等价关系。

$$\begin{aligned}&Lemma\ RRR\_T\_check:\ forall\ (rob:\ RRR\_type),\ getT\ rob =\\&\{\{\ cos\,\theta1*cos\,(\theta2+\theta3),\ -cos\,\theta1*sin\,(\theta2+\theta3),\ -sin\,\theta1,\ l3*cos\,\theta1*cos\,(\theta2+\theta3)+l2*cos\,\theta1*cos\,\theta2-d2*sin\,\theta1\ \},\\&\ \{\ sin\,\theta1*cos\,(\theta2+\theta3),\ -sin\,\theta1*sin\,(\theta2+\theta3),\ cos\,\theta1,\ l3*sin\,\theta1*cos\,(\theta2+\theta3)+l2*sin\,\theta1*cos\,\theta2+d2*cos\,\theta1\ \},\\&\ \{\ -sin\,(\theta2+\theta3),\ -cos\,(\theta2+\theta3),\ 0,\ -l3*sin\,(\theta2+\theta3)-l2*sin\,\theta2+d1\ \},\\&\ \{\ 0,\ 0,\ 0,\ 1\ \}\}.\end{aligned}$$

引理 17：在 DH 坐标系中，若机器人 *rob* 的类型为 *RRR_type*，则该类型机器人的末端执行器在全局坐标系中的位置具有以下等价关系。

$$\begin{aligned}&Lemma\quad RRR\_Gr\_check:\ forall\ (rob:\ RRR\_type),\ getGr\ rob\ Br =\\&\{\{l3*cos\,\theta1*cos\,(\theta2+\theta3)+l2*cos\,\theta1*cos\,\theta2-d2*sin\,\theta1\},\\&\ \{l3*sin\,\theta1*cos\,(\theta2+\theta3)+l2*sin\,\theta1*cos\,\theta2+d2*cos\,\theta1\},\\&\ \{\ -l3*sin\,(\theta2+\theta3)-l2*sin\,\theta2+d1\ \},\\&\ \{\ 1\ \}\}.\end{aligned}$$

4.3.3 空间站机械手摇控系统

空间站机械手摇控系统(Space Station Remote Manipulator System, SSRMS)，它是依附在航天飞机或者空间站

上的机械手。图 8 为 SSRMS 的一个简化原理模型图。该机械手的 DH 参数如表 4 所示。

表 4： SSRMS 机器人 DH 参数表

坐标系编号	连杆长度	连杆扭转	关节距离	关节角
1	0	$-\pi/2$	d_1	θ_1
2	0	$-\pi/2$	d_2	θ_2
3	l_3	0	d_3	θ_3
4	l_4	0	d_4	θ_4
5	0	$\pi/2$	d_5	θ_5
6	0	$-\pi/2$	d_6	θ_6
7	0	0	d_7	θ_7

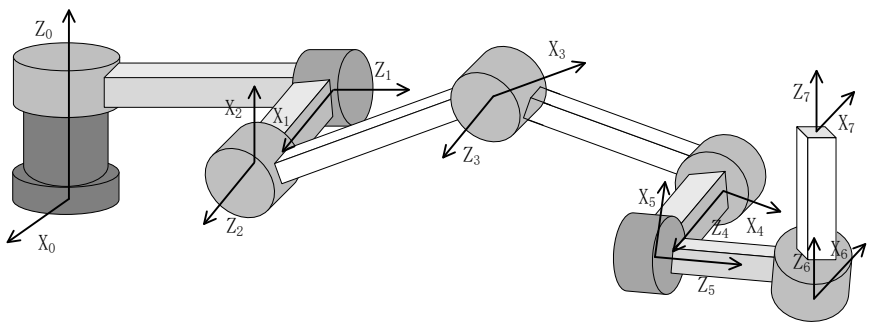

图 8 SSRMS机器人 DH 坐标系图

SSRMS机器人结构的形式化定义如 *SSRMS _type* 所示。

$$
\begin{aligned}
&Class\ SSRMS\_type : Type := \{\\
&\quad l3 : R; l4 : R;\\
&\quad d1 : R; d2 : R; d3 : R; d4 : R; d5 : R; d6 : R; d7 : R;\\
&\quad \theta 1 : R; \theta 2 : R; \theta 3 : R; \theta 4 : R; \theta 5 : R; \theta 6 : R; \theta 7 : R\ \}.
\end{aligned}
$$

在下述的 *Section* 中， *fun_SSRMS_T* 与 *fun_SSRMS* 分别为SSRMS机器人的转换矩阵与坐标转换函数的形式化定义。

```
Section  SSRMS.
  Variable rob: SSRMS_type.
  Let prms:=[(Build_DH_PRM  0 (-(PI/2)) d1 θ1);(Build_DH_PRM 0 (-(PI/2)) d2 θ2);
             (Build_DH_PRM l3 0 d3 θ3);(Build_DH_PRM l4 0 d4 θ4);
             (Build_DH_PRM 0 (PI/2) d5 θ5);(Build_DH_PRM 0 (-(PI/2)) d6 θ6);
             (Build_DH_PRM 0 0 d7 θ7)].
  Definition fun_SSRMS_T:=T 0 (length  prms) prms.
  Definition fun_SSRMS (Pb: mat 3 1):=
    let b0:= mnth Pb 0 0 in  let b1:= mnth Pb 1 0 in  let b2:= mnth Pb 2 0 in
    let b:={{b0},{b1},{b2},{1}}  in
    fun_SSRMS_T * b.
End  SSRMS.
```

同样地，SSRMS机器人在类型类 *fun_Get* 的实例化如 *get_SSRMS* 所示。

$$\#[export]\ Instance\quad get\_SSRMS : fun\_Get\ SSRMS\_type := \{$$
$$getT := fun\_SSRMS\_T;$$
$$getGr := fun\_SSRMS;\}$$

在 SSRMS 中，连杆 1 与连杆 2 为 $R \perp R(-\pi/2)$类型，连杆 3 与连杆 4 为 $R \parallel R(0)$类型，连杆 5 为 $R \perp R(\pi/2)$类型，连杆 6 为 $R \perp R(-\pi/2)$类型，连杆 7 为 $R \parallel R(0)$类型。因此 SSRMS 的转换矩阵如公式(7)所示。

$$T_7^0 = T_1^0 * T_2^1 * T_3^2 * T_4^3 * T_5^4 * T_6^5 * T_7^6$$

$$s.t.\begin{cases} T_1^0 = \begin{bmatrix} \cos\theta_1 & 0 & -\sin\theta_1 & 1 \\ \sin\theta_1 & 0 & \cos\theta_1 & 1 \\ 1 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T_2^1 = \begin{bmatrix} \cos\theta_2 & 0 & -\sin\theta_2 & 0 \\ \sin\theta_2 & 0 & \cos\theta_2 & 0 \\ 0 & -1 & 0 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T_3^2 = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & l_3 * \cos\theta_3 \\ \sin\theta_3 & \cos\theta_3 & 0 & l_3 * \sin\theta_3 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ T_4^3 = \begin{bmatrix} \cos\theta_4 & -\sin\theta_4 & 0 & l_4 * \cos\theta_4 \\ \sin\theta_4 & \cos\theta_4 & 0 & l_4 * \sin\theta_4 \\ 0 & 0 & 1 & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T_5^4 = \begin{bmatrix} \cos\theta_5 & 0 & \sin\theta_5 & 0 \\ \sin\theta_5 & 0 & -\cos\theta_5 & 0 \\ 0 & 1 & 0 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T_6^5 = \begin{bmatrix} \cos\theta_6 & 0 & -\sin\theta_6 & 0 \\ \sin\theta_6 & 0 & \cos\theta_6 & 0 \\ 0 & -1 & 0 & d_6 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ T_7^6 = \begin{bmatrix} \cos\theta_7 & -\sin\theta_7 & 0 & 0 \\ \sin\theta_7 & \cos\theta_7 & 0 & 0 \\ 0 & 0 & 1 & d_7 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{cases} \tag{7}$$

为验证公式（7）给出的 SSRMS 转换矩阵的正确性，本文采用形式化方法在引理 18 中对其进行验证，其中：*Hypothesis* 策略对 SSRMS 机器人的结构进行约束。

引理 18：在 DH 坐标系中，设机器人 *rob* 的类型为 *SSRMS_type*， *pl*1、*pl*2、*pl*3、*pl*4、*pl*5、*pl*6、*pl*7 为该机器人中的连杆，当连杆 *pl*1 为 $R \perp R(-\pi/2)$类型且连杆长度为 0 ,关节距离为 $d1$，关节角为 $\theta1$；连杆 *pl*2 为 $R \perp R(-\pi/2)$类型且连杆长度为 0 ,关节距离为 $d2$，关节角为 $\theta2$；连杆 *pl*3 为 $R \parallel R(0)$类型且连杆长度为 $l3$,关节距离为 $d3$，关节角为 $\theta3$；连杆 *pl*4 为 $R \parallel R(0)$类型且连杆长度为 $l4$,关节距离为 $d4$，关节角为 $\theta4$；连杆 *pl*5 为 $R \perp R(\pi/2)$类型且连杆长度为 0 ,关节距离为 $d5$，关节角为 $\theta5$；连杆 *pl*6 为 $R \perp R(-\pi/2)$类型且连杆长度为 0 ,关节距离为 $d6$，关节角为 $\theta6$；连杆 *pl*7 为 $R \parallel R(0)$类型且连杆长度为 0 ,关节距离为 $d7$，关节角为 $\theta7$，则该机器人的转换关系满足 $T_7^0 = T_1^0 * T_2^1 * T_3^2 * T_4^3 * T_5^4 * T_6^5 * T_7^6$。

Section SRMS.
Variable rob: SSRMS_type.
Variable pl1 pl2 pl3 pl4 pl5 pl6 pl7 : Pole.
Hypothesis H1 : pl1 = R_pi2' 0 d1 θ1. *Hypothesis H2 : pl2 = R_pi2' 0 d2 θ2.*
Hypothesis H3 : pl3 = R_0 l3 d3 θ3. *Hypothesis H4 : pl4 = R_0 l4 d4 θ4.*
Hypothesis H5 : pl5 = R_pi2 0 d5 θ5. *Hypothesis H6 : pl6 = R_pi2' 0 d6 θ6.*
Hypothesis H7 : pl7 = R_0 0 d7 θ7.
Let T1:= @fun_poleT pl1 0 d1 θ1. *Let T2 := @fun_poleT pl2 0 d2 θ2.*
Let T3:= @fun_poleT pl3 l3 d3 θ3. *Let T4 := @fun_poleT pl4 l4 d4 θ4.*
Let T5:= @fun_poleT pl5 0 d5 θ5. *Let T6 := @fun_poleT pl6 0 d6 θ6.*
Let T7:= @fun_poleT pl7 0 d7 θ7.
*Lemma SSRMS_T_check : getT rob = T1 * T2 * T3 * T4 * T5 * T6 * T7.*
Proof. … *Qed.*
End SRMS.

5 代码抽取与分析

使用形式化方法对机器人正向运动学进行定义和验证后，代码抽取可以将形式化语言转化为其它高性能语言，进而构建逻辑可靠、性能稳定的控制系统。本文将 Coq 中的函数抽取为 OCaml 语言的代码。值得注意的是，在代码抽取过程中，需要将 Coq 中的类型和目标语言之间建立类型映射。在本文中，将 Coq 中的实数类型（R）转换为 OCaml 中的“float”，自然数类型（nat）转换为 OCaml 中的“int”。图 9 显示了抽取的 OCaml 程序的接口信息，包括函数名、参数和返回值。本文选取了具有代表性的函数进行分析和验证。

```
val default_prm : dH_PRM = {l = 0.; alpha = 0.; d = 0.; theta = 0.}
val tnear : dH_PRM -> float list list = <fun>
val t : int -> int -> dH_PRM list -> float list list = <fun>
type rR_type = { l1 : float; l2 : float; theta1 : float; theta2 : float; }
val fun_RR_T : rR_type -> float list list = <fun>
val fun_RR : rR_type -> float list list -> float list list = <fun>
type rRR_type = { l0 : float; l3 : float; d1 : float; d2 : float;
                  theta0 : float; theta3 : float; theta4 : float;}
val fun_RRR_T : rRR_type -> float list list = <fun>
val fun_RRR : rRR_type -> float list list -> float list list = <fun>
type sSRMS_type = { l4 : float; l5 : float; d0 : float; d3 : float; d5 : float; d6 : float;
  d7 : float; d8 : float; d9 : float; theta5 : float; theta6 : float;
  theta7 : float; theta8 : float; theta9 : float; theta10 : float; theta11 : float; }
val fun_SSRMS_T : sSRMS_type -> float list list = <fun>
val fun_SSRMS : sSRMS_type -> float list list -> float list list = <fun>
```

图 9　OCaml 代码函数接口

表 5 显示了 SSRMS 机器人正向运动学方程的坐标求解函数 *fun_SSRMS*，该函数由经过形式化验证的 Coq 代码自动生成。*fun_SSRMS* 接收两个参数(*rob* 和 *pb*)，其中 *rob* 为 SSRMS 类型机器人的结构参数，*pb* 表示末端执行器在局部坐标系中的位置；该函数返回末端执行器在全局坐标系中的位置。需要注意的是，*MatrixR_DR.mmul* 表示实数类型的矩阵乘法运算需要 5 个自变量：前 3 个是运算矩阵大小的约束，而后 2 个是运算矩阵。此外，*Int.succ* 表示将整数值递增 1 的函数。因此，整数 4 被表示为 $Int.succ(Int.succ(Int.succ(Int.succ\ 0)))$。

表 5：　SSRMS 机器人正向运动学方程的坐标求解算法

```
let fun_SSRMS rob pb =
  let b0 = MatrixR_DR.DecMT.mnth
    (Int.succ (Int.succ (Int.succ 0))) (Int.succ 0) pb 0 0 in
  let b1 = MatrixR_DR.DecMT.mnth
    (Int.succ (Int.succ (Int.succ 0))) (Int.succ 0) pb (Int.succ 0) 0 in
  let b2 = MatrixR_DR.DecMT.mnth
    (Int.succ (Int.succ (Int.succ 0))) (Int.succ 0) pb (Int.succ (Int.succ 0)) 0 in
  let b = mat_4_1 b0 b1 b2 (iZR (Zpos XH)) in
  MatrixR_DR.mmul (Int.succ (Int.succ (Int.succ (Int.succ 0))))
                  (Int.succ (Int.succ (Int.succ (Int.succ 0)))) (Int.succ 0)
                   (fun_SSRMS_T rob) b
```

该代码可以通过一个特定的例子进行验证，假设 SSRMS 机器人的运动序列如示例 1 所示。

例 1：对于 SSRMS 机器人，假设 $l_3=10$，$l_4=8$，$d_1=1$，$d_2=2$，$d_3=3$，$d_4=4$，$d_5=5$，$d_6=6$，$d_7=7$，$\theta_1=\pi$，$\theta_2=\pi/2$，$\theta_3=\theta_4=\theta_5=\theta_6=\theta_7=\pi/4$，由图 8 可知，末端执行器在坐标系 B_7 中的坐标为 $P_r=[0\quad 0\quad 0]^T$，

可以得出末端执行器在 B_0 中的位置为 $\left[12+7\sqrt{2}/2 \quad 5/2+8\sqrt{2} \quad -5/2-8\sqrt{2}\right]^T \approx [16.95 \quad 13.81 \quad -13.81]^T$。

图 10 显示了在提取的 OCaml 代码中 *fun_SSRMS* 函数使用例 1 中数据的计算结果。首先我们使用 let 构建 *sSRMS_ type* 类型的数据 *rob*，然后调用 *fun_SSRMS* 函数进行计算，所示的结果与预期一致。

```
-( 16:52:50 )-< command 1 >-------------------------------------------------------------------------
utop # let rob={l4=10.0; l5=8.0; d0=1.0; d3=2.0; d5=3.0; d6=4.0; d7=5.0; d8=6.0; d9=7.0;
theta5=(Float.pi); theta6=(Float.pi/.2.0); theta7=(Float.pi/.4.0); theta8=(Float.pi/.4.0);
theta9=(Float.pi/.4.0); theta10=(Float.pi/.4.0); theta11=(Float.pi)/.4.0};;
val rob : sSRMS_type =
  {l4 = 10.; l5 = 8.; d0 = 1.; d3 = 2.; d5 = 3.; d6 = 4.; d7 = 5.; d8 = 6.;
   d9 = 7.; theta5 = 3.14159265358979312; theta6 = 1.57079632679489656;
   theta7 = 0.785398163397448279; theta8 = 0.785398163397448279;
   theta9 = 0.785398163397448279; theta10 = 0.785398163397448279;
   theta11 = 0.785398163397448279}
-( 16:52:52 )-< command 2 >-------------------------------------------------------------------------
utop # fun_SSRMS rob [[0.0];[0.0];[0.0]];;
- : float list list =
[[16.9497474683058336]; [13.8137084989847523]; [-13.8137084989847665]; [1.]]
```

图 10 *fun_SSRMS* 函数结果

6 总结与展望

本文设计了基于 DH 坐标系的机器人运动学形式化验证框架，在 Coq 中构造了机器人正向运动理论的形式化证明，并验证了控制算法的正确性。首先，对机器人运动学理论进行了形式化建模，构造了基于 DH 标定的机器人在不同坐标系下的变换，并完成了相关性质的证明。其次，构建了几种常见的机器人模型结构，并设计和验证了坐标变换算法。最后，将 Coq 代码提取为 OCaml 代码，进行数据验证。本文所有证明和验证都是使用 Coq 实现的，所有源文件均在 GitHub 中开源。开源地址：https://github.com/GuojunXie123/FVFK.git。

总的来说，本文形式化验证脚本包含 3000 余行代码，该代码在 Coq 8.16 下进行编译。表 6 提供了关于脚本文件的详细说明，为了使描述更为清晰，列举了文中各部分与 Coq 文件的对应关系，并统计了各文件代码量与证明脚本的行数。

表 6： 形式化验证代码统计

文件（夹）	章节	规格	证明
Msupp	3.2、3.3、4.1、4.2、4.3	2300	1600
Forward_Kinematics/D4	3.2、3.3、4.1、4.2、4.3	60	10
Forward_Kinematics/R4	3.2、3.3、4.1、4.2、4.3	80	20
Forward_Kinematics/DH.v	3.2、3.3、4.1	138	70
Forward_Kinematics/B2G.v	3.3	160	120
Forward_Kinematics/DH_Arm.v	4.2	150	90
Forward_Kinematics/RR_robot.v	4.3.1	50	30
Forward_Kinematics/RRR_robot.v	4.3.2	60	40
Forward_Kinematics/SSRMS_robot.v	4.3.3	120	90

未来工作主要包括以下两点：一是扩展本文设计的形式化验证框架，以涵盖更多的机器人控制技术，并针对更复杂的运动学逻辑进行改进，开发形式化证明的自动化策略；二是将自动编程技术与形式化验证相结合，将机器人控制算法自动生成 C 代码，进而提高机器人控制系统的安全性以及推动自动化控制系统验证的发展。

References:

[1] Hichri B, Gallala A, Giovannini F. Mobile robots path planning and mobile multirobots control: A review[J]. Robotica, 2022: 1-14.

[2] Tan Y, Ma D, Qiao L. A Formal Verification Method of Compilation Based on C Safety Subset[J]. Wireless Communications and Mobile Computing, 2021, 2021: 1-10.

[3] Zhou S, Wang J, Jia J. A formal verification method for the SOPC software[J]. IEEE Transactions on Reliability, 2022, 71(2): 818-829.

[4] Schreiber L T, Gosselin C. Determination of the Inverse Kinematics Branches of Solution Based on Joint Coordinates for Universal Robots-Like Serial Robot Architecture[J]. Journal of Mechanisms and Robotics, 2022, 14(3): 034501.

[5] Elqortobi M, El-Khouly W, Rahj A. Verification and testing of safety-critical airborne systems: A model-based methodology[J]. Computer Science and Information Systems, 2020, 17(1): 271-292.

[6] Sakaguchi K. Program extraction for mutable arrays[J]. Science of Computer Programming, 2020, 191: 102372.

[7] Annenkov D, Milo M, Nielsen J B. Extracting functional programs from Coq, in Coq[J]. Journal of Functional Programming, 2022, 32: e11.

[8] Vicentini F, Askarpour M, Rossi M Gl. Safety assessment of collaborative robotics through automated formal verification[J]. IEEE Transactions on Robotics, 2019, 36(1): 42-61.

[9] Isobe Y, Miyamoto N, Ando N. Formal Modeling and Verification of Concurrent FSMs: Case Study on Event-Based Cooperative Transport Robots[J]. IEICE TRANSACTIONS on Information and Systems, 2021, 104(10): 1515-1532.

[10] Lestingi L, Bersani M M, Rossi M. Model-Driven Development of Service Robot Applications Dealing With Uncertain Human Behavior[J]. IEEE Intelligent Systems, 2022, 37(6): 48-56.

[11] Askarpour M, Mandrioli D, Rossi M. Formal model of human erroneous behavior for safety analysis in collaborative robotics[J]. Robotics and computer-integrated Manufacturing, 2019, 57: 465-476.

[12] Praveen A T, Gupta A, Bhattacharyya S. Assuring behavior of multirobot autonomous systems with translation from formal verification to ROS simulation[J]. IEEE Systems Journal, 2022, 16(3): 5092-5100.

[13] Martin-Martin E, Montenegro M, Riesco A. Verification of the ROS NavFn planner using executable specification languages[J]. Journal of Logical and Algebraic Methods in Programming, 2023, 132: 100860.

[14] Dal Zilio S, Hladik P E, Ingrand F. A formal toolchain for offline and run-time verification of robotic systems[J]. Robotics and Autonomous Systems, 2023, 159: 104301.

[15] Bohrer R, Tan Y K, Mitsch S. A formal safety net for waypoint-following in ground robots[J]. IEEE Robotics and Automation Letters, 2019, 4(3): 2910-2917.

[16] Foughali M, Hladik P E. Bridging the gap between formal verification and schedulability analysis: The case of robotics[J]. Journal of Systems Architecture, 2020, 111: 101817.

[17] Paul S, Cruz E, Dutta A. Formal Verification of Safety-Critical Aerospace Systems[J]. IEEE Aerospace and Electronic Systems Magazine, 2023.

[18] Xie G, Yang H, Deng H. Formal Verification of Robot Rotary Kinematics[J]. Electronics, 2023, 12(2): 369.

[19] López J, Santana-Alonso A, Diaz-Cacho Medina M. Formal verification for task description languages. a petri net approach[J]. Sensors, 2019, 19(22): 4965.

[20] Sangnier A, Sznajder N, Potop-Butucaru M. Parameterized verification of algorithms for oblivious robots on a ring[J]. Formal Methods in System Design, 2020, 56: 55-89.

[21] Evangelidis A, Parker D. Quantitative verification of Kalman filters[J]. Formal Aspects of Computing, 2021, 33(4-5): 669-693.

[22] Abd Alrahman Y, Piterman N. Modelling and verification of reconfigurable multi-agent systems[J]. Autonomous Agents and Multi-Agent Systems, 2021, 35(2): 47.

[23] Rashid A, Hasan O. Formal analysis of the continuous dynamics of cyber–physical systems using theorem proving[J]. Journal of Systems Architecture, 2021, 112: 101850.

[24] Wang G, Chen S, Guan Y. Formalization of the inverse kinematics of three-fingered dexterous hand[J]. Journal of Logical and Algebraic Methods in Programming, 2023, 133: 100861.

[25] Murray Y, Sirevåg M, Ribeiro P. Safety assurance of an industrial robotic control system using hardware/software co-verification[J]. Science of Computer Programming, 2022, 216: 102766.

[26] Mkaouar H, Zalila B, Hugues J. A formal approach to AADL model-based software engineering[J]. International Journal on Software Tools for Technology Transfer, 2020, 22: 219-247.

[27] Sakata K, Fujita S, Sawada K. Model verification of fallback control system under cyberattacks via UPPAAL[J]. Advanced Robotics, 2023, 37(3): 156-168.

[28] Menghi C, Tsigkanos C, Askarpour M. Mission specification patterns for mobile robots: Providing support for quantitative properties[J]. IEEE Transactions on Software Engineering, 2022.

[29] Arcile J, Devillers R, Klaudel H. VerifCar: a framework for modeling and model checking communicating autonomous vehicles[J]. Autonomous agents and multi-agent systems, 2019, 33: 353-381.

[30] Pek C, Manzinger S, Koschi M. Using online verification to prevent autonomous vehicles from causing accidents[J]. Nature Machine Intelligence, 2020, 2(9): 518-528.

[31] Kabra A, Mitsch S, Platzer A. Verified train controllers for the federal railroad administration train kinematics model: balancing competing brake and track forces[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2022, 41(11): 4409-4420.

[32] Ma Y Y, Ma Z W, Chen G. Formalization of Operations of Block Matrix Based on Coq[J]. Journal of Software, 2021, 32(6): 1882-1909.

[33] Shi Z P, Chen G. Integration of Multiple Formal Matrix Models in Coq[C]//International Symposium on Dependable Software Engineering: Theories, Tools, and Applications. Cham: Springer Nature Switzerland, 2022: 169-186.

[34] Boldo S, Lelay C, Melquiond G. Coquelicot: A user-friendly library of real analysis for Coq[J]. Mathematics in Computer Science, 2015, 9: 41-62.

[35] Blanqui F, Koprowski A. CoLoR: a Coq library on well-founded rewrite relations and its application to the automated verification of termination certificates[J]. Mathematical Structures in Computer Science, 2011, 21(4): 827-859.

[36] Pous D. Untyping typed algebras and colouring cyclic linear logic[J]. Logical Methods in Computer Science, 2012, 8.

[37] Shi Z P, Xie G J, Chen G. CoqMatrix: Formal matrix library with multiple models in Coq [J]. Journal of Systems Architecture. https://doi.org/10.1016/j.sysarc.2023.102986.

附中文参考文献:

[32] 麻莹莹,马振威,陈钢.基于 Coq 的分块矩阵运算的形式化[J]. 软件学报, 2021, 32(6): 1882-1909.