

东 莞 理 工 学 院

本 科 毕 业 论 文

**毕业设计题目： 基于 ARM 的六足行走机器人设计
与实现**

学生姓名： 李海亮

学 号： 201941417520

院 系： 计算机科学与技术学院

专业班级： 19 级计算机科学与技术 5 班

指导教师姓名及职称： 谢仁平 副研究员

起止时间： 2022 年 12 月—2023 年 5 月

摘要

足式机器人具备高移动能力、高稳定性、高灵活性等特点，非常适合作为全向移动的运载平台，在城市、室内、洞穴等复杂环境场景中具备广阔的应用前景。相比于传统的履带式机器人或轮式机器人，六足机器人灵活性较强，能适应各种复杂多变的地形，因此成为当前机器人研究的热点。本文以六足机器人为研究对象，结合六足机器人的运动特性，对机器人进行步态规划以及运动学建模，进而实现六足机器人的全向移动与转弯算法设计。再通过 matlab 进行相关的仿真实验，并通过硬件设计、嵌入式软件设计以及机器人整机组装完成机器人实物搭建，并通过实物检验运动学建模以及步态分析算法的正确性。本文主要研究工作如下：

(1) 运动学解算与步态控制。对六足机器人结构进行简化分析，并通过 DH 参数法建立六足机器人运动学模型，将机器人机体和腿部联系起来，对机器人足端进行正逆运动学分析以及机器人机身六自由度控制，为后续机身六自由度控制和步态规划提供理论基础。与市面上大多使用动作组来控制六足机器人不同，本文采用自主设计的实时解算的方法进行六足机器人控制，便于实现全向移动、步幅可控、机身姿态可调等功能，这大大提高了其地形适应能力。

(2) 六足机器人软硬件设计。分析硬件需求，完成硬件选型以及 pcb 设计，通过 pcb 连接各个硬件，并在 FreeRTOS 操作系统上完成嵌入式软件设计。

(3) 整机组装及实验验证。完成六足机器人的整机组装并进行多种实验，验证本文运动学算法的正确性。本文完成了六足机器人的全向行走测试、全向行走的同时转弯功能测试、六足机器人机身的姿态控制及位置控制测试、姿态平衡测试，测试效果良好。

关键词： 六足机器人、运动学、硬件设计、整机实验、仿真

ABSTRACT

The legged robot has the characteristics of high mobility, high stability and high flexibility. It is very suitable for being an omnidirectional mobile carrier platform, and has a broad application prospect in complex environment scenes such as city, indoor and cave. Compared with traditional tracked robot or wheeled robot, hexapod robot is more flexible and can adapt to a variety of complex terrain, so it has become a hot topic in robot research. In this paper, hexapod robot as the research object, combined with the movement characteristics of hexapod robot, the robot gait planning and kinematic modeling, and then realize the design of hexapod robot omnidirectional movement and turning algorithm. Then through the matlab simulation experiment, and through the hardware design, embedded software design and robot machine assembly to complete the robot physical construction, and through the physical test kinematics modeling and the correctness of the gait analysis algorithm. The main research work of this paper is as follows:

(1) Kinematics solution and gait control. The structure of the hexapod robot was simplified and analyzed, and the kinematics model of the hexapod robot was established by DH parameter method. The robot body and legs were connected, and the forward and inverse kinematics analysis of the robot foot and the six degrees of freedom control of the robot fuselage were carried out, which provided the theoretical basis for the subsequent six degrees of freedom control of the fuselage and gait planning.

(2) Hardware and software design of hexapod robot. Analyzed hardware requirements, completed hardware selection and pcb design, connected each hardware through pcb, and completed embedded software design on FreeRTOS operating system.

(3) Complete machine assembly and experimental verification. The assembly of the hexapod robot is completed and a variety of experiments are carried out to verify the correctness of the kinematics algorithm in this paper. This paper has completed the omni-walk test of the hexapod robot, the simultaneous turning function test of omni-walk, the attitude control and position control test of the fuselage of the hexapod robot, and the attitude balance test, and the test results are good.

KEY WORDS: hexapod, kinematics, hardware design, machine test, simulation

目 录

第一章 绪论	1
1.1 研究背景及意义	1
1.2 六足机器人的国内外研究现状	1
1.2.1 国外研究现状	1
1.2.2 国内研究现状	2
1.3 论文的章节编排	2
第二章 运动学建模	4
2.1 对单腿进行建模	4
2.1.1 正运动学解算	5
2.1.2 逆运动学解算	6
2.1.3 机身六自由度控制	8
2.1.3.1 姿态控制	8
2.1.3.2 位置控制	11
第三章 步态控制	12
3.1 常见步态	12
3.1.1 二步态	12
3.1.2 四步态	12
3.1.3 六步态	12
3.2 不同斜坡环境下的步态规划	13
3.3 全向移动与转弯	13
3.4 二步态 matlab 仿真	15
3.4.1 前进步态仿真	16
3.4.2 右移步态仿真	17
第四章 硬件设计	18
4.1 主控选择	18
4.1.1 stm32H750VBT6 简介	18
4.1.1.1 内核	18
4.1.1.2 存储	18
4.2 核心板选择	18
4.3 舵机选型	19
4.4 转压电路	19

4.5 供电选择	20
4.6 遥控接收	20
4.7 陀螺仪	21
4.8 原理图	21
4.9 PCB	22
4.10 PCB 实物	23
4.11 整机实物图	24
第五章 软件设计	25
5.1 BootLoader	25
5.2 嵌入式操作系统	25
5.3 代码结构	25
5.4 软件系统框图	26
第六章 六足机器人实验	27
6.1 机身位置控制实验	27
6.1.1 控制机身位置 Z 轴	27
6.1.2 控制机身位置 Y 轴	28
6.1.3 控制机身位置 X 轴	28
6.2 机身姿态控制实验	28
6.2.1 控制机身姿态 Z 轴	28
6.2.2 控制机身姿态 Y 轴	29
6.2.3 控制机身姿态 X 轴	29
6.3 姿态平衡实验	30
第七章 总结与展望	31
7.1 总结	31
7.2 展望	31
参考文献	32
致 谢	33
附 录	34

第一章 绪论

1.1 研究背景及意义

随着科学技术的不断发展，机器人的应用场景不仅局限于室内环境，也会越来越多的被应用在室外场景中。随着家用机器人的普及，对移动机器人的需求越来越大，功能需求也越来高^[1]。移动机器人主要分为履带机器人，轮式机器人，腿式机器人，以及轮腿式机器人等。各种类型的机器人都有各自的特点和优势，在农业、工业、运输业，建筑业等领域都有着广泛的应用^[10]。

随着人们对于复杂环境下具备高移动能力，高稳定性，高灵活性的移动平台的需求的增加，腿式机器人作为一种拥有全向移动能力的运载平台，在复杂环境如洞穴、废墟、山地以及城市巷战中具有广阔的应用前景^[9]。在乌克兰战场上已经投入了四足机器人进行排雷工作，在降低人员伤亡的同时大大提高排雷效率。

六足机器人具有丰富的步态和冗余的肢体结构，相比于传统机器人，如履带式机器人和轮式机器人，具有更灵活的移动步态，能够适应复杂的地理环境，具有非常高的可靠性^[8]。基于上述分析以及描述，本文以六足机器人为研究对象，对其依次进行运动学建模、步态规划、硬件设计以及对应的嵌入式软件设计，制作出六足机器人。

1.2 六足机器人的国内外研究现状

1.2.1 国外研究现状

国外的六足机器人起步时间较早，取得的成绩较为丰富。在 20 世纪 80 年代，美国为了进行深空探测，对六足机器人进行了深入研究，研发了具备多功能的各种类型机器人，如机器人 Genghis（图 1-1）以及 Hannibal（图 1-2）。



图 1-1 Genghis 机器人

Hannibal 机器人具有 6 个三轴的腿和 1 轴的身体，总共 19 个自由度，腿上有很多传感器，并配备了多台计算机，为机器人提供了地形信息，使机器人能正在复杂地形上移动。

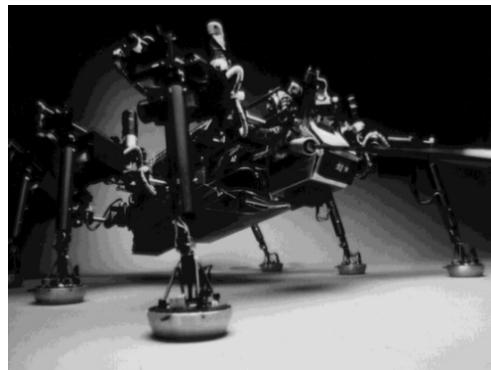


图 1-2 Hannibal 机器人

1.2.2 国内研究现状

国内六足机器人起步时间较晚, 但国内在六足机器人上的研究也取得了不少成果。哈尔滨工业大学设计出了六足全机器人 HITCR (图 1-3), 该系列机器人的腿部使用 4 连杆结构, 使机器人具备更加灵活的性能。使机器人能在不规则地形灵活行走。



图 1-3 HITCR 机器人

市面上的六足机器人, 大多使用动作组来控制六足机器人, 这大大降低了其地形适应的能力。因此本文对于六足机器人, 使用实时解算的方法进行控制, 便于实现全向移动, 步幅可控, 机身可调姿态等功能。

1.3 论文的章节编排

本文主要是通过运动学算法, 完成对六足机器人的运动学建模, 同时进行硬件设计和嵌入式软件设计, 将机器人进行组装并进行验证实验。

本论文的章节安排如下:

第 1 章是引言, 简要介绍本课题研究背景和意义, 并分析国内外六足机器人研究现状。

第 2 章运动学建模是核心章节, 主要根据六足机器人的几何特征, 完成正逆运动学解算以及机器人机身六自由度控制。

第 3 章是步态控制，说明了六足机器人常用步态，并完成了六足机器人全向移动与转弯的算法描述。

第 4 章是硬件设计，完成了硬件选型以及 `pcb` 设计，通过 `pcb` 连接各个硬件，完成机器人整机的组装。

第 5 章是软件设计，简单描述了软件启动过程以及软件代码框架。

第 6 章是实验部分，用于检验运动学建模以及步态分析算法的正确性。

第 7 章是总结和展望，用于对本文进行总结并阐述待改进的地方。

第二章 运动学建模

六足机器人运动学分析就是将空间直角坐标系建立在机器人腿部的关节上，将腿部各关节之间的间距，关节的夹角进行关系转换，求解其位置已及姿态矩阵矩阵，从而建立机器人的运动学方程。为了确定每个关节上坐标系之间的关系，需要一种合适的方法进行运动学分析。本文使用 D-H 建模法，该方法使用其次变换矩阵来描述机械臂上各个连杆之间的空间关系。每一个关节都可以通过一个四阶的齐次变换矩阵表示，按照连杆顺序对齐次变换矩阵相乘，从而得出首末坐标系之间的关系，构建一支运动学坐标系。

2.1 对单腿进行建模

首先以腿部为原点，如图 2-1 所示，以腿部起始端为原点，建立空间直角坐标系。

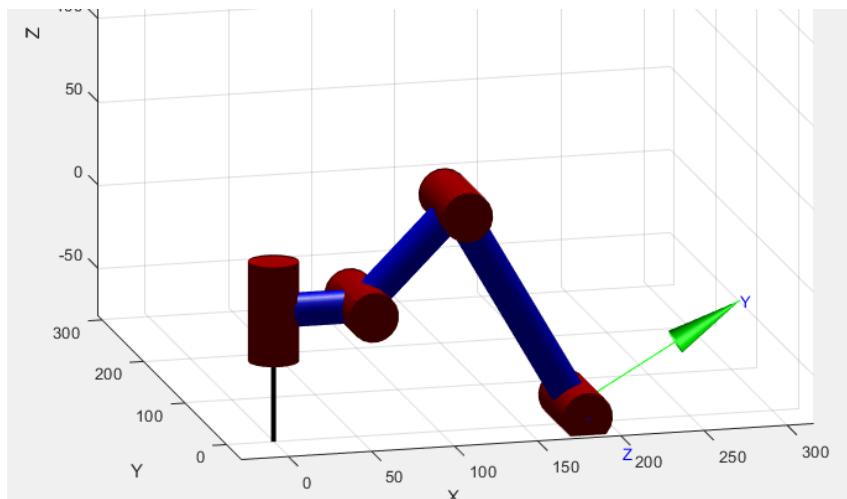


图 2-1 腿部建模

建模方法使用 D—H 建模法，该方法主要应用于机器人运动学，在每个连杆上建立一个坐标系，通过齐次坐标变换实现两个连杆坐标的变换，多次使用其次坐标变换，便可建立首末坐标系的关系^[3]。使用 D-H 建模法，即可构件一支运动学坐标系。

以腿部起始端为 0 系，“起始端-舵机 1”为 1 系，“舵机 1-舵机 2”为 2 系，“舵机 2-舵机 3”为 3 系，“舵机 3-腿末端”为 4 系。记舵机 1 到舵机 2 的轴间距为 L_1 ，舵机 2 到舵机 3 的轴间距为 L_2 ，舵机 3 到腿末端的轴间距为 L_3 ，则 D-H 参数如表 2-1 所示。

记 a 系下 b 系的坐标为 P_a^b ，则 $P_a^b = \begin{bmatrix} x_a^b \\ y_a^b \\ z_a^b \\ 1 \end{bmatrix}$

表 2-1 D-H 参数表

i	θ_i	d_i	a_{i-1}	α_{i-1}
1	θ_1	0	0	0
2	θ_2	0	L_1	$\pi/2$
3	θ_3	0	L_2	0
4	0	0	L_3	0

记从 a 系到 b 系的齐次变换矩阵为 T_a^b , 则

$$\begin{aligned}
 T_0^1 &= \begin{pmatrix} \cos(\theta_1) & -\sin(\theta_1) & 0 & 0 \\ \sin(\theta_1) & \cos(\theta_1) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\
 T_1^2 &= \begin{pmatrix} \cos(\theta_2) & -\sin(\theta_2) & 0 & L_1 \\ 0 & 0 & -1 & 0 \\ \sin(\theta_2) & \cos(\theta_2) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\
 T_2^3 &= \begin{pmatrix} \cos(\theta_3) & -\sin(\theta_3) & 0 & L_2 \\ \sin(\theta_3) & \cos(\theta_3) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\
 T_3^4 &= \begin{pmatrix} 1 & 0 & 0 & L_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}
 \end{aligned} \tag{2-1}$$

2.1.1 正运动学解算

正运动学解算过程是由舵机关节 $\theta_1, \theta_2, \theta_3$ 求解坐标系下腿部末端向量 P 的过程, 同时也是逆运动学的基础。首先求得 T_0^4 , 随后取该矩阵第四列, 即可完成正运动学解算。由齐次变换矩阵的对角相消, 可得:

$$T_0^4 = T_0^1 T_1^2 T_2^3 T_3^4 \tag{2-2}$$

记矩阵 T_0^4 的第四列为 $P_0^4, P_0^4 = \begin{bmatrix} x_0^4 \\ y_0^4 \\ z_0^4 \\ 1 \end{bmatrix}$ 其中:

$$\begin{aligned} x_0^4 &= \cos\theta_1(L_1 + L_3\cos(\theta_2 + \theta_3) + L_2\cos\theta_2) \\ y_0^4 &= \sin\theta_1(L_1 + L_3\cos(\theta_2 + \theta_3) + L_2\cos\theta_2) \\ z_0^4 &= L_3\sin(\theta_2 + \theta_3) + L_2\sin(\theta_2) \end{aligned} \quad (2-3)$$

2.1.2 逆运动学解算

逆运动学解算基于正运动学, 即已知腿部末端的位置, 求得这条腿上各个关节的角度, 常见方法有数值解和封闭解^[2], 3 自由度的机械腿较为简单, 因此这里使用封闭解对其进行逆运动学解算。这里设腿末端坐标 $P = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$ 。

解法 1: 左乘逆变换矩阵^[4]

$$(T_0^1)^{-1}T_0^4 = T_1^2T_2^3T_3^4 \quad (2-4)$$

解方程(2-4) 得:

$$\begin{aligned} \theta_1 &= \text{atan}2(y, x) \\ \theta_2 &= \text{acos} \left(\frac{L_2^2 - L_3^2 + (L_1 - \sqrt{x^2 + y^2})^2 + z^2}{2L_2\sqrt{(L_1 - \sqrt{x^2 + y^2})^2 + z^2}} \right) - \text{atan} \left(\frac{z}{L_1 - \sqrt{x^2 + y^2}} \right) \\ \theta_3 &= -\text{acos} \left(\frac{L_2^2 - L_3^2 + (L_1 - \sqrt{x^2 + y^2})^2 + z^2}{2L_2\sqrt{(L_1 - \sqrt{x^2 + y^2})^2 + z^2}} \right) - \text{acos} \left(\frac{L_3^2 - L_2^2 + (L_1 - \sqrt{x^2 + y^2})^2 + z^2}{2L_3\sqrt{(L_1 - \sqrt{x^2 + y^2})^2 + z^2}} \right) \end{aligned} \quad (2-5)$$

解法 2: 几何法

如图 2-2 所示, 设机械腿在 XOY 平面上的投影模长为 R, 由三角关系, 易得:

$$\begin{aligned} \theta_1 &= \text{atan}2(y, x) \\ R &= \sqrt{x^2 + y^2} \end{aligned} \quad (2-6)$$

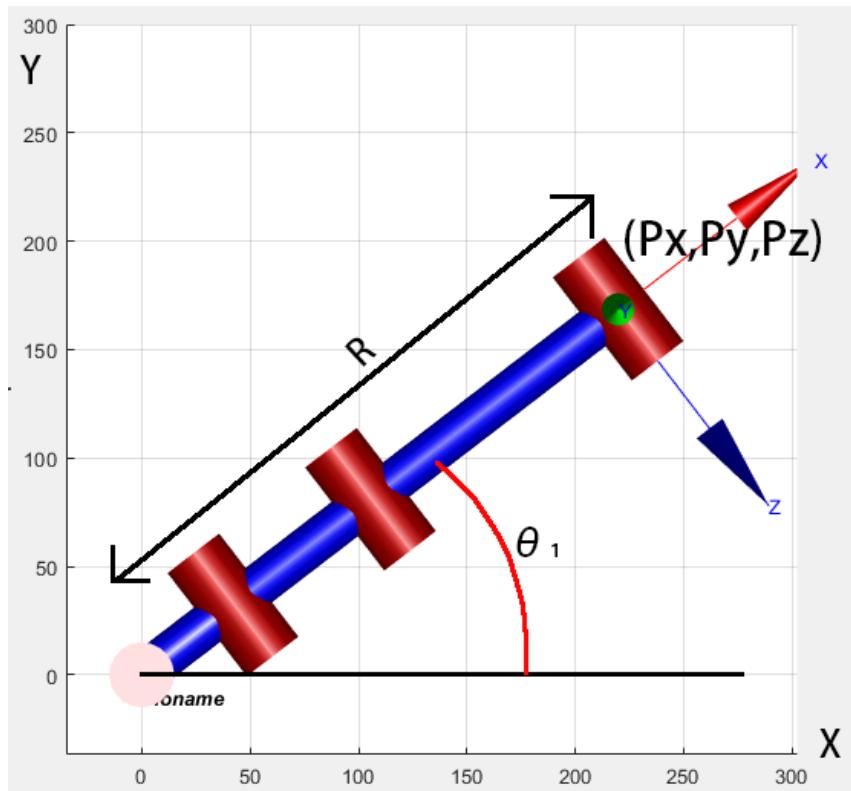


图 2-2 机械腿俯视图

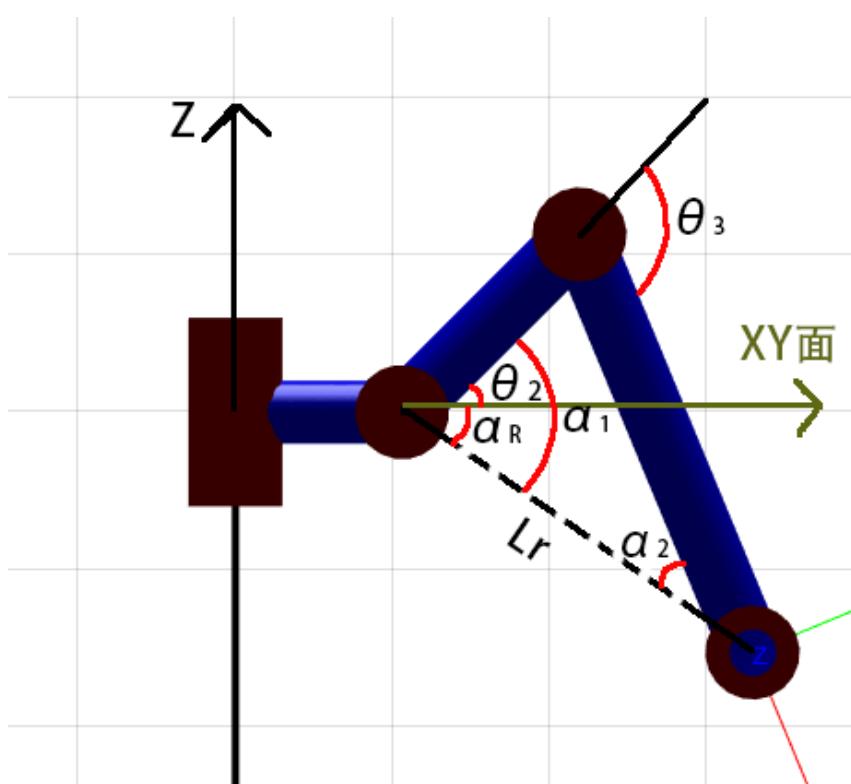


图 2-3 机械腿正视图

如图 2-3 再借由 α_1 和 α_R 求的第二关节角 θ_2 。

$$\begin{aligned}\alpha_R &= \text{atan}\left(\frac{-z}{R - L_1}\right) \\ L_r &= \sqrt{z^2 + (R - L_1)^2} \\ \alpha_1 &= \text{acos}\left(\frac{(L_2)^2 + (L_r)^2 - (L_3)^2}{2L_r L_2}\right) \quad (\text{余弦定理}) \\ \theta_2 &= \alpha_1 - \alpha_R\end{aligned}\tag{2-7}$$

最后借助 α_2 求的第三个关节角 θ_3 。

$$\begin{aligned}\alpha_2 &= \text{acos}\left(\frac{(L_r)^2 + (L_3)^2 - (L_2)^2}{2L_r L_3}\right) \\ \theta_3 &= \alpha_1 + \alpha_2\end{aligned}\tag{2-8}$$

最终解与(2-5)相同，明显几何法更为直观。

2.1.3 机身六自由度控制

机身六自由度控制，即腿末端坐标相对于地面静止的情况下，控制机身中心点位于空间中的位置以及姿态。

2.1.3.1 姿态控制

这里使用 Z-Y-X 欧拉角描述机身的姿态。即首先将坐标系 B 和一个已知坐标系 A 重合重合，先绕坐标系 B 的 Z 轴旋转 α 角，再绕着旋转后的坐标系 B 的 Y 轴旋转 β 角，最后在绕着旋转后的坐标系 B 的 X 轴旋转 γ 角^[5]。记绕 Z 轴旋转的齐次变换算子为 T_Z ，绕 Y 轴旋转的齐次变换算子为 T_Y ，绕 X 轴旋转的齐次变换算子为 T_X ，则

$$T_Z = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) & 0 & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2-9}$$

$$T_Y = \begin{pmatrix} \cos(\beta) & 0 & \sin(\beta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$T_X = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\gamma) & -\sin(\gamma) & 0 \\ 0 & \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (2-10)$$

$$T_A^B = T_Z T_Y T_X$$

连乘后, 得:

$$T_A^B = \begin{pmatrix} C(\alpha) C(\beta) & C(\alpha) S(\beta) & S(\gamma) - C(\gamma) S(\alpha) & S(\alpha) S(\gamma) + C(\alpha) C(\gamma) S(\beta) & 0 \\ C(\beta) S(\alpha) & C(\alpha) C(\gamma) + S(\alpha) S(\beta) S(\gamma) & C(\gamma) S(\alpha) S(\beta) - C(\alpha) S(\gamma) & 0 \\ -S(\beta) & C(\beta) S(\gamma) & C(\beta) C(\gamma) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (2-11)$$

为节省篇幅, 这里使用 C 代指 \cos , 使用 S 代指 \sin 。

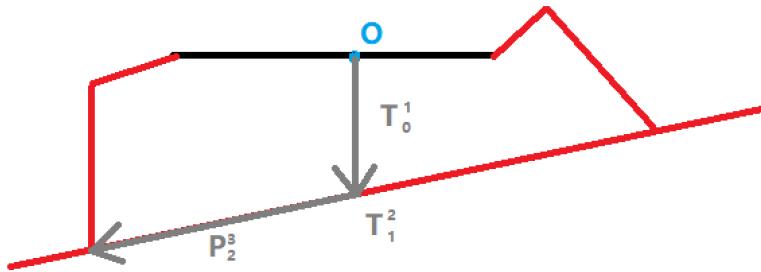


图 2-4 姿态解算

设机器人底盘中心为原点, 在腿末端坐标相对于地面静止的情况下, 可以看作是地面相对于机器人底盘进行旋转^[7], 如图 2-4 所示, T_0^1 为机器人底盘中心平移到地面的齐次变换矩阵, T_1^2 为地面旋转相对与机器人底盘旋转的齐次变换矩阵, P_2^3 为地面到机械腿末端的向量。 T_0^1 与 P_2^3 由机器人在水平站立的情况下, 底盘中心到机械腿起始端的向量以及机械腿起始端到末端的向量决定。记此时底盘中心到机械腿起始端的向量为

$$P_L, P_L = \begin{bmatrix} x_L \\ y_L \\ z_L \end{bmatrix}, \text{ 水平站立时机械腿起始端到末端的向量为 } P_S, P_S = \begin{bmatrix} x_s \\ y_s \\ z_s \end{bmatrix} \text{ 则:}$$

$$T_0^1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & z_e \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$T_1^2 = T_A^B$$

$$P_2^3 = \begin{pmatrix} x_L + x_s \\ y_L + y_s \\ 0 \\ 1 \end{pmatrix}$$
(2-12)

记此时机器人底盘中心到足末端的向量为 P_0^3 , 则:

$$P_0^3 = T_0^1 T_1^2 P_2^3$$
(2-13)

得到 P_0^3 后, 如图 2-5 通过向量加减, 即可得到在此姿态下的腿起始端到末端的向量 P_W , 记底盘中心到腿起始端向量为 P_L :

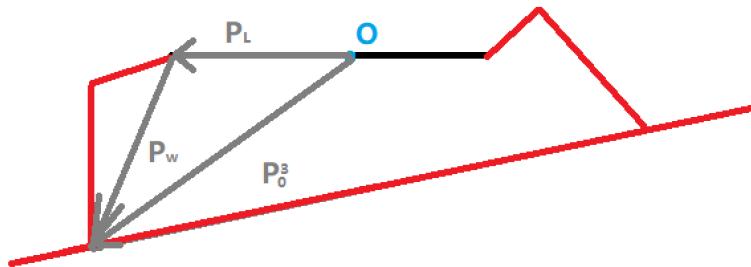


图 2-5 姿态解算

$$P_W = P_L - P_0^3$$
(2-14)

得到 P_W 后, 即可通过逆运动解算, 求得机械腿各个关节的角度, 对机器人六条腿完成解算, 完成 3 自由度姿态控制。

2.1.3.2 位置控制

如图 2-6, 以机器人正常站立时的底盘中心建立空间参考系, 记机身位置, 也就是底盘中心位置为 P_B 。

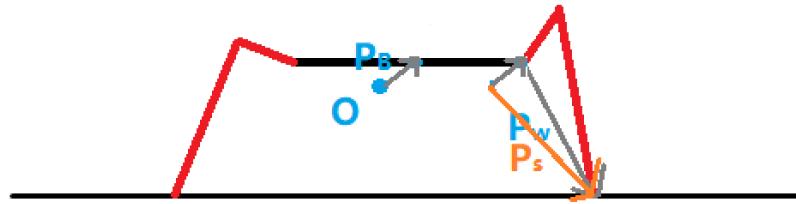


图 2-6 位置解算

控制机身移动时, 机身位置 $P_B = \begin{bmatrix} x_b \\ y_b \\ z_b \end{bmatrix}$, P_S 为正常站立时, 机械腿起始端到末端的向量, P_W 为移动机身后, 机械腿起始端到末端的向量。如图 2-6 所示, 通过向量加减, 即可获得 P_W 。

$$P_W = P_S - P_B \quad (2-15)$$

得到 P_W 后, 即可通过逆运动解算, 求得机械腿各个关节的角度, 对机器人六条腿完成解算, 完成 3 自由度位置控制。

将姿态控制与位置控制混合后, 完成六自由度机身控制。

第三章 步态控制

3.1 常见步态

六足机器人常见步态有二步态，四步态，以及六步态^[6]。支撑相又被称为站立相，为足底和地面接触的阶段。摆动相又被称为迈步相，指足端离开地面向前摆动的阶段。

3.1.1 二步态

二步态也叫三角步态，腿 L_1, L_3, L_5 为一组与腿 L_2, L_4, L_6 为一组，两组交替支撑，图 3-1 中红色为支撑腿，黑色表示支撑相。

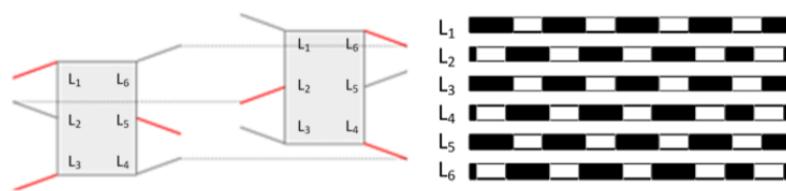


图 3-1 二步态示意图

3.1.2 四步态

四步态也叫四角步态，如图 3-2 所示，腿 L_1, L_5 为一组，腿 L_2, L_4 为一组，腿 L_3, L_6 为一组，三组依次摆动。

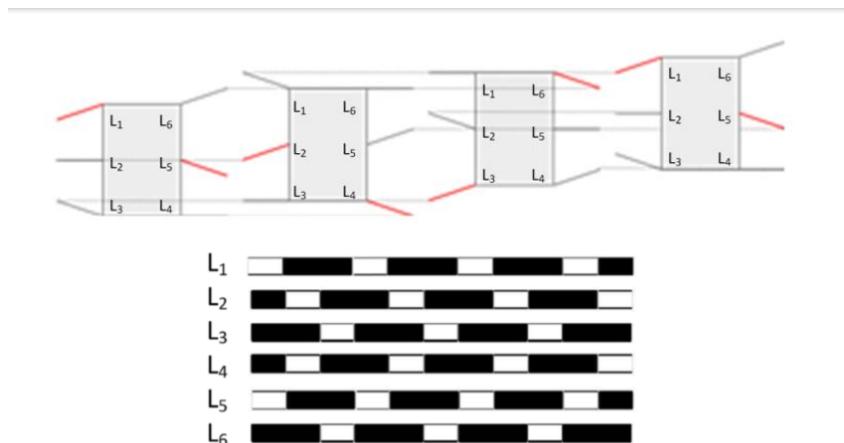


图 3-2 四步态示意图

3.1.3 六步态

六步态下，各足各自为一个分组，依次摆动，其行走如图 3-3 所示。

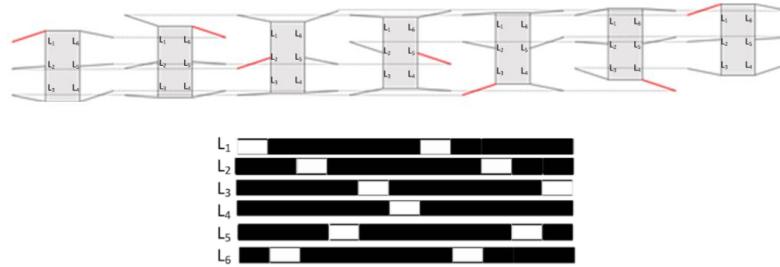


图 3-3 六步态示意图

支撑足越多，六足机器人行走时每条腿的负重越小，则机器人的行走最大负载越大，因此可以根据机器人的载重，灵活调整步态，但是支撑足越多，机器人移动速度也会越慢。

3.2 不同斜坡环境下的步态规划

当斜坡坡度较小时（图 3-4），重心稳定，机器人的落足点平稳，此时应当选用二步态。同理当斜坡坡度中等时（图 3-5），应当选用四步态。当斜坡坡度较大时（图 3-6），应当选用六步态。



图 3-4 坡度较小



图 3-5 坡度适中

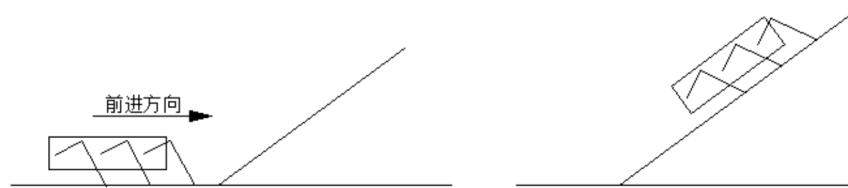


图 3-6 坡度较大

3.3 全向移动与转弯

腿式机器人灵活度较高，相对于普通的轮式机器人，可以轻易实现全向移动。

机器人移动时，可以认为机器人绕着一个圆行走。如图 3-7 所示。以机器人的底盘中心为原点，建立平面直角坐标系。

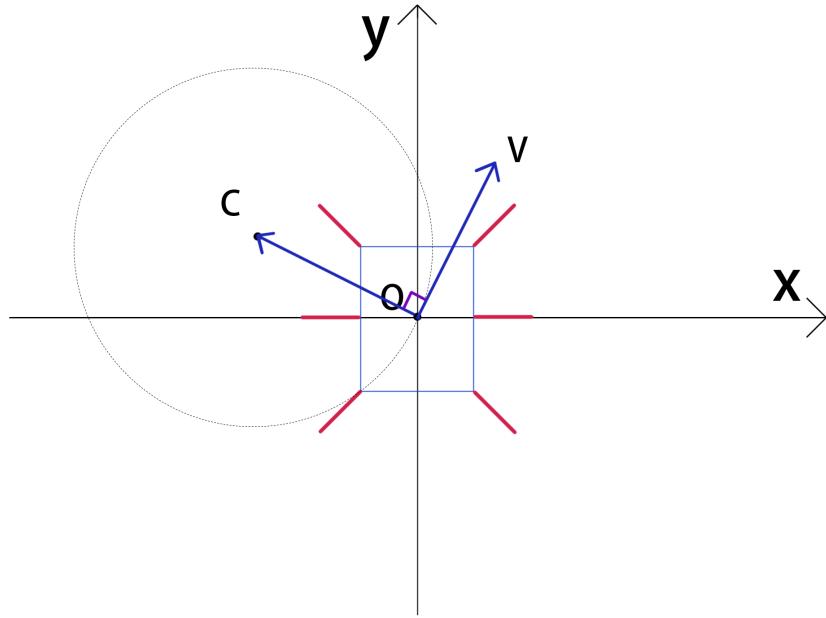


图 3-7 在平面直角坐标系做轨迹分析 1

记机器人移动时线速度为 V , 角速度为 ω , 圆心为 $C, C = \begin{bmatrix} c_x \\ c_y \end{bmatrix}, V = \begin{bmatrix} v_x \\ v_y \end{bmatrix}, v_x, v_y$ 分

别是机器人 x 轴和 y 轴方向的速度, ω 是机器人的角速度。向量 OC 垂直于向量 \vec{V} , 且规定 OC 的方向由 V 顺时针旋转 90 度获得。显然圆的半径即 $|OC|$ 与线速度 V 成正相关, 与角速度 ω 成反相关, 因此可以假设(3-1) :

$$|OC| = K_R \left(\frac{|V|}{|\omega|} \right) \quad (3-1)$$

其中, K_R 是控制圆半径的比例系数。

由 $OC \perp V$, 可得

$$OC \cdot V = 0 \quad (3-2)$$

联立(3-1) 和(3-2), 并加上约束条件 “ OC 的方向由 V 顺时针旋转 90 度获得” 可得(3-3) :

$$\begin{aligned} c_x &= \frac{v_x}{|v_x|} \sqrt{\frac{|OC|^2}{1 + \frac{(v_x)^2}{(v_y)^2}}} \\ c_y &= -\frac{c_x v_x}{v_y} \end{aligned} \quad (3-3)$$

从(3-3) 可以看出, 当 $V = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, 则 C 与原点 O 重合, 若此时 ω 不为 0, 即可完成原地旋转。当 $\omega = 0^+$, 则 $|OC| = +\infty$, 若此时 $V \neq \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, 则机器人实现直线行走。

获得圆心 C 的坐标后, 由于每条腿的位置不同, 还需要求出圆心到腿末端的向量,

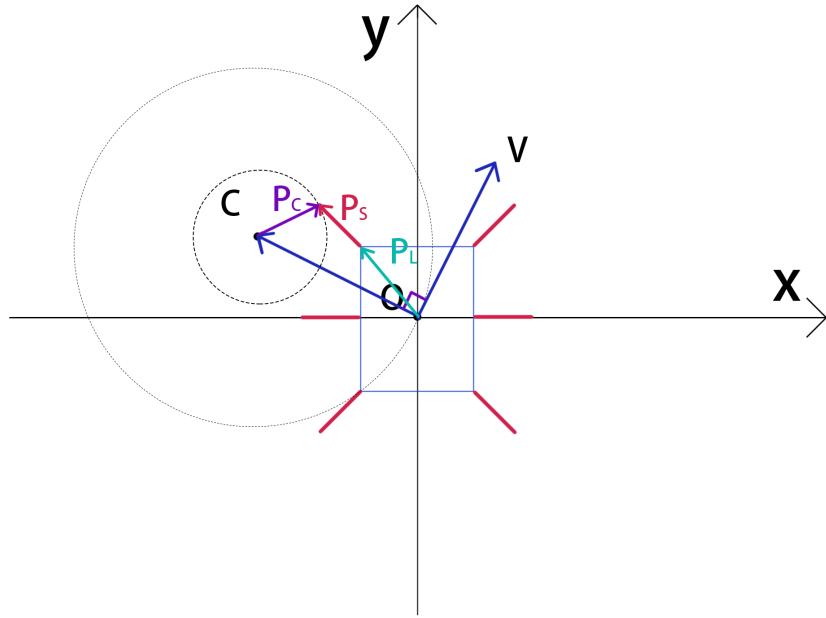


图 3-8 在平面直角坐标系做轨迹分析 2

这里以左上腿为例, 如图 3-8 所示, 设圆心到腿末端的向量为 P_C , 底盘中心到腿起始端的向量为 P_L , 正常站立时的腿起始端到末端的向量为 P_S , 则:

$$P_C = P_S + P_L - OC \quad (3-4)$$

求出 P_C 后, 即可通过对向量 P_C 的旋转进行插值, 求出机器人在行走时, 腿起始端到末端的向量 P_W 。如图 3-9 所示, 记机器人旋转角度为 θ , 旋转后圆心 C 到腿末端的向量为 P'_C , 在二维平面上的旋转矩阵为 R_Z , 易得:

$$\begin{aligned} R_Z &= \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \\ P'_C &= R_Z P_C \\ P_W &= P_L - P'_C - OC \end{aligned} \quad (3-5)$$

用同样的方法求出其他腿的 P_W 后, 进行逆运动解算, 得出机器人各关节的角度, 即可完成机器人的全向移动与转弯。该算法同样适用于 4 足机器人。

3.4 二步态 matlab 仿真

matlab 仿真使用机器人工具箱 (Robotics ToolBox), 该工具箱提供了许多机械臂仿真功能, 如运动学, 动力学和轨迹规划等。这里使用该工具箱进行六足机器人的二步态仿真, 代码见附录。

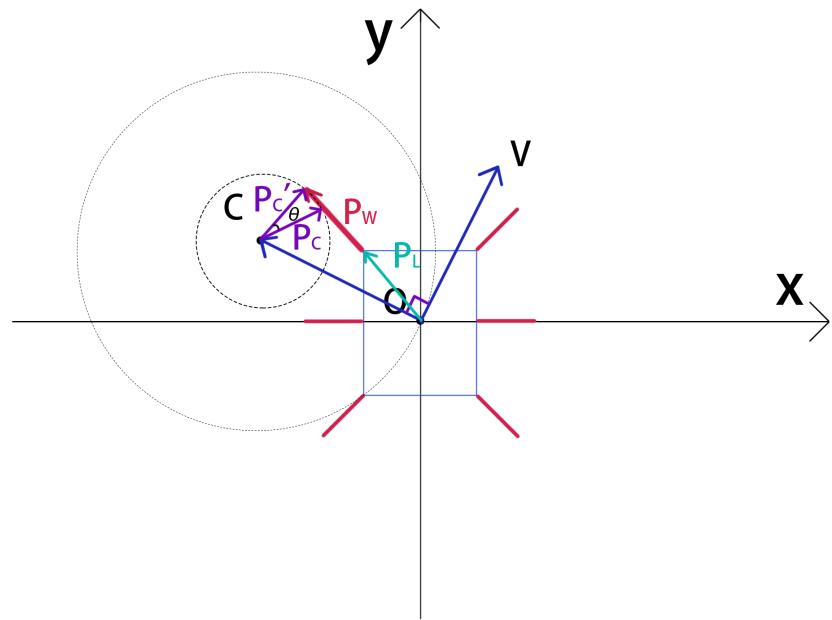


图 3-9 旋转插值

3.4.1 前进步态仿真

图 3-10 为前进步态仿真, 此时 $V = \begin{bmatrix} 0 \\ v_y \end{bmatrix}$, v_y 为任意正数, $\omega = 0$ 。

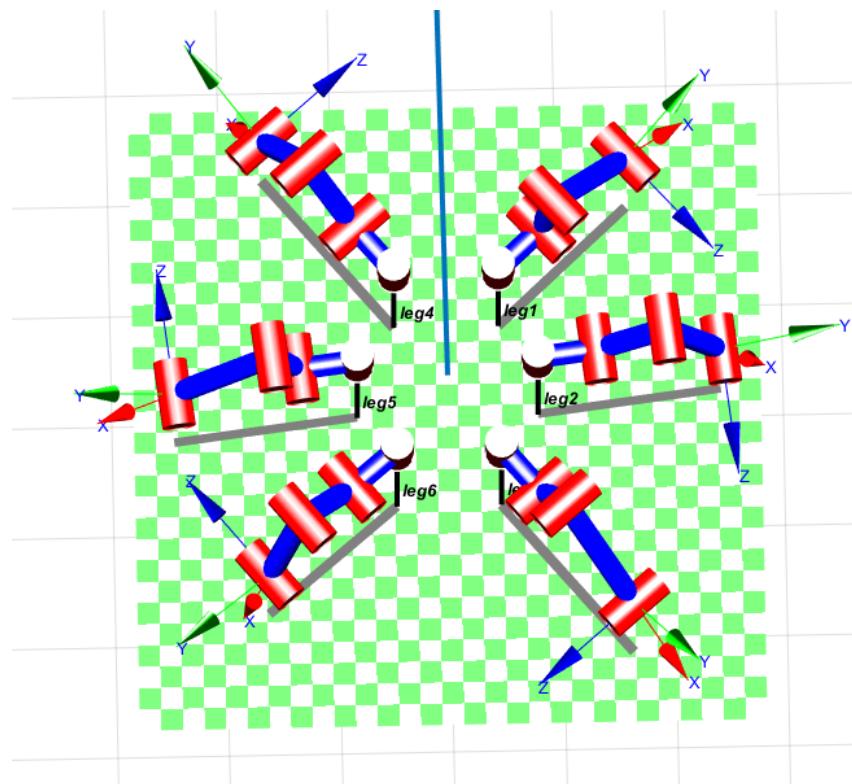


图 3-10 matlab 前进步态

3.4.2 右移步态仿真

图 3-11 为右移步态仿真, 此时 $V = \begin{bmatrix} v_x \\ 0 \end{bmatrix}$, v_x 为任意正数, $\omega = 0$ 。

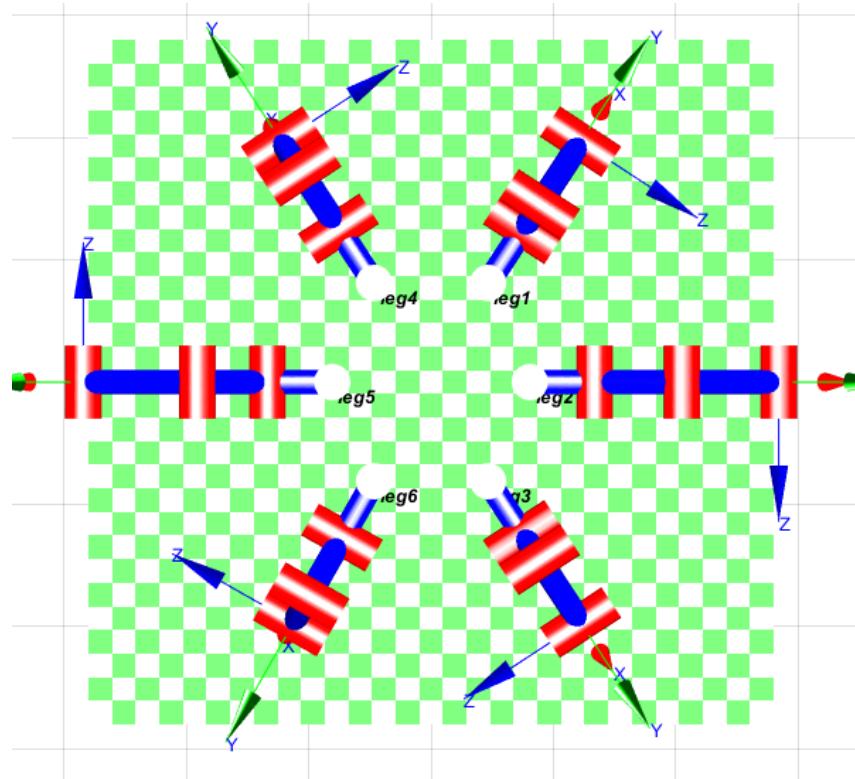


图 3-11 matlab 右移步态

第四章 硬件设计

4.1 主控选择

机器人运动解算对浮点运算要求比较高, 如果需要运动更加平滑, 则需要更多的算力, 因此该六足机器人使用了支持 FPU 且性能较高的 STM32H750VBT6。

4.1.1 stm32H750VBT6 简介

4.1.1.1 内核

stm32H750VBT6 使用 32 位 ARM Cortex-M7 内核, 携带双精度浮点 FPU, 16KB 的数据缓存和 16KB 的指令缓存, 主频高达 480MHz, 在超频的情况下可达 600MHz。

4.1.1.2 存储

128KB 的 Flash 和 1MB 的 RAM, 可通过 QSPI 拓展存储, 且时钟频率最高为 133MHz。

4.2 核心板选择

核心板使用 WeActStudio 设计的 stm32H750VBT6 迷你核心板, 该核心板内置了 8MB 的 Quad SPI Flash 和 8MB 的 SPI Flash, 提供了足够的空间用于存储机器人控制代码。该核心板提供了 SD 卡和 Type-C 接口, 具有较强的拓展能力。同时串有二极管, 防止电源回流。工作电压为 3.3V 到 5V。

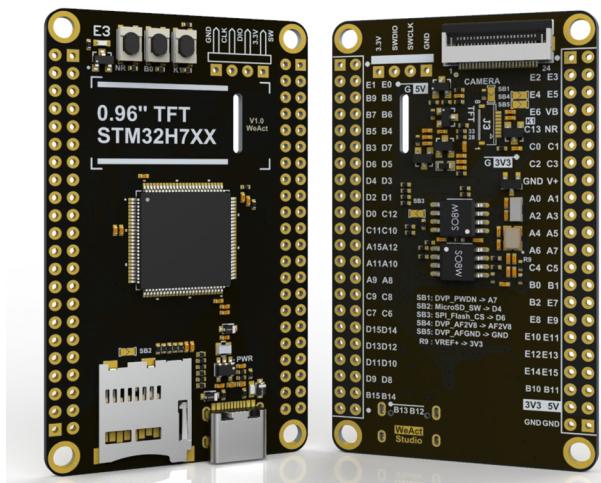


图 4-1 核心板

由于 stm32H750VBT6 的片内 Flash 较少, 仅为 128kb, 装载 FreeRTOS 后已经没剩下多少空间了, 因此通常需要将主体代码烧写至片外 Flash, 并将 BootLoader 程序烧写至片内 Flash, 通过 BootLoader 将程序跳转至存储在外部 Flash 的主体代码。

4.3 舵机选型

为了简化接线并降低故障率, 这里使用幻尔的 LX-224 串口总线舵机。舵机转动精度 0.24° , 扭矩为 $20\text{kg}\cdot\text{cm}$ (约 $1.98\text{N}\cdot\text{m}$)。



图 4-2 LX-224 串口总线舵机

舵机对 UART 一步串行接口进行时序控制, 实现半双工异步串口通讯, 通讯波特率为 115200bps , 与 stm32 进行通讯前, 需要如图 4-3 进行以下处理。

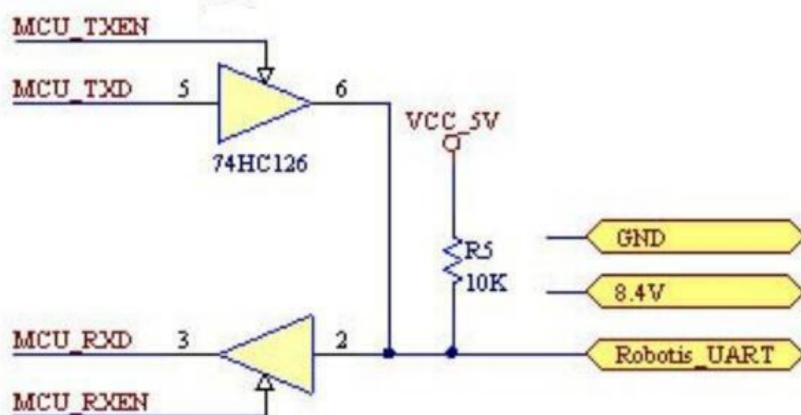


图 4-3 舵机处理

4.4 转压电路

舵机 LX-224 供电为 7.4V , 而主控 stm32H750VBT6 使用 3.3V 供电, 因此机器人使用 7.4V 供电, 并通过 LDO 线性稳压电路进行降压处理。LDO 内部相当于一个滑动变阻器, 当芯片输出电压高于设定值, 则会增加电阻值, 反之降低电阻值, 这导致 LDO 的发热较为严重, 因此需要在大的 GND 焊盘上铺足够的过孔, 增加其散热能力。

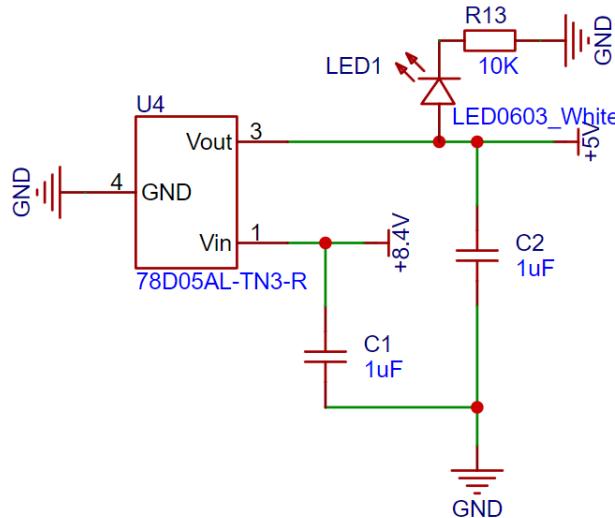


图 4-4 转压电路

4.5 供电选择

供电方面,由于六足机器人较重,行走所需功率较高,因此这里选用放电能力较强的大功率航模电池。



图 4-5 电池

4.6 遥控接收

这里选择使用大疆的DR16遥控接收机,DR16的遥控距离较远,且抗干扰能力较强。由于其使用D-BUS电平,与串口电平相反,因此需要在配置串口的时候对逻辑电平进行取反。DR16基于2.4G无线传输,具有省点,更远的传输距离,更大的带宽等特点,可以实现六足机器人的远距离控制,满足了足式机器人在复杂环境下工作的特点。工作电压为5V



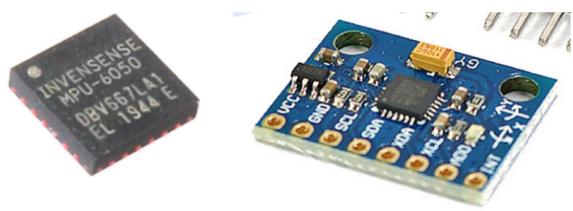
图 4-6 DR16 遥控接收机

表 4-1 DR16 的 DBUS 通信参数

DBUS 参数	数值
波特率	100Kbps
单位数据长度	8bit
奇偶校验位	偶校验
结束位	1
流控	无

4.7 陀螺仪

陀螺仪使用 MPU6050 模块，MPU6050 模块是 InvenSense 公司推出的 6 轴运动处理器组件，内部整合用有 3 轴陀螺仪和 3 轴加速度传感器，通过 IIC 接口与外部通信，且自带数字运动处理器 DMP（Digital Motion Processor）。通过 DMP，可以大大降低单片机的运算负担，该模块被广泛应用于航模等产品。工作电压为 3.3V 到 5V。



<https://blog.csdn.net/ljn5103151>

图 4-7 陀螺仪

4.8 原理图

综上所述，根据需求绘制原理图 4-8。绘制原理图使用了国产的嘉立创 EDA。嘉立创 EDA 依托于立创商城么，拥有齐全的封装库，对于 pcb 设计师较为友好。

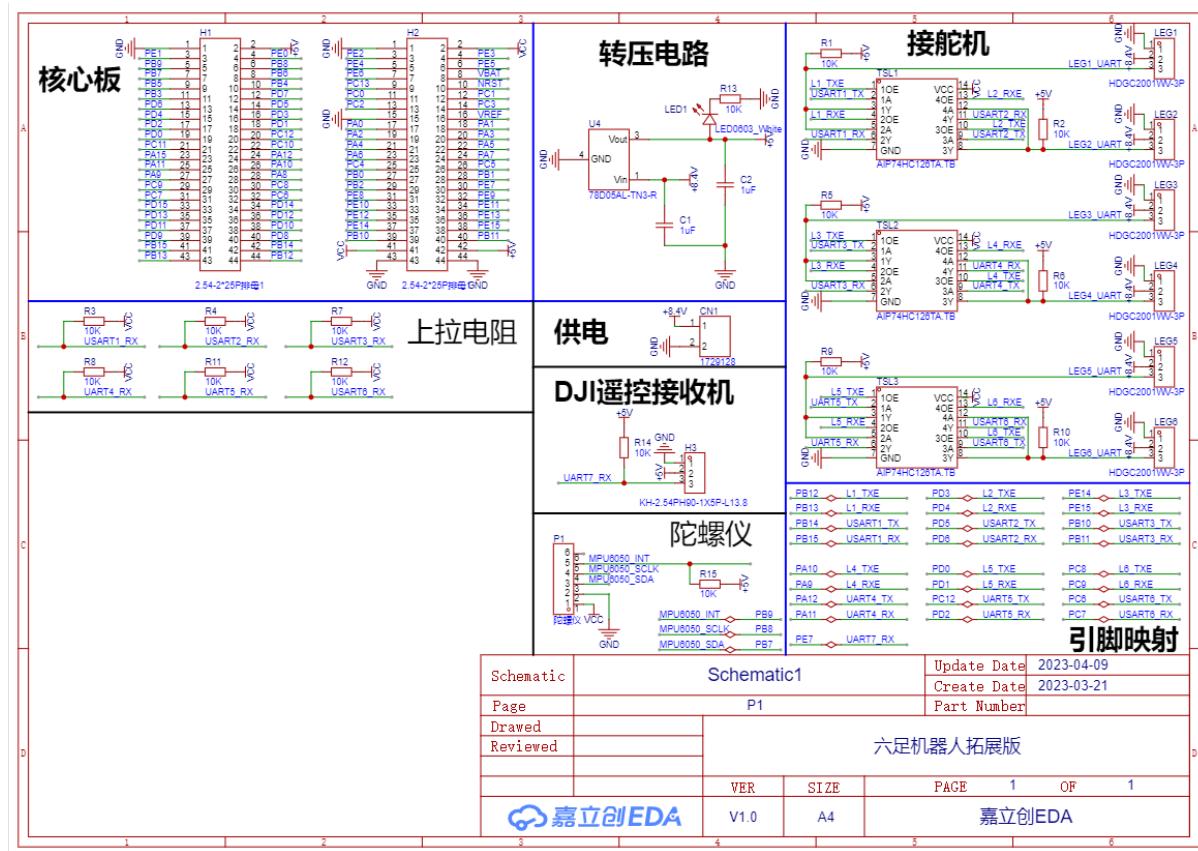


图 4-8 原理图

4.9 PCB

绘制原理图后即可绘制 PCB 图 4-9 和图 4-10，由于舵机的电流较大，因此 7.4V 供电线画的更宽。这里的铺铜接地，使其成为良好的屏蔽层，以抵消部分电磁干扰。铺铜后打满过孔，降低底层 GND 与顶层 GND 的阻抗，能滤除一些杂波，稳定传输性能。为了方便 PCB 与机械结构固定，PCB 四角打上 3mm 的过孔，这些过孔不与网络连接，可以通过 M3 尼龙柱和 M3 螺丝进行固定。LDO 线性稳压器的输入输出电阻要改尽可能靠近 LDO 芯片。为了标识转压电路是否正常工作，在转压芯片旁放置一个贴片发光二极管，用于充当电源灯。

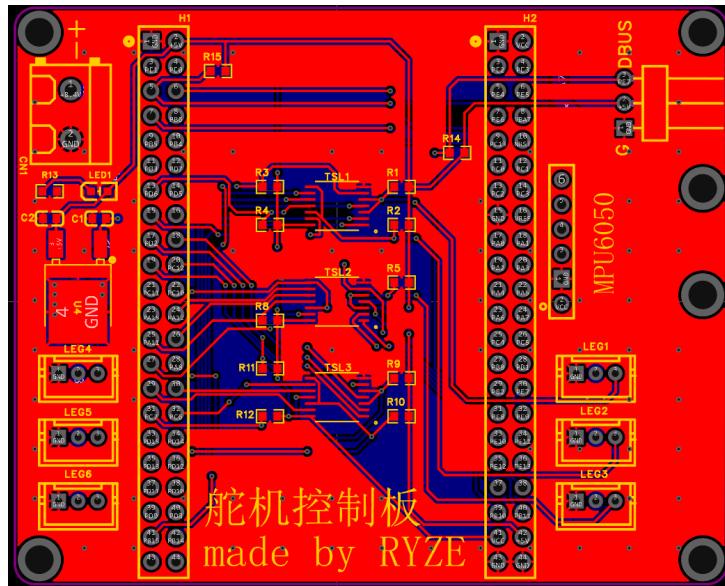


图 4-9 顶层 PCB

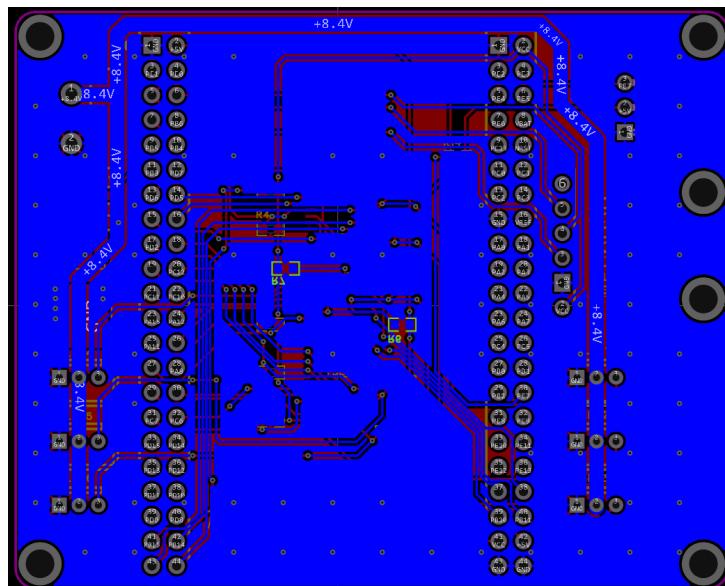


图 4-10 底层 PCB

4.10 PCB 实物

通过 PCB 打样后即可进行焊接，焊接后成品如图 4-11 所示，pcb 通过排母与陀螺仪模块和核心板进行连接。为了保证供电稳定，供电端使用绿色端子头，接入红色管状端子进行连接，并使用螺丝拧紧这样供电线不易松动，若供电线松动，可能造成短路，引发安全问题。

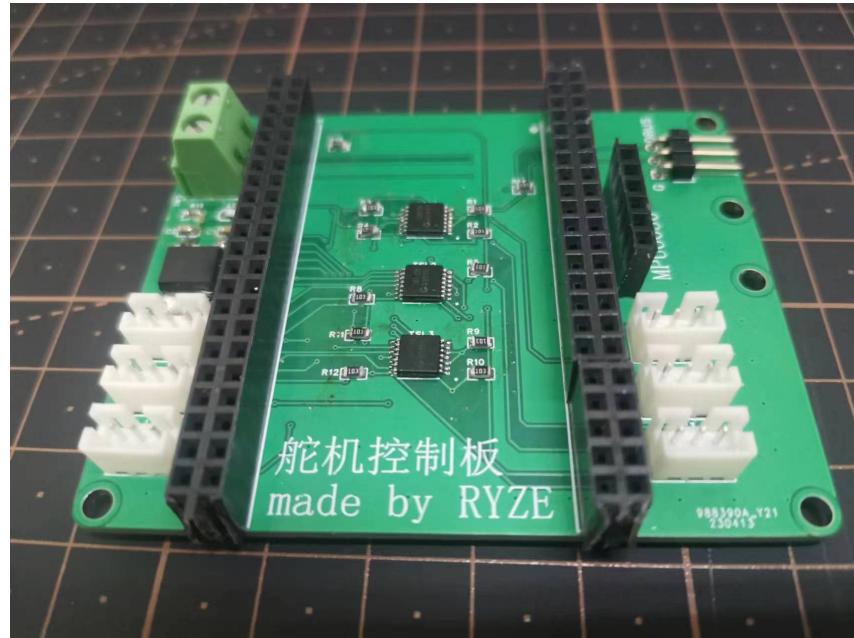


图 4-11 PCB 实物

4.11 整机实物图

制作完成后整机如图 4-12。机械结构主要使用 3d 打印进行制作，打印所使用耗材为 LEDO 6060 SLA 光敏树脂。为了降低整机的故障率，布线，舵机和主控板都使用 3d 打印件包裹在里面。如图所示，几乎没有外露走线。

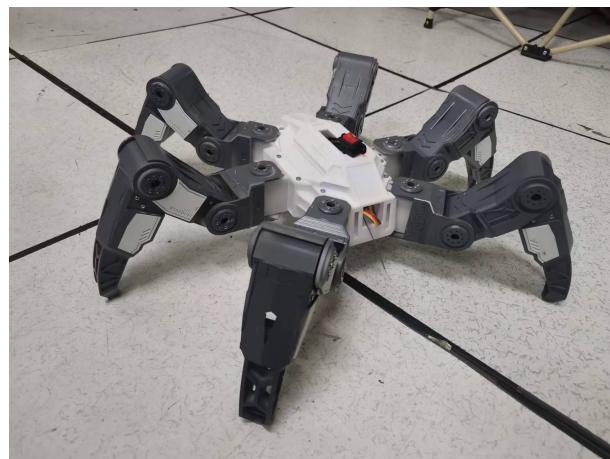


图 4-12 整机实物

第五章 软件设计

5.1 BootLoader

stm32H750VBT6 的片内 Flash 容量仅为 128kb，这对于六足机器人来说是不够的，因此需要将六足机器人主体代码烧写至外部 Flash，并在片内 Flash 中烧写 BootLoader 程序。复位后，程序在片内 Flash 中运行，执行 BootLoader，由 BootLoader 将程序跳转至片外 Flash 中运行。

Bootloader 是嵌入式系统在上电后加载的第一段代码，完成 CPU 和硬件初始化后，再跳转至嵌入式应用程序。BootLoader 执行流程如图 5-1 所示。

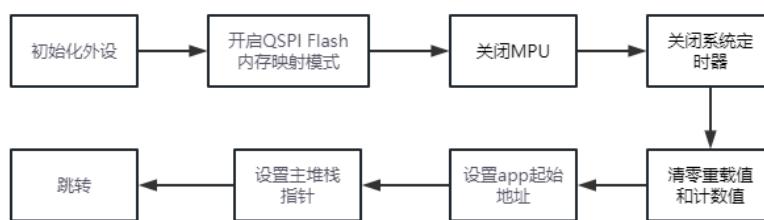


图 5-1 BootLoader 配置流程

跳转到 QSPI Flash 后，由于六足机器人的应用程序中断服务函数存储在 QSPI Flash，因此需要设置内核的中断向量表地址，否则将无法进入中断服务函数，并导致硬件异常。

5.2 嵌入式操作系统

本文所研究的机器人控制系统为嵌入式软件系统，嵌入式软件按照系统结构可分类为有操作系统的嵌入式软件和无操作系统的嵌入式软件（即裸机）。为了满足做组六足机器人实现多任务功能，操作系统使用 FreeRTOS。FreeRTOS 是一个嵌入式实时操作系统，占用资源非常小，仅占用 5KB 到 10KB 的 ROM，占用 2KB 左右的 RAM。可以运行在微控制器中。FreeRTOS 仅为内核提供了实时的调度功能，任务通信，时序和同步原语。

5.3 代码结构

代码结构如图 5-2 所示。代码以 main 函数为入口，在 main 函数中创建 Task 任务，其中包括流水灯任务 LED_Task，陀螺仪任务 IMU_Task 和六足机器人控制任务 Hexapod_Task。流水灯任务用于控制板载 LED，方便开发者观察程序是否正常运行，若程序卡死，则流水灯停止闪烁。陀螺仪任务用于向 IMU 发送请求，并获取陀螺仪姿态数据并进行解算。六足机器人控制任务则进行运动学解算，对各个舵机进行角度控制。APP 被 Task 所调用，APP 包括舵机驱动，腿部控制，用于向量运算的数学库等。

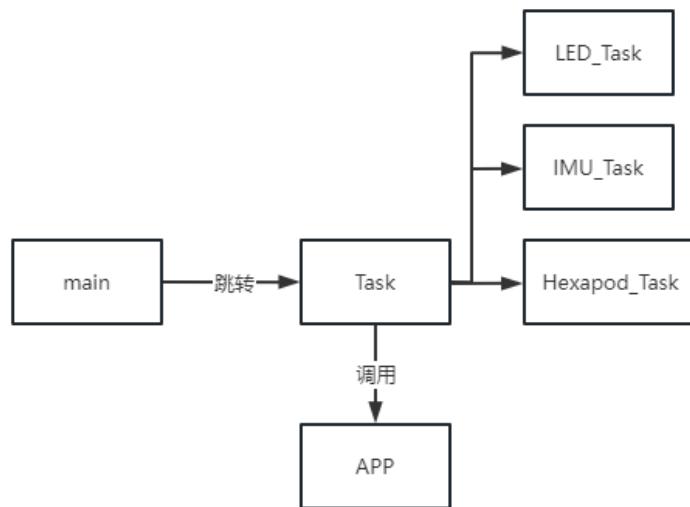


图 5-2 代码结构

5.4 软件系统框图

软件系统框图如图 5-3 所示。机器人接受到遥控器发来的数据以及传感器发来的姿态数据后，进行运动控制，运动控制包括运动学控制，即正逆运动学解算以及六足机器人的步态规划。计算完成后根据计算结果控制各个舵机的角度。

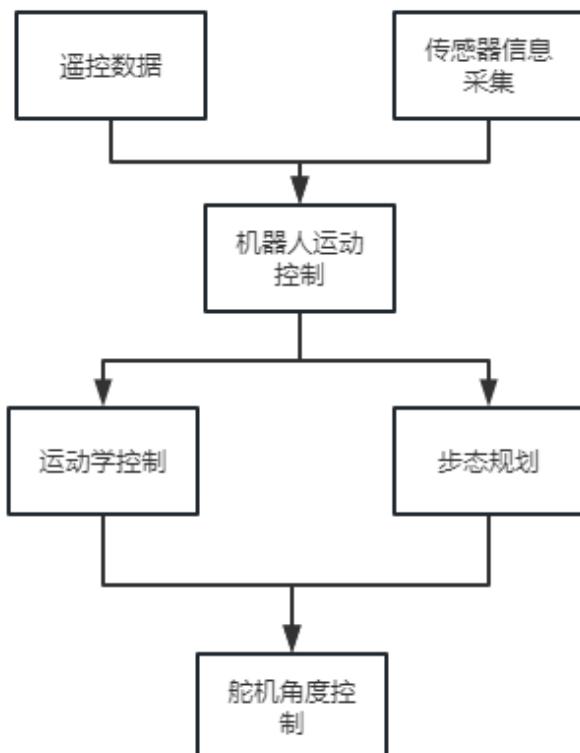


图 5-3 软件系统框图

第六章 六足机器人实验

对机器人完成整机组装，硬件设计后，将嵌入式代码烧录至核心板，通过实验来验证该六足机器人的运动性能。实验包括机身位置控制实验，机身姿态控制实验和姿态平衡实验。

6.1 机身位置控制实验

本实验分别控制机器人机身的 X 轴，Y 轴和 Z 轴位置，以此证明算法的可行性。

6.1.1 控制机身位置 Z 轴

控制机身位置 Z 轴结果如图 6-1 和图 6-2 所示。



图 6-1 位置 Z 轴降低



图 6-2 位置 Z 轴增加

6.1.2 控制机身位置 Y 轴

控制机身位置 Y 轴结果如图 6-3 所示, 此时机身偏前。



图 6-3 位置 Y 轴控制

6.1.3 控制机身位置 X 轴

控制机身位置 X 轴结果如图 6-4 所示, 此时机身偏左。



图 6-4 位置 X 轴控制

6.2 机身姿态控制实验

本实验中, 分别控制机身姿态 Z, Y, X 轴, 这里 Z 轴即 yaw 轴, Y 轴即 roll 轴, X 轴即 pitch 轴。

6.2.1 控制机身姿态 Z 轴

控制机身姿态 Z 轴, 结果如图 6-5 所示。



图 6-5 姿态 Z 轴控制

6.2.2 控制机身姿态 Y 轴

控制机身姿态 Y 轴, 结果如图 6-6 所示。



图 6-6 姿态 Y 轴控制

6.2.3 控制机身姿态 X 轴

控制机身姿态 X 轴, 结果如图 6-7 所示。



图 6-7 姿态 X 轴控制

6.3 姿态平衡实验

姿态平衡指的是, 六足机器人在不同坡度的斜坡中, 自主调整机身角度, 使机身角度维持在水平状态下, 如图 6-8 所示。

实验如图 6-9 和图 6-10 所示, 将一杯水放到机器人上面, 若杯中的水位置不变, 则实验成功。很明显, 倾斜桌面后, 机器人改变了自身姿态, 杯中的水位置基本不变。

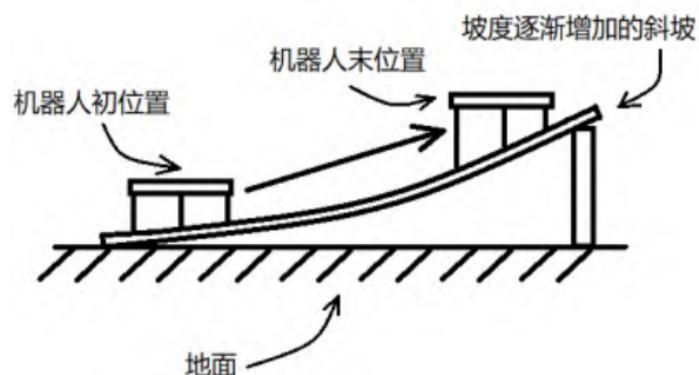


图 6-8 姿态平衡示意图

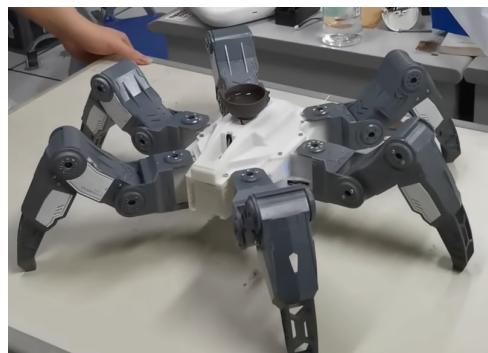


图 6-9 桌面水平



图 6-10 桌面倾斜

第七章 总结与展望

7.1 总结

随着人们对于复杂环境下具备高移动能力，高稳定性，高灵活性的移动平台的需求增加，腿式机器人作为一种拥有全向移动能力的运载平台，在复杂环境如洞穴、废墟、山地以及城市巷战中具有广阔的应用前景。相比于传统的履带式机器人或轮式机器人，六足机器人灵活性较强，能适应各种复杂多变的地形，因此成为当前机器人研究的热点。本文以六足机器人为研究对象，结合六足机器人的运动特性，对机器人进行步态规划以及运动学建模，进而实现六足机器人的全向移动与转弯算法设计。再通过 matlab 进行相关的仿真实验，并通过硬件设计、嵌入式软件设计以及机器人整机组装完成机器人实物搭建，并通过实物检验运动学建模以及步态分析算法的正确性。

在本文的研究过程之中，主要完成了以下工作。

1. 对六足机器人做了简要介绍，并通过 D-H 参数建立六足机器人的运动学模型。对腿部进行正逆运动学分析，理论上验证了机身姿态和位置的控制，为后续步态控制提供基础。与市面上大多使用动作组来控制六足机器人不同，本文采用自主设计的实时解算的方法进行六足机器人控制，便于实现全向移动、步幅可控、机身姿态可调等功能，这大大提高了其地形适应能力。
2. 根据六足机器人的特点，在理论上验证了六足机器人全向行走以及全向行走的同时转弯的可行性。并使用 matlab 完成了仿真。
3. 根据六足机器人的特点，对硬件进行选型，并通过 PCB 进行连接，实现六足机器人的硬件设计。
4. 对机器人实物进行了机身控制实验，实验验证了机器人机身控制算法的正确性。

7.2 展望

当然，在不断的实验中也发现本文实现的机器人存在一些不足的地方，后期对于六足机器人的研究可以从以下几个方面进行优化。

1. 实际环境的路面情况往往较为复杂，需要机器人根据地面情况实时调整步态，这需要机器人具有较强的感知能力。
2. 在实际使用的过程之中，六足机器人的某个腿可能会出现故障，此时需要灵活调整步态以完成行走、转弯等功能。

参考文献

- [1] 黄麟, 韩宝玲, 罗庆生, 等. 仿生六足机器人步态规划策略实验研究 [J]. 华中科技大学学报: 自然科学版, 2007(12):4.
- [2] 韩建海, 赵书尚, 李济顺. 六足机器人行走步态的协调控制 [J]. 机电工程, 2004, 21(4):3.
- [3] 李满宏, 张建华, 张明路. 新型仿生六足机器人自由步态中足端轨迹规划 [J]. 中国机械工程, 2014, 25(6):5.
- [4] 庄红超, 高海波, 邓宗全, 等. 电驱动重载六足机器人关节转速分析方法 [J]. 机械工程学报, 2013, 49(23):9.
- [5] 梁美彦, 薛太林, 王昱, 等. 基于 Android 控制的智能六足机器人动力学建模及实现 [J]. 测试技术学报, 2017, 31(6):7.
- [6] 桂博文. 复杂环境下的六足机器人运动控制研究 [D]. 上海交通大学, 2016.
- [7] 阳方平, 李洪谊, 王越超, 等. 一种求解冗余机械臂逆运动学的优化方法 [J]. 机器人, 2012, 34(1):6.
- [8] 徐扣. 六自由度机械臂的逆运动学求解与轨迹规划研究 [D]. 广东工业大学, 2016.
- [9] 邓学磊, 杨灿军, 毕千, 等. 舵机驱动的足式机器人腿部运动性能优化策略 [J]. 浙江大学学报 (工学版), 2015, 49(4):638-643.
- [10] 张金柱, 金振林, 陈广广. 六足步行机器人腿部机构运动学分析 [J]. 农业工程学报, 2016, 32(9):8.

致 谢

在本论文完成之际，我首先要感谢我的指导老师谢仁平，他不仅在学术上给予了我无私的指导和帮助，还在生活上给予了我关心和鼓励。他严谨的治学态度、深厚的专业知识、广博的学术视野和热情的教学风格，对我产生了深远的影响，让我受益终生。

其次，我要感谢我的学院和实验室，为我提供了良好的学习环境和设施，让我能够顺利地完成我的研究工作。特别感谢实验室的所有老师和同学，他们在我遇到困难时给予了我及时的帮助和建议，让我感受到了浓浓的温暖和友谊。

再次，我要感谢我的师兄和舍友们，他们在我学习和生活中给予了我很多的支持和陪伴。他们不仅与我分享了他们的经验和见解，还与我一起度过了许多难忘的时光。他们是我最宝贵的朋友，也是我最可靠的伙伴。

最后，我要感谢我的家人，他们是我最坚强的后盾，一直默默地支持着我、鼓励着我、理解着我。没有他们的爱和牺牲，就没有我的今天。在此，我向他们致以最诚挚的感谢和最深切的祝福！

附录

MATLAB 步态仿真代码:

代码附-1 步态仿真代码

```

1 Len1 = 53;Len2=80;Len3=144;
2 chassis_len = 162.2;chassis_wide = 161.5;front_wide=93.3;
3
4 syms xita1 xita2 xita3 L1 L2 L3;
5 L1 = 53,L2=80,L3=144;
6 T0_1=traOp(0,0,0,xita1)
7 T1_2=traOp(pi/2, L1, 0, xita2)
8 T2_3=traOp(0, L2, 0, xita3)
9 T3_4=traOp(0, L3, 0, 0)
10 T0_4=simplify((T0_1*T1_2*T2_3*T3_4))
11 P0_4=[T0_4(1,4),T0_4(2,4),T0_4(3,4)];
12
13 %腿1
14 % theta d a alpha
15 joint(1) = Link([0,0,0,0],'modified');
16 joint(2) = Link([45,0,Len1,pi/2],'modified');
17 joint(3) = Link([-120,0,Len2,0],'modified');
18 joint(4) = Link([0,0,Len3,0],'modified');
19 Leg1 = SerialLink(joint);
20 Leg1.name = 'leg1';
21 Leg1.base = [front_wide/2,chassis_len/2,0];
22 %腿2
23 joint(1) = Link([0,0,0,0],'modified');
24 joint(2) = Link([45,0,Len1,pi/2],'modified');
25 joint(3) = Link([-120,0,Len2,0],'modified');
26 joint(4) = Link([0,0,Len3,0],'modified');
27 Leg2 = SerialLink(joint);
28 Leg2.name = 'leg2';
29 Leg2.base = [chassis_wide/2,0,0];
30 %腿3
31 joint(1) = Link([0,0,0,0],'modified');
32 joint(2) = Link([45,0,Len1,pi/2],'modified');
33 joint(3) = Link([-120,0,Len2,0],'modified');
34 joint(4) = Link([0,0,Len3,0],'modified');
35 Leg3 = SerialLink(joint);
36 Leg3.name = 'leg3';
37 Leg3.base = [front_wide/2,-chassis_len/2,0];
38 %腿4

```

```

39 joint(1) = Link([0,0,0,0], 'modified');
40 joint(2) = Link([45,0,Len1,pi/2], 'modified');
41 joint(3) = Link([-120,0,Len2,0], 'modified');
42 joint(4) = Link([0,0,Len3,0], 'modified');
43 Leg4 = SerialLink(joint);
44 Leg4.name = 'leg4';
45 Leg4.base = [-front_wide/2,chassis_len/2,0];
46 %腿5
47 joint(1) = Link([0,0,0,0], 'modified');
48 joint(2) = Link([45,0,Len1,pi/2], 'modified');
49 joint(3) = Link([-120,0,Len2,0], 'modified');
50 joint(4) = Link([0,0,Len3,0], 'modified');
51 Leg5 = SerialLink(joint);
52 Leg5.name = 'leg5';
53 Leg5.base = [-chassis_wide/2,0,0];
54 %腿6
55 joint(1) = Link([0,0,0,0], 'modified');
56 joint(2) = Link([45,0,Len1,pi/2], 'modified');
57 joint(3) = Link([-120,0,Len2,0], 'modified');
58 joint(4) = Link([0,0,Len3,0], 'modified');
59 Leg6 = SerialLink(joint);
60 Leg6.name = 'leg6';
61 Leg6.base = [-front_wide/2,-chassis_len/2,0];
62
63 Vx = 1000; Vy=0; omega=0;
64 k_cen = 1; %用于确定圆心模长的系数
65 %数据预处理, 避免出现0
66 if(Vx==0), Vx= Vx+0.0001; end
67 if(Vy==0), Vy = Vy+0.0001; end
68 if(omega==0), omega = omega+0.0001; end
69 module_CEN = k_cen/omega*sqrt(Vx^2+Vy^2)           %计算圆心的模长
70
71 if(omega<0)
72     Vx = -Vx;
73     Vy = -Vy;
74 end
75
76 %旋转90度
77 Vs_x = -Vy;
78 Vs_y = Vx;
79
80 if Vs_x>=0
81     CENx = sqrt(module_CEN^2/(1+Vx^2/Vy^2));

```

```

82 else
83     CENx = -sqrt(module_CEN^2/(1+Vx^2/Vy^2));
84 end
85
86
87 kr_1 = 1;kr_2=1; %用于计算步伐大小的系数
88 R = kr_1*abs(omega)+kr_2*sqrt(Vx^2+Vy^2)
89 if(R>50),R=50;end%限制步伐大小
90
91 CENy = -CENx*Vx/Vy;
92 CEN = [CENx,CENy] '
93
94 n_point=20; %点的数量
95 theta_stands = [40/180*pi,-110/180*pi]; %机械腿站立状态下最后两个关节的角度
96 action = struct('Leg',zeros(3,n_point));
97 actions = repmat(action,[1,6]); %初始化动作组
98 Pws = zeros(3,6);
99 P_legs = zeros(2,6);
100
101 % 腿的排序
102 %   4   1
103 %   5   2
104 %   6   3
105
106 %各个机械腿相对于起始端的末端坐标
107 Pws(:,1) = single(subs(P0_4,[xita1,xita2,xita3],[pi/4,theta_stands]));
    %机械腿站立时的坐标
108 Pws(:,2) = single(subs(P0_4,[xita1,xita2,xita3],[0,theta_stands]));
109 Pws(:,3) = single(subs(P0_4,[xita1,xita2,xita3],[-pi/4,theta_stands]));
110 Pws(:,4) = single(subs(P0_4,[xita1,xita2,xita3],[3*pi/4,theta_stands]));
111 Pws(:,5) = single(subs(P0_4,[xita1,xita2,xita3],[pi,theta_stands]));
112 Pws(:,6) = single(subs(P0_4,[xita1,xita2,xita3],[5*pi/4,theta_stands]));
113 %各个机械腿起始端相对于机器人中心的坐标
114 P_legs(:,1) = [front_wide/2,chassis_len/2]';
115 P_legs(:,2) = [chassis_wide/2,0]';
116 P_legs(:,3) = [front_wide/2,-chassis_len/2]';
117 P_legs(:,4) = [-front_wide/2,chassis_len/2]';
118 P_legs(:,5) = [-chassis_wide/2,0]';
119 P_legs(:,6) = [-front_wide/2,-chassis_len/2]';
120
121 %计算圆心到每个腿部末端的向量
122
123 Vec_R_legs = Pws(1:2,:) + P_legs - CEN;

```

```

124 %计算圆心与机械腿末端的夹角
125 thetas = atan2(Vec_R_legs(2,:),Vec_R_legs(1,:));
126 %计算圆心到机械腿末端的模长
127 norm_P_legs = zeros(1,6);
128 for i=1:6
129     norm_P_legs(i) = norm(Vec_R_legs(:,i)');
130 end
131 %计算各个机械腿步态规划的大小比例
132 r_ratios = norm_P_legs/max(norm_P_legs);
133 %计算各个机械腿步态规划的半径
134 Rs = norm_P_legs/max(norm_P_legs)*R;
135 %计算机械腿走一步绕圆心拐的角度, 随便拿一组数据来算就行
136 d_theta = 2*Rs(1)/norm_P_legs(1);
137 step_size = d_theta/(n_point/2);
138
139 points_z = zeros(1,n_point); %先设置每个点的z轴坐标
140 y_temp = -R:4*R/n_point:R; %用于设置z轴坐标的临时数组
141 points_z(n_point/2+2:n_point) = sqrt(R^2 - y_temp(2:n_point/2).^2);
    %根据y坐标计算z坐标
142
143 %开始步态规划
144 for i=1:6
145     %计算圆心相对于机械腿起始端的坐标
146     Center_circle = CEN - P_legs(:,i);
147     %根据圆的大小缩小z轴高度, 并迁移坐标系到机械腿末端, 因为站立时z轴都是一样的,
148     %所以随便用一个Pw就行
149     points_z_temp = points_z*r_ratios(i) + Pws(3,1);
150
151     %先画初始点
152     angle_t = thetas(i)+d_theta/2; %计算这个点的角度
153     point_x = Center_circle(1) + norm_P_legs(i)*cos(angle_t);
        %计算这个点的x轴坐标(相对于机械腿起始端)
154     point_y = Center_circle(2) + norm_P_legs(i)*sin(angle_t);
        %计算这个点的y轴坐标(相对于机械腿起始端)
155     point_z = points_z_temp(1);
156     [actions(i).Leg(1,1),actions(i).Leg(2,1),actions(i).Leg(3,1)] =
         my_ikine(point_x,point_y,point_z);%画下半圆
157     %画中间点
158     for j=2:n_point/2
159         angle_t = thetas(i)+d_theta/2 - step_size*(j-1); %计算这个点的角度
160         point_x = Center_circle(1) + norm_P_legs(i)*cos(angle_t);
            %计算这个点的x轴坐标(相对于机械腿起始端)
161         point_y = Center_circle(2) + norm_P_legs(i)*sin(angle_t);

```

```

    %计算这个点的y轴坐标(相对于机械腿起始端)
162 point_z = points_z_temp(j);
163 [actions(i).Leg(1,j),actions(i).Leg(2,j),actions(i).Leg(3,j)] =
    my_ikine(point_x,point_y,point_z);%画下半圆
164 %因为中间点的x和y有些重合, 所以顺便画上半圆
165 point_z = points_z_temp(end-(j-2));
166 [actions(i).Leg(1,n_point-(j-2)),actions(i).Leg(2,n_point-(j-2)),...
167 actions(i).Leg(3,n_point-(j-2))] = my_ikine(point_x,point_y,point_z);%画上半圆
168 end
169 %画终点
170 angle_t = thetas(i)-d_theta/2; %计算这个点的角度
171 point_x = Center_circle(1) + norm_P_legs(i)*cos(angle_t);
    %计算这个点的x轴坐标(相对于机械腿起始端)
172 point_y = Center_circle(2) + norm_P_legs(i)*sin(angle_t);
    %计算这个点的y轴坐标(相对于机械腿起始端)
173 point_z = points_z_temp(n_point/2+1);
174 [actions(i).Leg(1,n_point/2+1),actions(i).Leg(2,n_point/2+1),...
175 actions(i).Leg(3,n_point/2+1)] = my_ikine(point_x,point_y,point_z);%画下半圆
176 end
177
178 %这三条腿是另一组, 执行动作组的时间不同
179 for i=1:n_point/2
180     actions(4).Leg(:,[i;i+n_point/2]) = actions(4).Leg(:,[i+n_point/2;i]);
181     actions(2).Leg(:,[i;i+n_point/2]) = actions(2).Leg(:,[i+n_point/2;i]);
182     actions(6).Leg(:,[i;i+n_point/2]) = actions(6).Leg(:,[i+n_point/2;i]);
183 end
184
185 hold on;
186 xita_ = 0:0.02:2*pi;
187 module_CEN = norm(CEN);
188 x = CEN(1) + module_CEN*cos(xita_);
189 y = CEN(2) + module_CEN*sin(xita_);
190 z = repmat(Pws(3,1),[1,length(xita_)]);
191 Leg1.plot([actions(1).Leg(:,1)',0]);
192 Leg2.plot([actions(2).Leg(:,1)',0]);
193 Leg3.plot([actions(3).Leg(:,1)',0]);
194 %左边腿
195 Leg4.plot([actions(4).Leg(:,1)',0]);
196 Leg5.plot([actions(5).Leg(:,1)',0]);
197 Leg6.plot([actions(6).Leg(:,1)',0]);
198 plot3(x,y,z,'linewidth',3);
199 i=1;
200 while true

```

```
201 if(omega>0)
202     i=i+1;
203     if i==n_point+1
204         i=1;
205     end
206 else
207     i=i-1;
208     if i==0
209         i=n_point;
210     end
211 end
212
213 %右边腿
214 Leg1.plot([actions(1).Leg(:,i)',0]);
215 Leg2.plot([actions(2).Leg(:,i)',0]);
216 Leg3.plot([actions(3).Leg(:,i)',0]);
217 %左边腿
218 Leg4.plot([actions(4).Leg(:,i)',0]);
219 Leg5.plot([actions(5).Leg(:,i)',0]);
220 Leg6.plot([actions(6).Leg(:,i)',0]);
221 end
```