湖北工業大學

毕 业 设 计（论 文）

题 目 自动四足信使机器人设计

姓　　名 余磊涛

学　　号 1510100413

所在学院 机械工程学院

专业班级 15 机自_创新 1

指导教师 许万

日　　期 2019 年 5 月 28 日

湖北工業大學

毕业设计（论文）任务书

学　　院	机械学院	指导教师	许万	职　　称	教授
学生姓名	余磊涛	专业班级	15 创新_机自 1	学　　号	1510100413
设计题目	自动四足信使机器人设计				
题目来源及工程背景	第十八届全国大学生机器人大赛 ROBOCON				
设计内容目标和要求	**一、毕业设计内容** 设计一台自动四足信使机器人，使其能够从手动机器人接过“令牌”，通过场地中的平地、障碍、坡地，到达指定区域，将令牌交予自动机器人，并在指定区域抛投“兽骨”，该机器人需全自动操作，不得人工干预。场地及道具具体要求详见第“十八届全国大学生机器人大赛 ROBOCON 比赛规则” 主要技术指标：机器人重量：不得超过 25KG； 机器人尺寸：长不得超过 1000mm、宽度不得超过 800，高度不得超过 800mm 供电电压：DC24V 具体设计内容：1．收集整理与本课题有关的相关资料； 2．根据比赛规则拟定整机的设计方案; 3．对整个机械的结构进行设计计算； 4. 设计机器人控制系统； 5．绘制机器人相关机械结构图纸； 6. 绘制机器人控制系统原理图； 7．制作样机并调试； 8. 翻译外文文献。 **二、毕业设计目标** 1．培养学生查阅文献、分析资料、编写报告的能力； 2. 进一步培养学生设计、绘图和计算机软件应用的能力； 3. 培养学生综合运用所学知识解决本专业较为复杂工程实际问题的能力。				

三、毕业设计成果及要求

1．在查阅资料的基础上，规划设计进度，构思设计方案，完成开题报告；

2．充分考虑经济、环境和可持续发展要求，通过方案对比，完成总体方案设计；

3．完成整机工程图设计，要求图纸表达准确；

4．完成整机控制系统设计，绘制原理图；

5．制作样机并调试;

6．在上述基础上完成论文，论文符合规范要求，不少于 1 万字；

7．完成与专业或课题相关的外文资料翻译，要求通顺、准确，外文不少于 1-2 万印刷符或译文不少于 5000 汉字；

四、设计进度

2018.12.20~2019.2.28 前期阶段：完成毕业设计选题，查阅有关论文、资料，了解本课题的现状及动态，拟定具体实施方案，完成开题报告；

2019.3.1~2019.3.30 机械设计阶段：完成机械结构设计；完成英文资料的翻译；

2019.4.1~2019.4.20 控制系统设计阶段：完成整机控制系统设计；

2019.4.21~2019.5.13 制作调试阶段：制作样机并调试；

2019.5.14~2019.5.25 文档整理编辑阶段：整理设计文档，撰写并完善设计说明书；

2019.5.26~2019.5.30 答辩准备阶段：答辩 PPT 准备，打印装订毕业设计说明书（或论文）、英文翻译资料、图纸等有关资料。

指导教师签名：

年　　月　　日

基层教学单位审核		学　院 审　核	

此表由指导教师填写学院审核

湖北工業大學

毕业设计（论文）学生开题报告

课题名称	自动四足信使机器人设计				
课题来源	第十八届全国大学生机器人大赛 Robocon	课题类型	AX	指导教师	许万
学生姓名	余磊涛	学 号	1510100413	专业班级	15 创新—机自

1. 研究目的及意义

随着人类活动空间的不断扩大和延伸，机器人正逐渐的应用于太空探索、极地考察、野外作业、核电站维护、抢险以及军事等领域，成为人类不可缺少的重要工具。在应用的过程中，人们要求机器人不仅要在已知的结构化环境下稳定运动，而且还要求机器人在复杂未知的非结构化环境中具备良好的适应性。足式机器人以其稳定的行走特性被广泛地应用于各种复杂环境中，相较于轮式机器人、履带式机器人，足式机器人对路面要求不高，它可以自由跨越障碍、趟过沙地、掠过沼泽等特殊路面。 足式机器人的主要类型有双足、四足、多足机器人（多足主要包括六足、八足机器人等）。在足式机器人中，双足机器人的结构简单，但运动的稳定性差，对控制算法有着较高的要求；多足机器人的运动稳定性较好，但结构比较复杂；相比较于双足和多足机器人，四足机器人的运动稳定性要优于双足机器人，机械结构又比多足机器人简单。仿生四足机器人既能以静步态实现在复杂地形上的缓慢行走，又能实现高速运动，在复杂地形环境下的军事或者民用的物资输送、野外勘测、救援抢险、教育和娱乐等许多方面有着及其广泛的应用前景。在足式机器人研究领域，从结构的简易程度和运动的稳定性两个方面进行考虑，在工程实践中，四足机器人是一种较优的类型。

2. 国内外研究现状

早在机器人学兴起之前，就有生物学家对于四足动物的运动进行了深入的研究。代表的研究有 1975 年 McMahon 使用跑步机和高速摄像机对从老鼠到马的各种动物的运动进行了观察研究，得到了动物的步频、步长、最大速度等步态参数与动物质量有幂函数关系的结论；以及 1977 年 Cavagna 等人用力台得到了四足动物行走和奔跑时得到触地力曲线等。这些研究的结果具有普遍适用性，对于随后足式机器人的研究有着指导性和参照性的意义，有力支撑了仿生足式机器人学科的发展。

对于四足机器人的研究公认始于 1968 年美国 GE 公司的工程师 Mosher 的 Walking Truck，该机器人是一个没有计算机控制、完全由工作人员对机器人各个执行器进行控制以实现其运动的四足机器人。在四足机器人的发展过程中，主要出现了如下几种主流的机器人步态控制方法。

1.基于静态稳定的机器人步态控制 在四足机器人的早期研究中，计算机及电

子、电气技术发展的限制使得机器人的控制水平较差，导致机器人的动态性能很差，只能基于静态稳定的控制策略进行慢速、静步态的运动，这种基于静力学的机器人控制方法在生成步态时必须保证机器人的重心在地面上的投影必须始终位于机器人各支撑腿形成的多边形之内，同时在运动控制时令重心尽量远离多边形的边缘以提高稳定裕度。

2.基于动力学模型行为特性的机器人步态控制 第一台基于动力学模型行为特性的机器人是美国麻省理工大学的 Raibert 等人在 1982 年设计的 MITleg 单足机器人。该机器人以弹簧-负载倒立摆模型（SLIP 模型）作为动力学模型的参照，通过对于弹簧负载倒立摆动力学模型的分析成功地找到了对于单足机器人的前进速度、竖直弹跳高度该方法以 PID 控制规律通过控制腿部落地角控制机器人前进速度，通过控制腿部着地力控制竖直弹跳高度，通过控制髋关节力矩控制俯仰角度，在机器人的实验上获得了较好的效果。这种控制方法的每一个控制量都单独对应着一个动力学上的量，故称这种方法为基于动力学特性的机器人控制方法。Raibert 等人随即将这种方法的应用拓展至双足、四足机器人上。通过将同时着地的两条腿视作一条虚拟腿，合理规划机器人的各足的着地角、足底力和髋关节力矩，麻省理工的四足机器人可以进行中速下对称步态的运动，如对角小跑步态（trot 步态）、跳跃步态（bound 步态）下的行进。

3.基于被动行走的机器人步态控制 被动行走，是指机器人能够利用自身的动力学特性，令自身的重力势能、弹性势能及动能合理地转化，从而能够在不依靠控制及主动力驱动的情况下自发行走。这一研究始于 Raibert 等人在 1987 年提出的一种带扭簧及直线弹簧的单腿运动模型，通过合理计算该模型的结构参数，可以使其在理想的无能量损耗状况下持续以单腿弹跳的步态行进。

4.基于中枢模式发生器的步态控制 基于中枢模式发生器（Central Pattern Generator, CPG）的步态控制思想来源于低级生物的运动。低等级生物的中枢模式发生器能够产生节律性的神经信号，从而控制自身的每条腿进行节律性的运动。类似地，机器人的中枢模式发生器也是一个由神经元模型组成的神经网络模型，该神经网络模型中每一个神经元控制一个关节的运动，在这些神经元之间搭建的神经网络能够保证整个 CPG 处于自激振荡的模式，向外发出节律性的信号控制各关节，实现机器人节律性的运动。通过建立合理的神经网络模型，可以生成稳定的机器人步态。基于 CPG 的步态控制的优势在于其不需要机器人过多的结构参数及运动参数，且适应性较强，对地形的一些较小的干扰如坡面、崎岖等有一定的自适应能力。采用 CPG 作为机器人步态控制策略的代表机器人包括了德国 Dillmann 等人设计的 BISAM 机器人，该机器人能够实现四足静态行走；我国清华大学研制的四足机器人 Biosbot 也是一款应用了 CPG 控制策略的四足机器人，该机器人能够进行低速下稳定的步行，同时可以行走于小倾角的坡面。

3. 本课题的研究内容

使用无刷电机制作一个直驱式的机械狗，满足比赛任务需要，能够具有基本行走，对角小跑，定航向行走等功能。

1. 根据系统要求拟定系统的总体设计方案;
2. 设计制作出四足机架的基本结构，腿部结构，足端结构等；
3. 设计并且制作电路板，完善电路的基本功能，电路板设计；
4. 编写控制模块的程序，使用单片机完成控制；

4. 本课题研究的实施方案

机械结构部分：我们使用了电机直连的方式，小腿和大腿的长度分别为 10cm 和 20cm，机架部分的宽度和长度分别为，40cm 和 50cm，在机架部分我们还给了专门的 5 度的斜角以保证机身的稳定性。

控制设计：机器人的全部都基于 stm32，通过 CAN 总线控制八个电机驱动机器人的运动。

5. 进度安排

2018.12.20~2019.2.28 前期阶段：完成毕业设计选题，查阅有关论文、资料，了解本课题的现状及动态，拟定具体实施方案，完成开题报告；

2019.3.1~2019.3.30 电路设计阶段：完成控制模块电路设计；完成英文资料的翻译；

2019.4.1~2019.4.20 软件设计阶段：完成系统软件设计；

2019.4.21~2019.5.13 制作调试阶段：制作演示系统并调试；

2019.5.14~2019.5.25 文档整理编辑阶段：整理设计文档，撰写并完善设计说明书；

2019.5.26~2019.5.30 答辩准备阶段：答辩 PPT 准备，打印装订毕业设计说明书（或论文）、英文翻译资料、图纸等有关资料。

6. 参考文献

［1］ 辛全彬. 小型电动直驱仿生四足机器人系统设计[D].大连理工大学,2018.

［2］ Massachusetts Institute of Technology; Mini cheetah is the first four-legged robot to do a backflip[J]. NewsRx Health & Science,2019.

［3］ 梁美彦.基于 STM32 的四足仿生机器人设计与实验研究[J].测试技术学报,2019,33(01):34-42.

［4］ 魏兵，喻全余，孙末． 机械原理［M］． 武汉：华中科技大学出版社，2011.

［5］ 王维，王建晓． 机械设计［M］． 武汉：华中科技大学出版社，2011.

［6］ 朱雅乔,陈国松,王姣姣,范广宏.四足机器人仿生关节的研究现状综述[J].机械传动,2019,43(01):159-164.

［7］ 陈榕婷,叶泳仪,杨柳娟,李金林,郑誉煌.基于 Solidworks 的四足机器人的步态分析[J].科学技术创新,2019(07):70-71.

［8］ 李岩.四足步行机器人结构设计分析[J].山东工业技术,2019(10):138.

［9］ 谭浩强． C 程序设计［M］． 北京：清华大学出版社，2010.

［10］ 赵大兴，高成慧，谢跃进．现代工程图学教程［M］． 武汉：湖北科学技术出版社， 2009.

指导教师意见

指导教师签名：

年 月 日

湖北工業大學

毕业设计（论文）学生申请答辩表

课 题 名 称	自动四足信使机器人设计				
指导教师（职称）	许万（教授）				
申 请 理 由	毕业设计已按要求完成。				
学生所在学院	机械工程学院	专业班级	15 创新_机自 1	学号	1510100413

学生签名：　　日期：

毕业设计（论文）指导教师评审表

序号	评分项目（理工科、管理类）	评分项目(文科)	满分	评分
1	工作量	外文翻译	15	
2	文献阅读与外文翻译	文献阅读与文献综述	10	
3	技术水平与实际能力	创新能力与学术水平	25	
4	研究成果基础理论与专业知识	论证能力	25	
5	文字表达	文字表达	10	
6	学习态度与规范要求	学习态度与规范要求	15	
是否同意参加答辩：			**总分**	
评语	指导教师签名： 另附《毕业设计（论文）指导记录册》　　年　月　日			

湖北工業大學

毕业设计（论文）评阅人评审表

学生姓名	余磊涛	专业班级	15 创新_机自 1	学号	1510100413
设计（论文）题目	自动四足信使机器人设计				
评阅人		评阅人职称			

序号	评分项目（理工科、管理类）	评分项目(文科)	满分	评分
1	工作量	外文翻译	15	
2	文献阅读与外文翻译	文献阅读与文献综述	10	
3	技术水平与实际能力	创新能力与学术水平	25	
4	研究成果基础理论与专业知识	论证能力	25	
5	文字表达	文字表达	10	
6	学习态度与规范要求	学习态度与规范要求	15	
			总分	

评语	

评阅人签名：_____

年　月　日

湖北工業大學

毕业设计（论文）答辩表及成绩评定表

<table>
<tr><td colspan="2">学生姓名</td><td>余磊涛</td><td>专业班级</td><td colspan="2">15 创新_机自 1</td><td>学号</td><td>1510100413</td></tr>
<tr><td colspan="2">设计（论文）题目</td><td colspan="6">自动四足信使机器人设计</td></tr>
<tr><td>序号</td><td>评审项目</td><td colspan="4">指　　标</td><td>满分</td><td>评分</td></tr>
<tr><td>1</td><td>报告内容</td><td colspan="4">思路清新；语言表达准确，概念清楚，论点正确；实验方法科学，分析归纳合理；结论有应用价值。</td><td>40</td><td></td></tr>
<tr><td>2</td><td>报告过程</td><td colspan="4">准备工作充分,时间符合要求。</td><td>10</td><td></td></tr>
<tr><td>3</td><td>创　　新</td><td colspan="4">对前人工作有改进或突破，或有独特见解。</td><td>10</td><td></td></tr>
<tr><td>4</td><td>答　　辩</td><td colspan="4">回答问题有理论依据，基本概念清楚。主要问题回答准确，深入。</td><td>40</td><td></td></tr>
<tr><td colspan="6"></td><td>总分</td><td></td></tr>
<tr><td>答辩组评语</td><td colspan="7">主要包括：设计（论文）成果的规范性、正确性、创新性，不足之处，学生答辩过程中回答问题的表现，总体评价如是否完成毕业设计任务要求及完成质量。

答辩组组长（签字）：　　年　月　日</td></tr>
<tr><td rowspan="3">答辩委员会意见</td><td>成绩类别</td><td>指导教师评定成绩</td><td>评阅人评定成绩</td><td>答辩组评定成绩</td><td colspan="2">总评成绩
（Ⅰ×40%+Ⅱ×20%+Ⅲ×40%）</td></tr>
<tr><td></td><td></td><td></td><td></td><td colspan="2"></td></tr>
<tr><td colspan="6">答辩委员会负责人（签字）：　　年　月　日</td></tr>
</table>

注：总评成绩应按四舍五入取整。

摘 要

本文主要从机械结构设计、嵌入式控制系统硬件设计和嵌入式控制系统设计三个部分对四足机器人进行详细的说明。

机械设计部分主要有直驱四足机器人的腿部设计，机器腿部与机身连接机构设计，机器机身部分设计，令牌举升机构设计，脚部设计。

嵌入式控制硬件系统设计介绍了 stm32 最小系统设计，动力电路的设计，电压检测电路的设计。

控制系统的软件设计包括了 freertos 的介绍，任务关系，正弦轨迹生成器，周期步态算法，外设通信，计算机视觉等。

关键词：直驱四足；周期步态算法；正弦轨迹生成器； STM32 ；

Abstract

In this paper, the mechanical structure design, embedded control system hardware design and embedded control system design are described in detail.

The mechanical design includes the leg design of the direct-drive quadruped robot, the connecting mechanism between the leg and the fuselage, the fuselage design, the token lifting mechanism design and the foot design.

The design of embedded control hardware system introduces the design of STM32 minimum system, power circuit and voltage detection circuit.

The software design of the control system includes the introduction of freertos, task relationship, sinusoidal trajectory generator, periodic gait algorithm, peripheral communication, computer vision and so on.

Key words: direct drive quadruped; periodic gait algorithm; sinusoidal trajectory generator; STM32

原创性声明

本人郑重声明:所呈交的学位论文是本人在导师的指导下独立进行研究所取得的研究成果。除了文中特别加以标注引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写的成果作品。本人完全意识到本声明的法律后果由本人承担。

作者签名： 日期： 年 月 日

学位论文版权使用授权书

本学位论文作者完全了解学院有关保管、使用学位论文的规定，同意学院保留并向有关学位论文管理部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权省级优秀学士学位论文评选机构将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等 复制手段保存和汇编本学位论文。

本学位论文属于

1、保密 □，在 年解密后适用本授权书。

2、不保密 □

（请在以上相应方框内打“√”）

作者签名： 日期： 年 月 日

导师签名： 日期： 年 月 日

目 录

一 绪论

1.1 课题研究背景及意义

自动信使四足机器人是全国大学生机器人大赛 robocon 2019 的机器人其中之一。比赛在所示的场地（如图 1.1）上进行，红、蓝两队各占一半。比赛最多持续 3 分钟。

每支参赛队有一个名为“手动信使机器人”的手动机器人和一个名为“自动四足信使机器人”的自动机器人。手动机器人携带的令牌作为信物的从“龙门驿”（手动机器人的启动区）启程。 它沿着树林、桥梁行走，越过界线 1，到达“大漠驿”（自动机器人的启动区）。这时，手动信使机器人将令牌交给在大漠驿的机器自动四足信使机器人。

一旦令牌被自动四足信使机器人成功地接收到，它就可以沿着大漠区前进。其次自动四足信使机器人必须有四条腿像马一样，不能用轮子前行。自动四足信使机器人通过沙丘和草地，向“高山驿”进发。自动四足信使机器人到达高山驿后，手动信使机器人可以进入投掷区投掷兽骨得分。

如果手动信使机器人获得 50 或更多得分，自动四足信使机器人就可以登山。此后，如果它到达山顶区，并且举起了令牌，则该队获胜，这就是“登顶”。

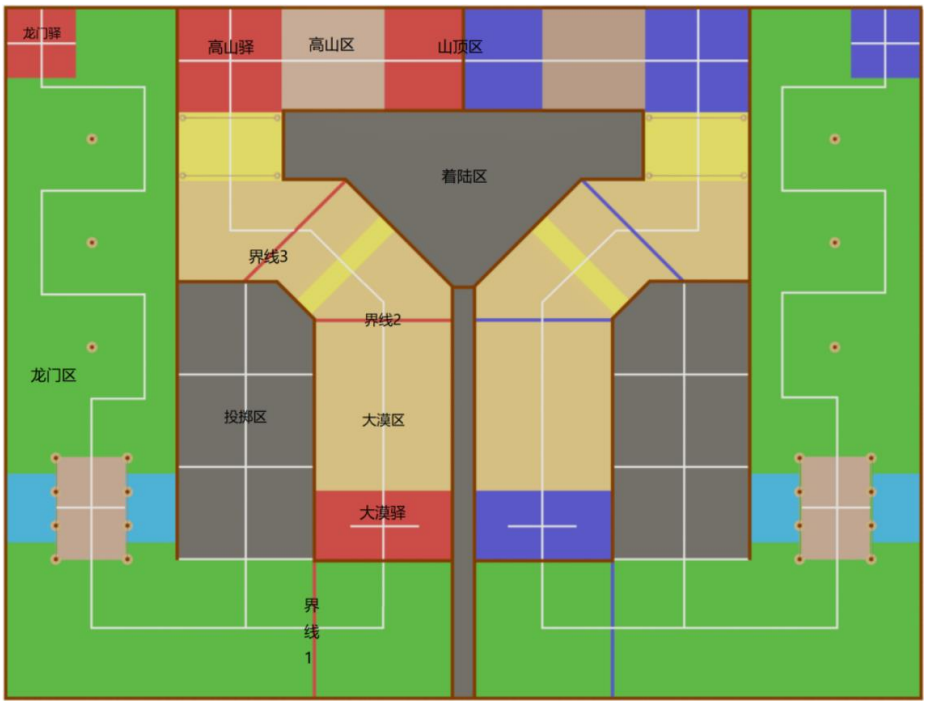

图 1.1 场地示意图

1.2 国内外发展现状

早在机器人学兴起之前，就有生物学家对于四足动物的运动进行了深入的研究。代表的研究有 1975 年 McMahon [1]使用跑步机和高速摄像机对从老鼠到马的各种动物的运动进行了观察研究，得到了动物的步频、步长、最大速度等步态参数与动物质量有幂函数关系的结论；以及 1977 年 Cavagna[17] 等人用力台得到了四足动物行走和奔跑时得到触地力曲线等。这些研究的结果具有普遍适用性，对于随后足式机器人的研究有着指导性和参照性的意义，有力支撑了仿生足式机器人学科的发展。

对于四足机器人的研究公认始于 1968 年美国 GE [8]公司的工程师 Mosher 的 Walking Truck[2]，该机器人是一个没有计算机控制、完全由工作人员对机器人各个执行器进行控制以实现其运动的四足机器人。在四足机器人的发展过程中，主要出现了如下几种主流的机器人步态控制方法。

1.基于静态稳定的机器人步态控制[10] 在四足机器人的早期研究中，计算机及电子、电气技术发展的限制使得机器人的控制水平较差，导致机器人的动态性能很差，只能基于静态稳定的控制策略进行慢速、静步态的运动，这种基于静力学的机器人控制方法[13]在生成步态时必须保证机器人的重心在地面上的投影必须始终位于机器人各支撑腿形成的多边形之内，同时在运动控制时令重心尽量远离多边形的边缘以提高稳定裕度。

2.基于动力学模型行为特性的机器人步态控制[21]第一台基于动力学模型行为特性的机器人是美国麻省理工大学的 Raibert 等人在 1982 年设计的 MITleg [4]单足机器人。该机器人以弹簧-负载倒立摆模型[20]（SLIP 模型）作为动力学模型的参照，通过对于弹簧负载倒立摆动力学模型的分析成功地找到了对于单足机器人的前进速度、竖直弹跳高度该方法以 PID 控制[14]规律通过控制腿部落地角控制机器人前进速度，通过控制腿部着地力控制竖直弹跳高度，通过控制髋关节力矩控制俯仰角度，在机器人的实验上获得了较好的效果。这种控制方法的每一个控制量都单独对应着一个动力学上的量，故称这种方法为基于动力学特性的机器人控制方法[5]。Raibert 等人随即将这种方法的应用拓展至双足、四足机器人上。通过将同时着地的两条腿视作一条虚拟腿[7]，合理规划机器人的各足的着地角、足底力和髋关节力矩，麻省理工的四足机器人[11] 可以进行中速下对称步态的运动，如对角小跑步态（trot 步态）、跳跃步态（bound 步态）下的行进。

3.基于被动行走的机器人步态控制[16]被动行走，是指机器人能够利用自身的动力学特性，令自身的重力势能、弹性势能及动能合理地转化，从而能够在不依靠控制及主动力驱动的情况下自发行走。这一研究始于 Raibert[3] 等人在 1987 年提出的一种带扭簧及直线弹簧的单腿运动模型，通过合理计算该模型的结构参数，可以使其在理想的无能量损耗状况下持续以单腿弹跳的步态行进。

4.基于中枢模式发生器的步态控制基于中枢模式发生器[17]（Central Pattern Generator, CPG）的步态控制思想来源于低级生物的运动。低等级生物的中枢模式发生器

能够产生节律性的神经信号，从而控制自身的每条腿进行节律性的运动。类似地，机器人的中枢模式发生器也是一个由神经元模型组成的神经网络模型[19]，该神经网络模型中每一个神经元控制一个关节的运动，在这些神经元之间搭建的神经网络能够保证整个 CPG 处于自激振荡的模式，向外发出节律性的信号控制各关节，实现机器人节律性的运动。通过建立合理的神经网络模型，可以生成稳定的机器人步态。基于 CPG 的步态控制的优势在于其不需要机器人过多的结构参数及运动参数，且适应性较强，对地形的一些较小的干扰如坡面、崎岖等有一定的自适应能力。采用 CPG 作为机器人步态控制策略的代表机器人包括了德国 Dillmann 等人设计的 BISAM[15] 机器人，该机器人能够实现四足静态行走；我国清华大学研制的四足机器人 Biosbot[19] 也是一款应用了 CPG 控制策略的四足机器人，该机器人能够进行低速下稳定的步行，同时可以行走于小倾角的坡面。

1.3 本文的主要设计内容

在本文中，我们将研究各种这些技术在实现的新四足动物上的应用独特的 4 杆运动腿设计。直驱电机和 4 杆连杆腿设计有几个优点。首先，直接驱动系统允许高频率步态。腿部配置允许马达位于机器人的臀部，从而减小腿质量和减少这些高速运动期间的惯性效应。高扭矩电机和 4 杆通过产生可变的传输速率，最小化反射时间，腿也减少了齿轮箱的必要性惯性。所使用的无刷电动机通过低扭矩提供足够的扭矩来产生行驶步态工作区的区域可能构成挑战。

第一章，主要讨论四足机器人的背景意义，四足机器人的国内外现状。根据课题背景，论述四足机器人广阔应用环境，其次分析了国内外的现状，为本文研究内容提供了思路。

第二章，对四足机器人的运动原理进行分析，从结构、单腿协调控、多腿协调控制、惯性导航进行了分析。

第三章，四足机器人的结构进行设计，具体分为腿部设计，机身连接机构，机身设计，令牌举升机构设计，脚部设计。

第四章，四足机器人的控制系统硬件设计，主要进行了需求分析，控制电路以及外设电路的设计。

第五章，四足机器人控制系统的软件设计，介绍了控制系统的核心算法，外设通信，机械视觉部分

第六章，对机器人的环保发展问题进行了思考。

第七章，对整篇论文所完成内容进行了归纳总结，同时指出了不足和展望。

1.4 本章小结

本章主要介绍了四足机器人的发展，对比了国内外现状，讨论了移动机器人的研究意义。 针对本课题的研究背景，讨论四足机器人的广阔应用前景。概述了本论文的研究内容与各章节内容。

二 并联腿四足机器人运动原理

本章对四足机器人的行走原理进行了分析，通过分析，运动数学公式分析出足端的轨迹以及四足机器人行走的时候几个腿部的相位关系，进而实现多种步态，层层递进。

2.1 并联四足机器人的结构原理

并联腿四足机器人的结构主要在于机器人的腿上，我们使用了电机对装直连的方式，机器人的腿部示意图如图所示，在通过控制两个电机的角度，可以控制足端的轨迹从而模拟出机器人的正常足端轨迹。如图 2. 1 所示。

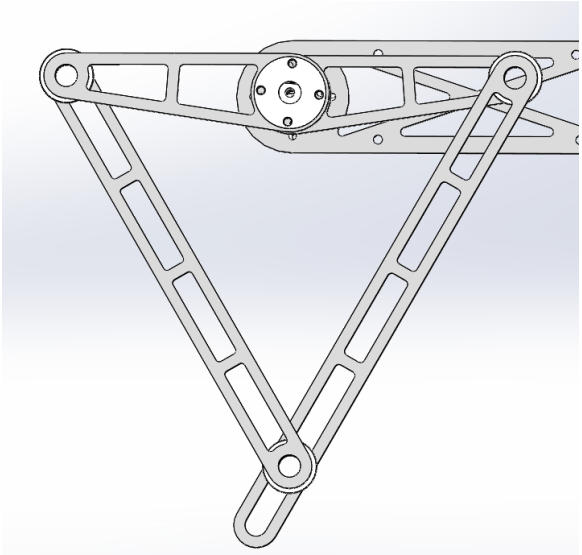

图 2. 1 腿部结构示意图

2.2 足间协调控制（步态）

一条腿轨迹的基础上，进行足间协调控制，才能生成相位，组件协调控制通过建立 4 个腿部振荡器之间的相位耦合关系来实现，四足机器人的典型步态有四种，walk(行走),trot(对角小跑),pace(同侧遛步),gallop(跳跑), walk 步态的特点是四足交替起落，相位差为 0.25，负载因子为 0.75；trot 步态的特点是对角足成对起落；pace 步态的特点是同侧足成对起落；gallop 步态的特点是前后足成对起落。后三种步态的相位差和负载因子均为 0.5。图 3-17 所示为四种步态对应的四条腿之间的相对相位关系。图中 LF、RF、RH、LH 分别代表左前腿、右前腿、右后腿、左后腿。

在图中，以机器人左前腿为基准，将其相位定义为 L＝0，这样由四种步态对应的相位差即可得到其他三条腿的相位，通过分析得到以下规律: 如图 2. 2 所示。

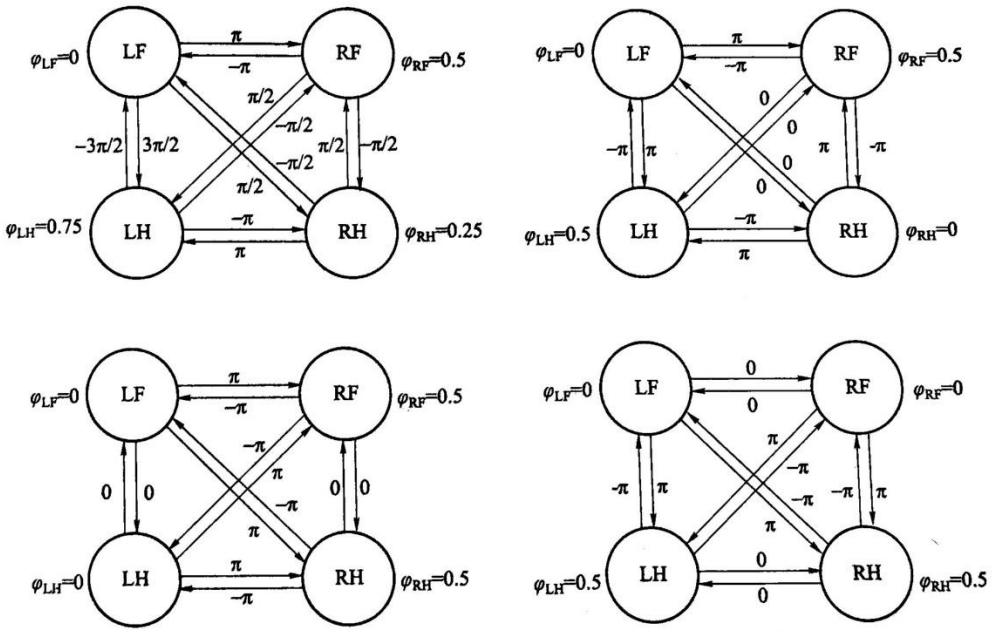

图 2.2 四种典型步态示意图

2.3 足内协调控制（足端轨迹）

足端轨迹的规划是一个四足机器人的重要部分，在一般的研究中，一个周期可以被分为两部分，（1）支撑相，（2）摆动相，如图 2.3 所示。

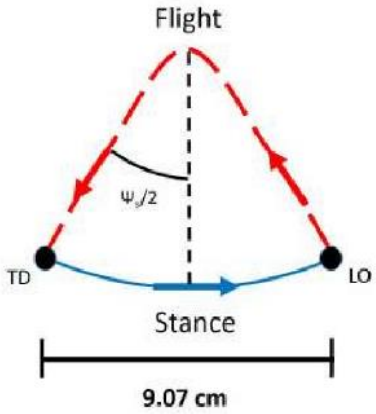

图 2.3 足端轨迹示意图

2.4 惯性导航

四足机器人开环行走的时候，由于机械误差以及足端摩擦力的影响，不能保持自己的方向，由于比赛需要，需要通过定向的导航来行走。 在增加了陀螺仪的基础上，引入了航

向纠偏。航向纠偏的思路有两种，分别为（1）改变左右两边的腿部的差速进行左右线偏向，（2）改变左右侧腿的占空比（即飞行占比）来改变左右偏向。

实现思路，使用了一个 P(比例系统)，传入设定的角度和陀螺仪反馈回来的真实方向角，比例系统输出了反向的差值，输出正负为左右，输出的值通过运算输出成范围为-21 到+21 的值，引入到振荡器中实时改变左右两侧的腿的步长，达到实时改变方向，纠偏的效果

2.5 本章小结

在本章中分析了四足机器人所需的功能与整体方案设计，从结构，控制两个方面分析了四足机器人的行走原理，步态控制，电机运动控制思路。为接下来的章节将对设计的实现奠定基础。

三 四足机器人机械结构设计

本章节主要介绍并联腿四足机器人的机械结构设计，并联腿四足机器人结构设计主要分为以下几个部分：

（1）机器人腿部设计

（2）机器人腿部与机身连接机构设计

（3）机器人机身设计

（4）机器人交接机构设计

（5）机器人脚部设计

并联腿四足机器人整体如图 3.1 所示：

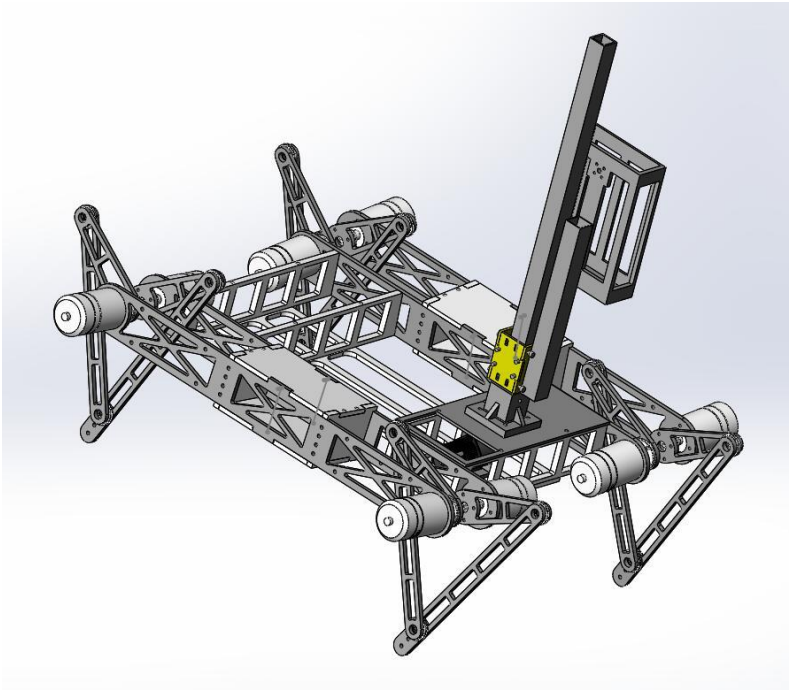

图 3.1 并联腿四足机器人图

以下内容即为机器人各个机构设计的详细介绍。

3.1 机器人腿部设计

3.1.1 腿部四杆机构

一般来讲，设计机器人都是从机身开始设计，但由于并联腿四足机器人的两个驱动电机是对立放，需要通过腿部的间距来确定机身的驱动电机安装位的相对距离，故先从

腿部机构开始设计。

四足机器人要求具有越障能力，故需要机器人的站立高度足够高又不至于在行走的时候倾倒，这样腿的长度就有一定的限制。

首先，我们将腿部机构定位在四杆机构，根据相关文献以及我们的多次试验，我们发现当大腿的长度是小腿长度的两倍时，腿部机构的运动灵活性最高，可以仿真出各种运动路径，而且电机的转速和力矩利用率也在一个可控的范围内。

然后，考虑到四足机器人要跨越 10cm 高、30cm 宽的台阶和 20cm 高的绳子，我们将大腿的长度定在 20cm，将小腿的长度定在 10cm，由于大腿有一根需要与地面接触，故从转动副处延伸 20mm，该处用来承接与地面接触的机构。

另外，为保证腿部有足够的强度，我们将四杆的厚度都设置为 5mm,并采用 5052 型号的镁铝合金板加工。为在保证结构强度的前提下减轻腿部的重量，我们将腿部各机构进行了适当的镂空。

腿部四杆机构如图 3.2 所示：

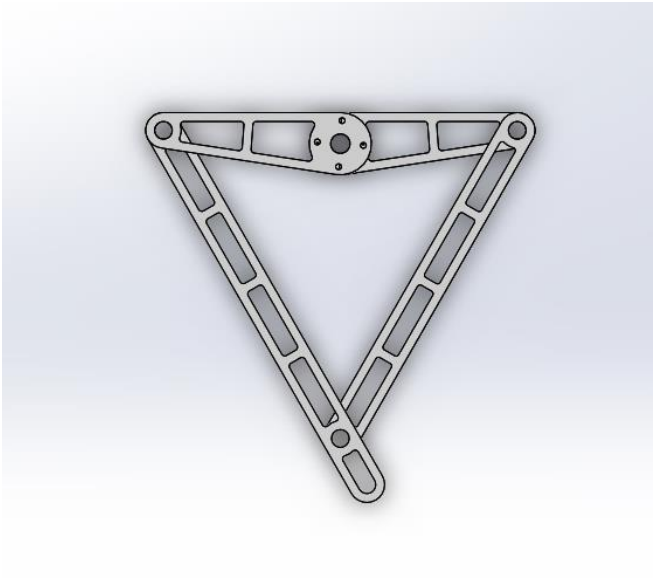

图 3.2 腿部四杆机构图

3.1.2 腿部四杆机构旋转副的组成

设计完四杆机构后，该机构并不是一个完整的机构，还需要用旋转副去连接它们，经过仔细筛选，我们选择了直径为 10mm 的塞打螺钉（如图 3.3）和相对应的防松螺母作为旋转副。

图 3.3 塞打螺钉

另外，为保证两杆之间没有相对接触使其旋转顺畅，我们选择了内径为 10mm,外径为 24mm,厚度为 4mm 的平面推力轴承。经过多次试验，我们确定该旋转副稳定可靠而且顺滑。

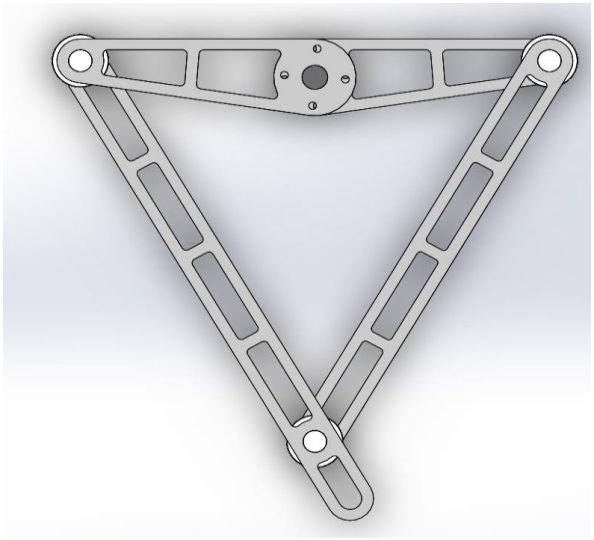

图 3.4 腿部四杆机构+旋转副图

3.2 机器腿部与机身连接机构设计

设计完腿，我们需要设计腿部与电机的连接装置，我们的电机是使用的大疆公司制造的 M3508 电机（如图 3.5），它是一款无刷电机，具有很大的力矩以及轴向承受能力，该电机轴的直径为 10mm，并配有径向定位槽和轴向定位孔。

图 3.5 M3508 电机

该公司也出售了相应的连接机构法兰盘，但并不适用用本机器人。且网上的法兰盘都是按国家标准制作的，并不能满足我们对法兰盘的需求。故我们自己设计法兰盘（如图 3.6），并外包制作，既满足了我们的个性化需求，又保证了法兰盘与电机轴的装配精度。

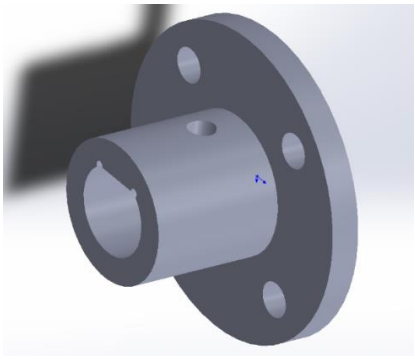

图 3.6 定制法兰盘

电机与腿部连接如图 3.7：

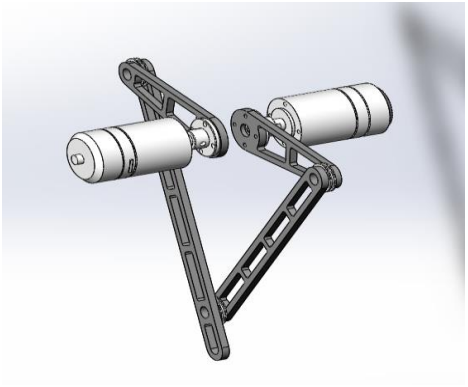

图 3.7 电机与腿部连接图

3.3 机器机身设计

确定好了机器人的腿部机构以及电机与腿部的连接机构，也就知道腿部的纵向距离了，这样就可以得出两电机的相对距离，并设计出相相应的连接架。为了保证连接架的强度以及方便加工，我们也使用了 5052 型号的镁铝合金，并将其厚度定在 5mm,其连接处全部采用直径为 4mm 的孔。为保证机架（如图 3.8）的轻盈以及外观具有欣赏性，我们将电机的承接板进行了适当的镂空。

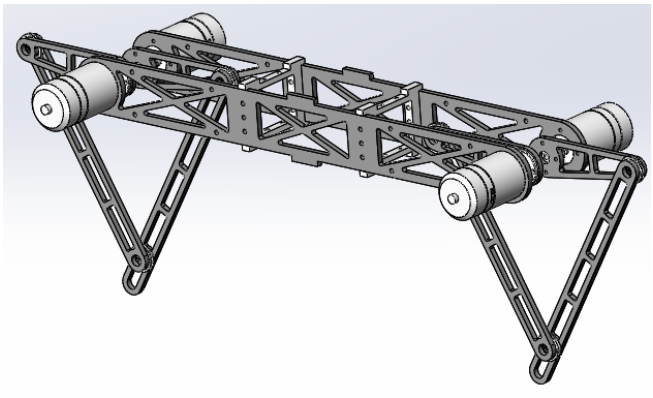

图 3.8 机架

对于两承接板之间的连接板（图 3.9），我们采用了 8mm 的镁铝合金板并镂空。

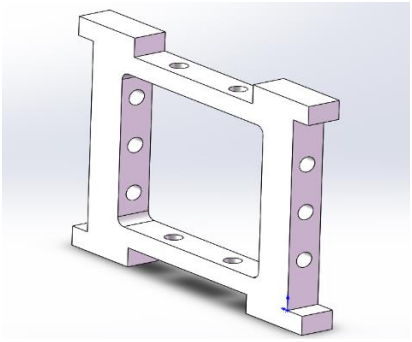

图 3.9 连接板

其连接处使用 M4 的螺纹孔，连接螺钉选用 M4x16 的螺钉，并设置 6 处连接位，在保证了连接强度的基础上，也不破坏电机承接板的强度。

为了防止螺钉因到大量剪切力而发生断裂给维修带来很大的麻烦，我们在连接板和电机承接板之间设计了两块加固板（图 3.10），将连接板和电机承接板卡住，并用 M4 的螺钉连接，使两边的连接螺钉几乎不受剪切力而能维持正常的功能。

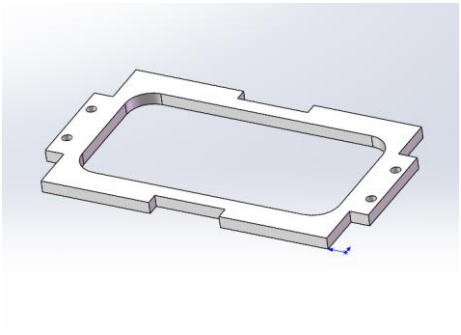

图 3.10 加固板

机器人机身半边结构如图 3.11 所示：

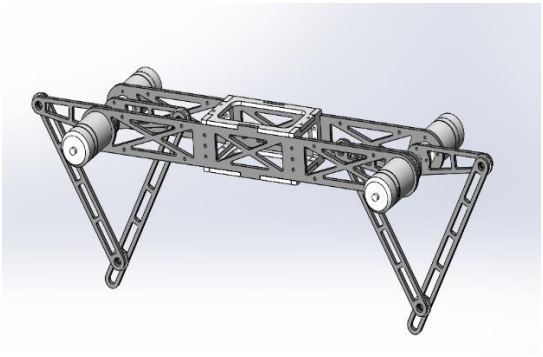

图 3.11 机器人机身半边结构

设计完了机器人机身半边结构之后，也只是完成了两条腿的连接，作为一个四足机器人，是需要有四条腿的，故还需要另外两条腿的连接，即需要中间承接板（如 3.12）。考虑到两电机的长度，我们将中间承接板的长度定为 220mm，

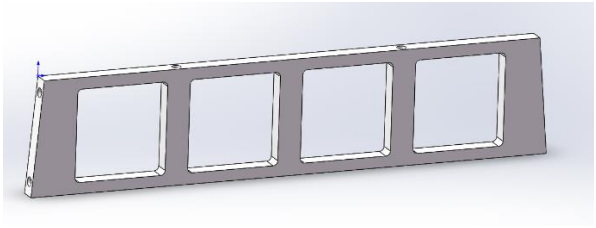

图 3.12 中间承接板

因为有打螺纹孔的需求，故该中间承接板的厚度为 8mm,材料选择 5052 型号的镁铝合金板，螺纹孔为 M4 的孔。为了保证并联腿四足机器人在地上行走的稳定性，我们给机器人的机身两边各设计了 5 度的倾角，即将中间承接板设计为梯形，为保证机器人的连接强度，我们采用了 4 块中间承接板。将上述机构，即得出机器人的机身（如图 3.13）：

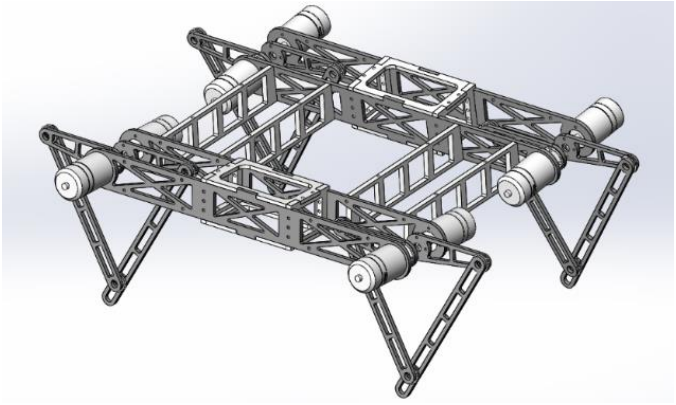

图 3.13 机器人机身图

除此之外，由于并联四足机器人内部还要安装主控板、action、电机电调等电子元件，故还要设计一个底板（图 3.14）。

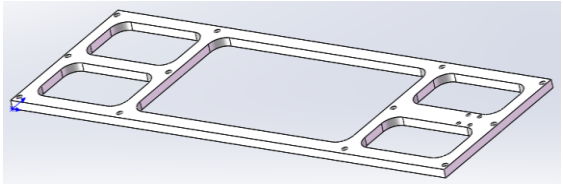

图 3.14 底板

我们采用 5052 型号的镁铝合金板，厚度选择 5mm，由于该处不是承力处，故采用大幅度镂空。机器人机身总图如下（图 3.15）：

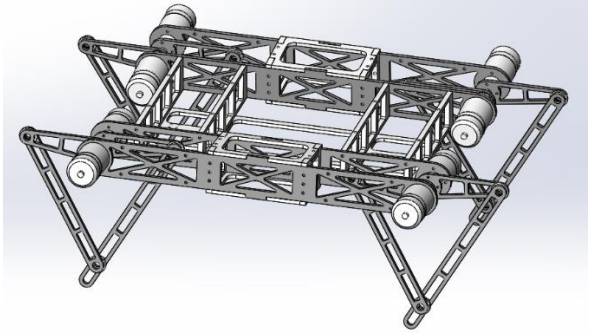

图 3.15 机器人机身总图

3.4 四足机器人令牌举升机构设计

在四足机器人完成登顶动作之后，需要将手持的令牌举起，使令牌最高沿离地面至少1000mm，考虑到四足机器人在登顶前的行走过程中要高速行走和起跳，在保证落地机身稳定性的同时要求举升机构不能对机身平衡性造成太大的影响，所以采用分截式升起机构，采用类似卷扬机卷起重物原理，通过电机卷起绳子，绳子通过滑轮与滑台相连接，固定令牌的盒子通过一根铝制管与滑台固定为一体，因此，在电机卷起滑台的同时，令牌盒带着令牌升起(如图 3.16)。

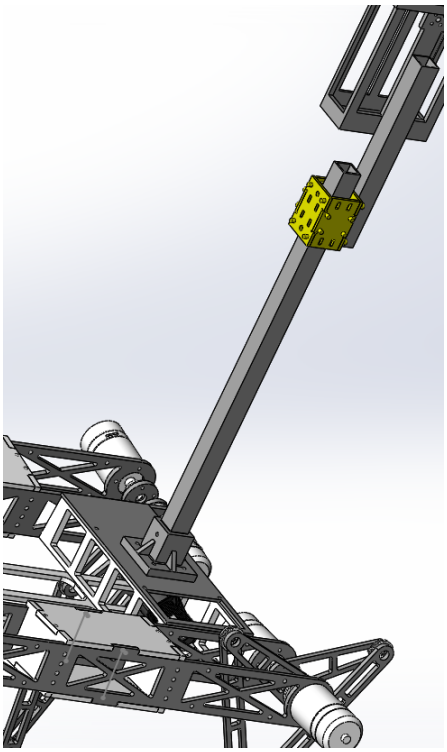

图 3.16 令牌升降装置图

执行机构采用的电机扭矩适中，经过测量，举升机构所要承担的重量为令牌盒，滑车，200mm 铝制管，舵机，以及启动所用的红外光电传感器。为了尽量减轻机身的负担，减小升降电机的负载，盒子在原先玻璃纤维板模型的基础上，改选用 3D 打印的 ABS 材质盒子，重量更轻，强度适中，韧性较好。整套盒子下来大约重 180g，盒子用 M4 的孔与 200mm 铝制管通过螺栓连接，上防松螺母，保证连接的稳定性。

滑车装置，通过自主设计达到以一种低成本的方式达到与价格昂贵的导轨同等的效果，根据所用铝制管的外部尺寸 20mm×20mm 进行设计，使滑车(如图 3.17)内部恰好容得下方管。

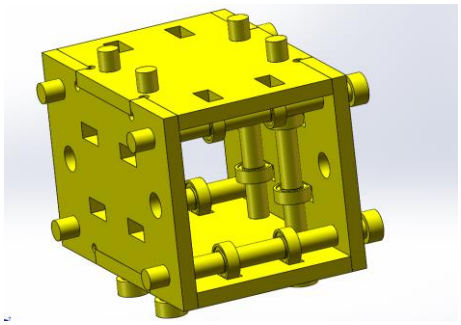

图 3.17 滑车装置

滑车相邻两片板子采用槽式楔合连接，在每个相对的面用 4 个 M4×45 的长螺栓相连接，整体尺寸 37×37（不考虑螺栓尺寸）。体积小巧，重量较轻。滑车的重要滑动功能的实现依靠 16 个每面 4 个外径 6mm、内径 4.5mm 的小轴承。轴承与螺栓之间为间隙配合，保证滑动的流畅性。四面的面板一般情况下可以采用碳素纤维板，铝制板，雕刻定制。考虑到成本和使用环境并没有严苛的强度问题，我们采用 3mm 的玻璃纤维板，用雕刻机雕刻成四块板子，手工装配。另外为了方便滑车与其他结构的装配，在四个面上可以选择性根据实际要求布置所需要的孔。因此这里在其中两面打了两个 M5 的孔。方便铝方管的安装。

升降中所用到的两根铝制方管，为了达到功能要求，配合盒子本身的高度，需要升起的高度为 600mm。考虑到四足机器人行动过程中的跳跃运动，要求中心尽可能的靠底盘，通过降低重心达到稳定机身的效果。选用 400mm 的主杆加 200mm 的升降杆搭配(如图 3.18)。

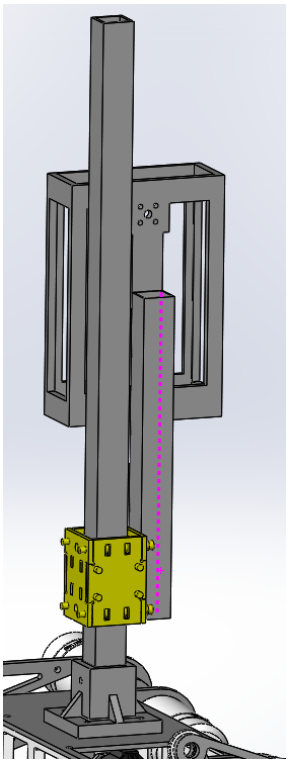

图 3.18 升降机构示意图

主杆与底盘的连接采用 3D 打印连接件(如图 3.19)

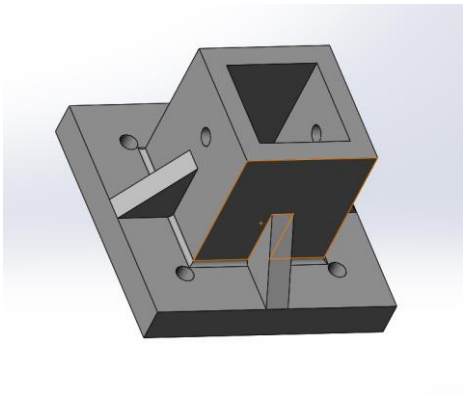

图 3.19 底盘连接件

侧边打 M3 的孔与铝制方管进行 Z 轴的定位。在 Z 轴与平面的连接上，采用四个筋肋的结构增加纵向与平面的承重能力。筋厚 10mm，内部 20mm×20mm 的方管孔因为 3D 打印的精度给 0.1mm 的余量。底面通过四个 M4 的孔与底盘上的玻璃纤维板进行紧固。

由于与电机相连接的联轴器为内接直径 15mm 圆的六边形。通过 M3×6 的短螺钉与电机轴上自带的定位槽进行定位。联轴器本身长度为 20mm，放置位置恰好与铝方管内孔 18mm 相吻合。经过实际的实践卷起效果测试，发现由于电机转速较快，绳子在 20mm 的联轴器上容易移出联轴器，达不到想要的卷起效果。因此在设计过程中，在旁边的侧面上利用轴向的 M3 螺纹孔，考虑到机身的上底板和下底板之间的空间要求，加上一个直径 30mm 的圆片，防止绳子做卷起运动时卷出联轴器(如图 3.20)。

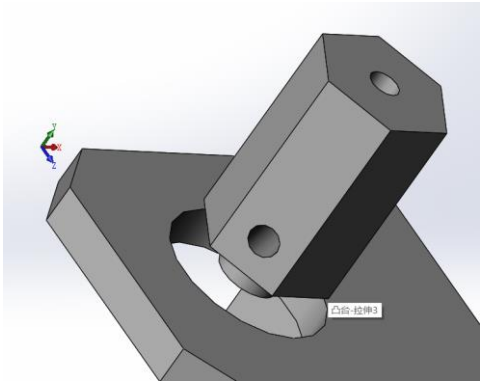

图 3.20 卷连器

在主杆的顶部需要用滑轮将绳子转向，在保证转向功能要求的同时，还必须考虑绳子不能够与机架本身产生摩擦，减少绳子的使用寿命，避免在真正比赛过程中造成绳子的断裂。因此 M3 的孔选择打在靠近面的一边，中间选择用同步带轮来代替滑轮(如图 3.21)，中间的孔为直径 4mm 的洞，滚轮直径 10mm，确保在滚动时不会出现卡死的情况，

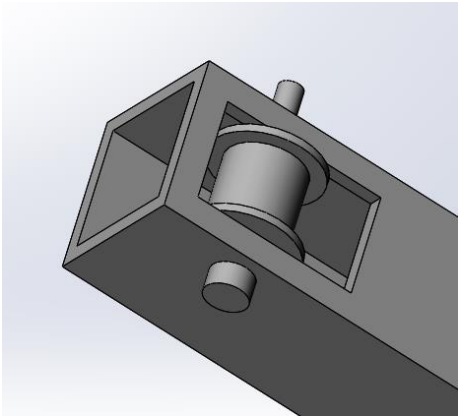

图 3.21 定滑轮结构

最后的连接绳子的选材，经过近一个月的实验，最初使用现有的织毛衣的绳子，在卷起过程中，不能承受电机瞬时加速造成的冲击，经过几次的升降之后从中间断开了。此后选用了强度较高的铁丝绳，直径 2mm 的铁丝绳。但是在实验过程中发现由于铁丝的弯曲限度有限，不能够很好的贴合电机联轴器，造成与铝方管卡滞。虽然在强度上有了保证，但是柔韧性和实用性都不太强。最后在老师提醒之下选用了钓鱼用的鱼线，型号选用 9 编 PE 材质线号 3.5mm 的大力马鱼线，承受重力 160N。但是在实际的卷起实践过程中发现，静置以及跳跃过程中，松弛的线碰到铝方管会有一定的磨损，尽管线的耐磨耐拉能力很强，但是为了保险起见，还是选择三股绳子编织在一起使用。线的一头与联轴器上 M3 的螺丝打水手结，防止脱散。电机放在主杆铝方管 18mm×18mm 孔的下方，使联轴器正对孔。线沿孔向上延申 400mm 至滚轮出通过滚轮换向，滚轮正好把绳子与铝管壁隔开，避免产生摩擦。换向后向下与滑车相连接，滑车与铝管通过两个 M5 的长螺钉相连，绳子与螺钉缠绕之后用防松螺母紧固。考虑到更换的原因，绳子适当的加长，多余的部分可以通过联轴器卷起来储存使用，并不影响正常功能的使用。

在交接之前的静置状态下，因为腿部步态可以自由调整，交接高度的调整较为灵活，这里暂时设定令牌盒的最高处高度≤60mm，一方面使重心尽可能的底，另一方面为了 M1 机器人能够较为容易的完成交接动作(如图 3.22)。

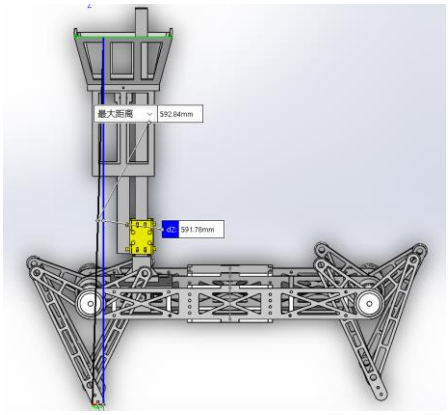

图 3.22 举升示意图

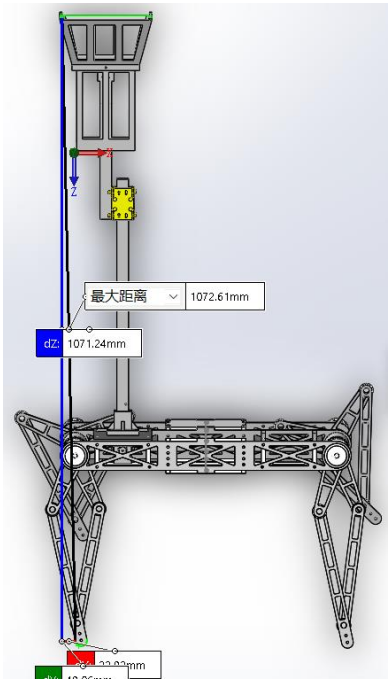

图 3.23 最大高度示意图

如图(3.23)所示，在腿部能够站起的限度范围内，举升机构能够举起的高度为 1070mm，不考虑令牌所占的额外的高度，已经能够完全满足功能要求。实际上，综合腿部支撑的高度，我们可以适当的进一步增加主杆的长度，并减小升降杆的长度，达到再次降低重心的目的，把对四足机器人行动过程中可能造成的影响降低至最小。但这里的性能要求已经满足，实验后发现对平衡影响并不大，所以可以直接采用该种方案。

3.5 机器人脚部设计

由于机器人是要在各种环境下进行行走，所以机器人的脚部需要与地面有较大的摩擦力，就需要较大的接触面积，故单个 5mm 的大腿并不能满足机器人的这个需求，故还要在腿上增机构，但由于设计之初并没有考虑到这点，没有设计脚步安装位，而整个机架已经加工出来了，只能在现有的基础上进行改进。于是我们利用大腿伸出部分的镂空部分，设计出带两个螺纹安装位的 5mm 塞打片（如图 3.24），并采用 5mm 的环氧板加工得出。

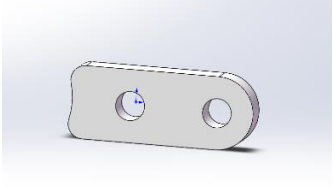

图 3.24 5mm 塞打片

其刚好塞打进腿部伸出部分的镂空部分，成为完整腿（如图 3.25）。

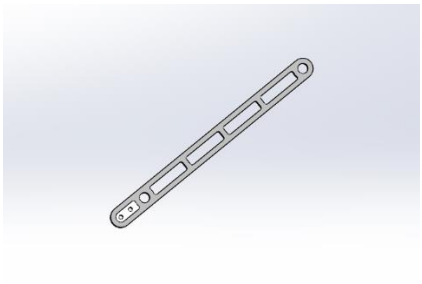

图 3.25 完整腿

由于要增大与地面的接触面积，故采用 5mm 的环氧板加工出脚片（如图 3.26）。

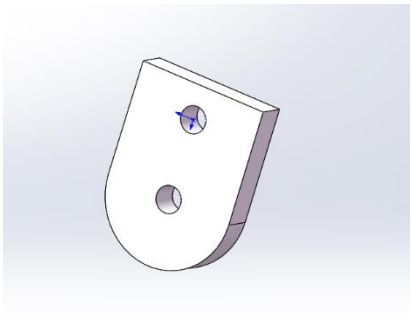

图 3.26 5mm 脚片

再将其脚片装在大腿延伸部分的两侧，由于采购的 M4 螺丝长度有限，故腿部两边各装两片脚片，再用螺钉连接起来。考虑到脚的摩擦力以及耐磨强度，我们选用了 2mm 的橡胶片作为脚底板，并两块叠加，得到可使用的脚（图 3.27）。

图 3.27 可使用的脚

3.6 本章小结

本章中主要对四足机器人的机械结构进行总结，四足机器人的主要结构为腿部，机身，令牌举升，脚部。

四 控制系统硬件设计与实现

四足机器人选用了 stm32f427 作为主控制器，由于四足运动是一个运算量较大的工程，stm32f427，具有大容量，高达 180MHZ 的主频提供了强大的性能。

硬件外设分为以下几块

（1）主控板通过 CAN 总线通讯与电机驱动通讯，然后电机驱动根据主控板的指令输出不同的电压与电流以驱动电机。

（2）主控板通过 RS232 通讯与定位模块通讯，在通讯过程中主控板只读取信息。

（3）主控板通过串口通讯与电脑通讯，主要是用于发送当前控制器的运行参数。

（4）主控板通过 I/O 口与显示屏、按钮、红外测距和红外光电开关。

4.1 控制系统硬件综述

4.1.1 需求分析

对于四足机器人本身，需要有一个基本的行走功能，这个行走的功能涉及到很多的方面。

对于比赛任务，四足机器人需要完成行走、跨越台阶，跨越绳子，上坡等任务。

对于行走，需要实时的对步态进行纠偏。

4.1.2 系统硬件总框架

在上一小节中提出了四足机器人的功能需求，接下来对每个需求进行硬件上的方案设计。

4.1.3 STM32F427 芯片介绍

STM32F427 / 437 系列 MCU 面向需要高集成度，高性能，嵌入式存储器和 10 x 10 mm 小型外设的医疗，工业和消费类应用。

STM32F427 / 437 微控制器采用 CortexTM-M4 内核（带浮点单元），工作频率为 180 MHz，静态功耗（下行模式）低于 STM32F405 / 415/407 / F417。

性能：在 180 MHz 时，STM32F427 / 437 微控制器使用 ST 的 ART 加速器，在从闪存执行时，在零等待模式下提供 225 DMIPS / 608 CoreMark 性能。 DSP 指令和浮点单元扩展了应用范围。

能效：该系列 MCU 采用 ST 的 90 纳米工艺和 ART 加速器，在运行模式下执行闪存时的程序，动态功率调节，260μA/ MHz（@ 180 MHz）实现低功耗）。关断模式消耗 100μA（典型值）。这是 STM32F405 / 415/407 / F417 的三倍。

STM32F429 / 439 与此相反，STM32F427 / 437 微控制器没有内置的 LCD-TFT 控制器接口。但是，您可以通过并行或串行接口连接到显示器。您还可以使用 ST 的 Chrom-ART 图形加速器创建内容，速度是内核的两倍。除了复制原始数据外，Chrom-ART AcceleratorTM 还支持其他功能，如图像格式转换和图像混合（图像混合的指定透明度）。因此，Chrom-ART 加速器可加速图像内容的创建，并节省 MCU 内核在其他应用中的处理带宽。

音频：支持专用音频 PLL 的串行音频接口（SAI），两个全双工 I2S 和一个新的时分复用（TDM）模式。

提供更多接口，最多 20 个通信接口（4 个 USART，4 个 UART，最高 11.25 Mbps，6 个 SPI，最高 45 Mbps，包括 3 个新功能）。

I2C，2 CAN，1 SDIO 的数字滤波器功能可选。

模拟：两个 12 位 DAC，三个 12 位 ADC，每秒 2.4 M 样本或每秒 7.2 M 样本（交错模式）

最多 17 个定时器：16 位和 32 位定时器，最高 180 MHz。

通过灵活的内存控制器轻松扩展存储容量，该控制器支持高达 90 MHz 的 Compact Flash，SRAM，PSRAM，NOR，NAND 和 SDRAM 内存以及 32 位并行接口。

基于模拟技术的随机数发生器。

STM32F 437 实现 AES-128，-19 和-256 的硬件加速，以及支持 GCM 和 CCM，3DES 和散列（MD5，SHA-1，SHA-2）的加密/散列模块。

4.2 嵌入式系统控制电路设计

4.2.1 主控板最小系统控制电路设计

Stm32 的最小系统电路主要有：①晶振电路、②复位电路、③SWD 调试接口。

① 晶振电路：为 STM32 MCU 提供外部的时钟基准脉冲，选用了 12MHZ 的晶振。电路如图 4.1 所示：

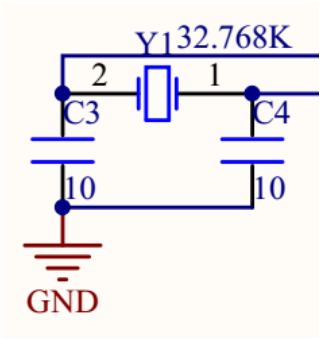

图 4.1 晶振电路图

② 复位电路：为单片机提供复位功能，电路如图 4.2 所示：

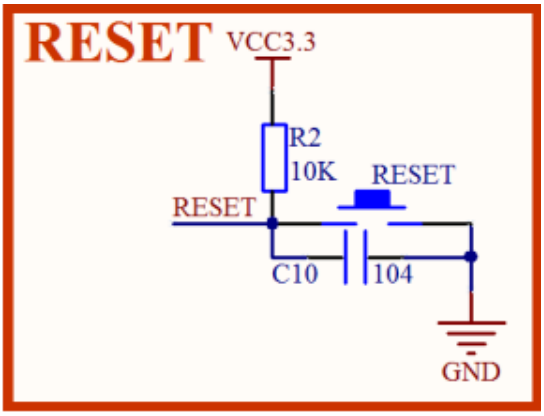

图 4.2 复位电路图

③ SWD 调试接口：配备了一个 SWD 调试接口，SWCLK 和 SWDIO 在开发板中串联了 100 欧姆的电阻起到保护单片机的作用，电路如图 4.3 所示：

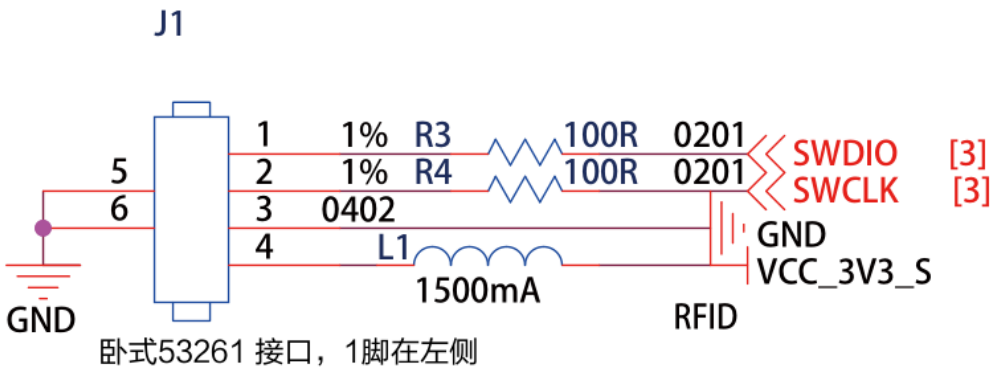

图 4.3 SWD 调试接口

4.2.2 主控板电路防反接、缓启动、过压保护设计

主控板采用了标准的 xt30 接头，输入进来的电源首先并联 28 TVS 管，可防止瞬间高压烧坏主控板，通过 PMOS Q 1 和 NMOS Q2 组成缓启动与防反接电路，可防止电源接触孙瞬间打火造成的接头损坏，和电源正负极反接造成的事故，齐纳二极管可以在输入电压超过其击穿电压（30+-2V）时，三极管 Q3 导通，PMOS Q1 断开，起到过压保护的作用(如图 4.4)。

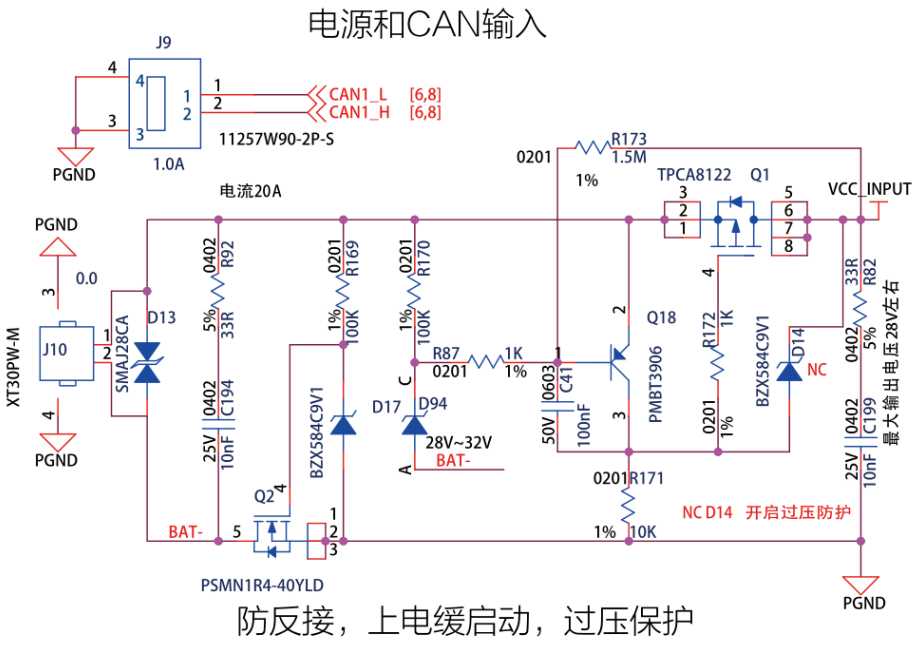

图 4.4 防反接、缓启动、过压保护电路

4.2.3 主控板电源树

主控板电源树，如图 4.5 所示。

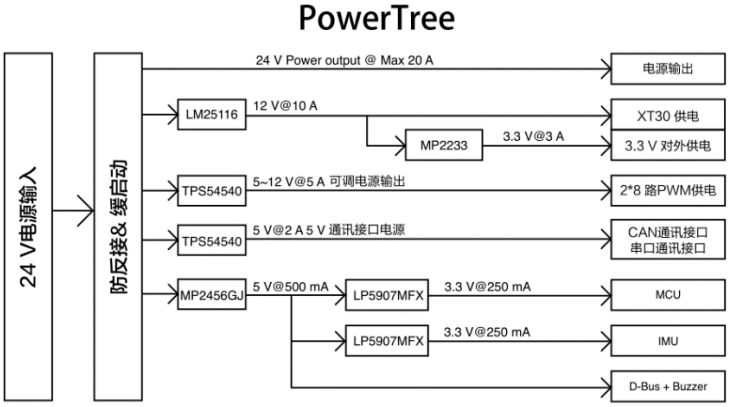

图 4.5 电源分级设计

4.2.4 IMU 模块

IMU 模块由 MPU6500 陀螺仪和 IST8310 地磁传感器组成，采用了独立的 LDO 供电，供电电路图（4.6）（4.7）如下

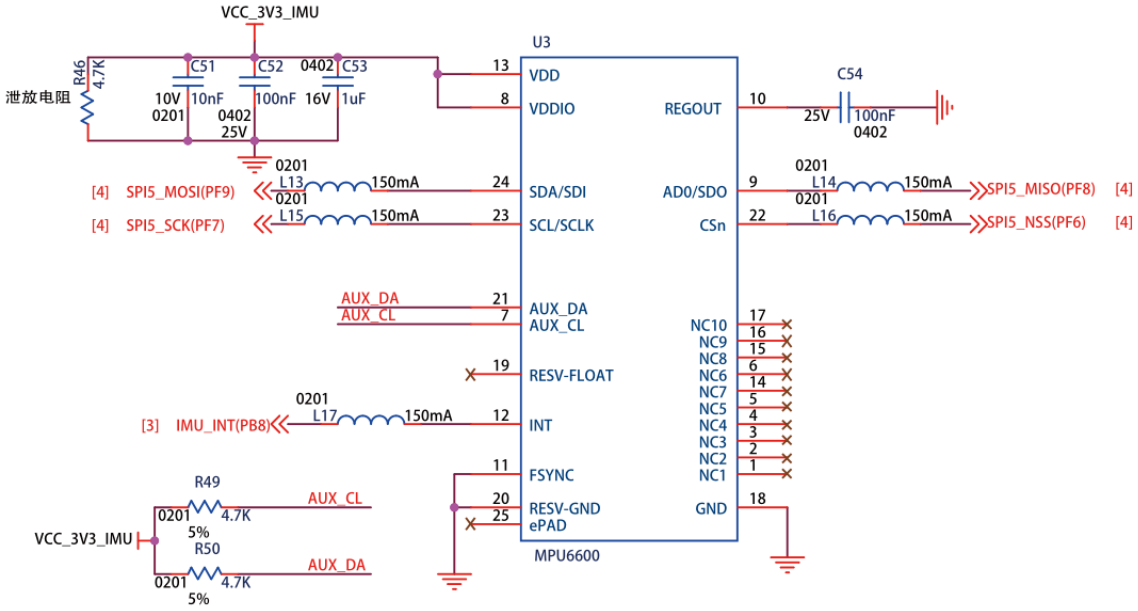

图 4.6 MPU6500

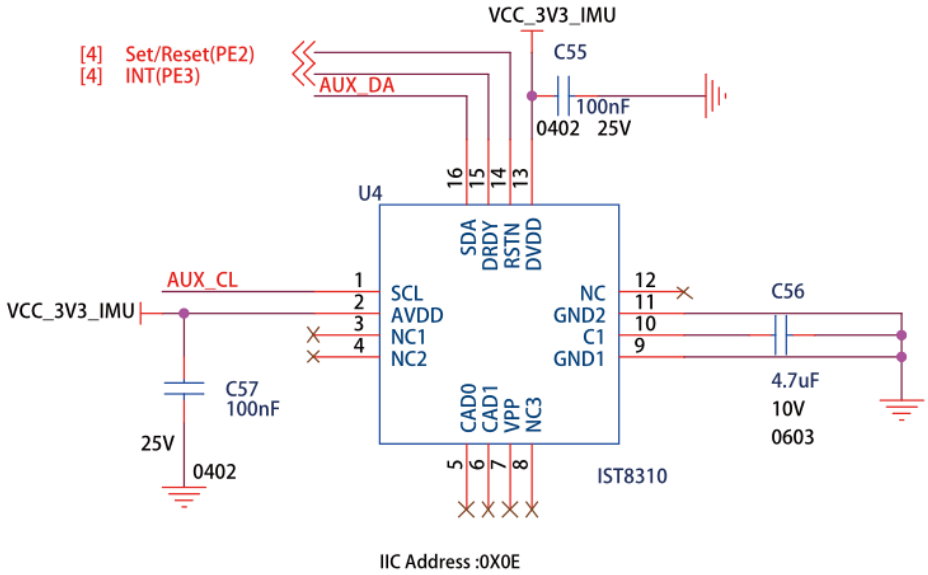

图 4.7 IST8310

4.3 嵌入式系统动力电路设计

4.3.1 无刷电机驱动电路

动力部分是机器人的硬件部分的基础也是最重要的部分之一，动力电源的好坏直接影响了机器人的性能，比赛中要求的电路最高电压不得超过 24V 所以我们选用了 22.2V 的航模锂电池，由于使用了 8 个电机，每个电机持续电流有 5-10A,所以在主电路里面我们选用了，28AWG 的多股铜芯线，可以承受高达 100A 的持续电流，不至于烧毁，在每个电机的链接电路里选哟给 20AWG 的多股铜芯线就可以达到电流要求，安全经济。无刷电机动力电路电路框图如下：

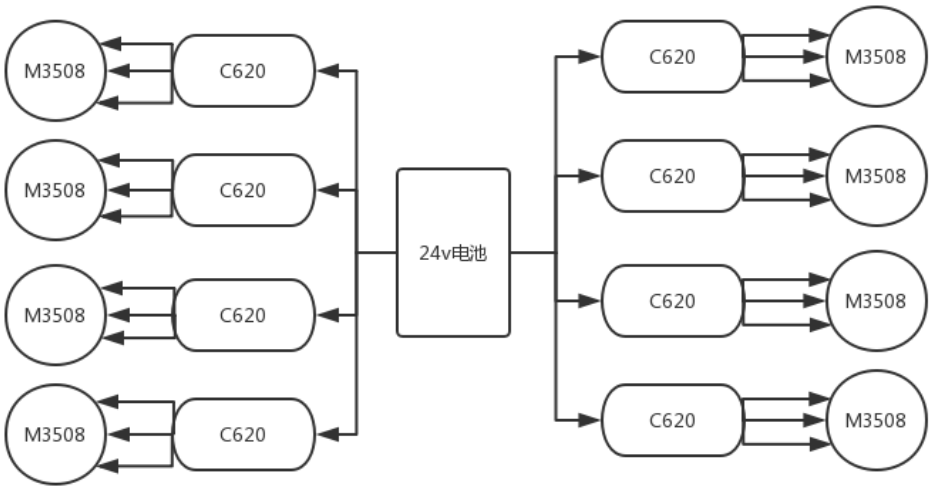

图 4.8 动力电路连接图

4.3.2 舵机驱动电路

在开发机器人的过程中由于要使用到舵机，一般舵机的电压是 5V，但是普通的单片机 IO 口并不具备足以驱动舵机的供电电流能力，因此需要单独的硬件电路给舵机供电。

在开发板上端口直接配置到了 5v 供电芯片上，用户还可以根据自己的实际需要，通过拨码开关调整电源电压。如图（4.9）所示

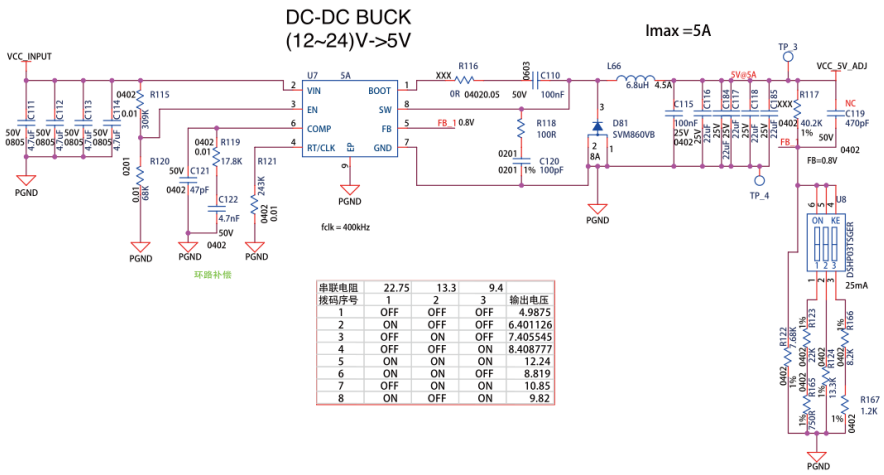

图 4.9 舵机 PWM 接口

4.4 电压检测电路设计

航模锂电池放电能力强，机器人的用电能力大，稍不注意锂电池就可能过放，造成永久性的失效，同理电池电压过低，驱动能力不足，对机器人性能的影响非常的大，我们的主电路是 24V，而 STM32 单片机的 ADC 电压范围为 0-3.3v，因此我们需要一个电压转换电路来实现将 24V 电压线性转换成 0-3.3V 的电压，从而在程序里面实时监测主电路电压，估算剩余电量，低电压报警。

电路如图（4.10）所示

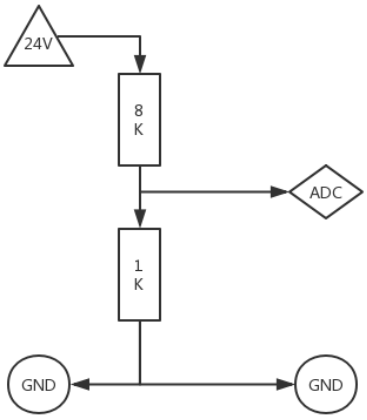

图 4.10 电压检测电路原理图

4.5 章节小结

本章节介绍了控制系统的硬件部分，主要涉及到开发板最小系统和外设的供电，机器人工程中，硬件是一台机器的基础，是机器人性能是否优秀的重要保障。

五 控制系统软件设计与实现

控制软件主要是基于 stm32 平台，KEIL MDK IDE，全程使用了 stm32cubemx 配置外设生成代码，使用最新的 HAL 库，遵循 ST 官方的代码规范，整体框架基于 FREERTOS，核心功能是正弦函数发生器，步态振荡器，以及导航中的方向纠偏，姿态矫正。

5.1 嵌入式实时操作系统

5.1.1 设计分析

机器人工程是一个庞大的工程，需要处理非常多的信息，并且要保持高度的实时性，传统的程序写法实现起来逻辑复杂，容错率低，因此选择使用 FREERTOS，是一款免费的嵌入式实时操作系统，经过多版本的迭代，现已经是众多嵌入式实时炒作系统中比较成熟稳定的一个，STM32CubeMx 中支持了添加 FREERTOS 功能，固使用 CubeMx 生成操作系统的代码，使用嵌入式实时操作系统可以极大的简化编程人员的负担，可以在逻辑中简单的将每个任务认为是同时运行的进程。

5.1.2 FREERTOS 介绍及运行机制概述

FREERTOS 是一种实时操作系统，选择 FREERTOS 的原因

1. FreeRTOS 是一个没有任务数限制的、可裁剪的、可剥夺型的多任务内核。
2. FreeRTOS 提供实时操作系统所需的所有功能，包括资源管理，同步，任务集成等。
3. FreeRTOS 是用 C 语言和汇编语言编写的，大部分用 C 语言编写，与处理器密切相关的代码只用汇编语言编写。 非常强大！
4. 最重要的是，它非常适合实时操作系统，嵌入式系统开发人员和新爱好者。
5. 相比于其他 RTOS，FREERTOS 有很多优势，热门、开源、免费，是 st 公司钦定的操作系统。

简而言之，使用 RTOS 的实时应用程序可认为是一系列独立任务的集合。每个任务在本身的环境中运转，不依赖于系统中的其它任务或 RTOS 调度器。在任何时刻，只有一个任务得到运行，RTOS 调度器决定运行哪个任务。调度器会不断的启动、停止每一个任务，宏观看上去就像整个应用程序都在执行。作为任务，不需要对调度器的活动有所了解，在任务切入切出时保存上下文环境（寄存器值、堆栈内容）是调度器主要的职责。为了实现这点，每个任务都需要有自己的堆栈。当它的任务切出时，它的运行环境会被保存在该任务的堆栈中，可以从堆栈中正确的恢复上次的运行内容和环境。

5.1.3 任务功能介绍

按照任务调度优先级的顺序，FREERTOS 的 IRQ 任务和高优先级任务必须要设置为阻塞式（在这里我使用了延时函数），只有这样，高优先级任务才会释放 CPU 的占有权，从而低优先级任务可以得到机会得到执行。

因此高优先级的任务必须要有延时，延时了 200ms 给低优先级的任务提供运行时间。

高优先级任务，检测、调试、遥控、山外上位机、逻辑流控制、测试，这些任务的特点，全部都拥有 200ms 以上的延时，为阻塞式，不需要过快的速度，处理任务量比较小。

低优先级任务如姿态控制、电机控制任务，将以很高频率运行。

5.1.4 任务调度关系

在该项目中我创建了 10 个任务，分别是开始任务、电机控制、姿态控制、导航、检测、调试、遥控、上位机通信、比赛流程控制、测试任务，他们的逻辑关系如下图（5.1）

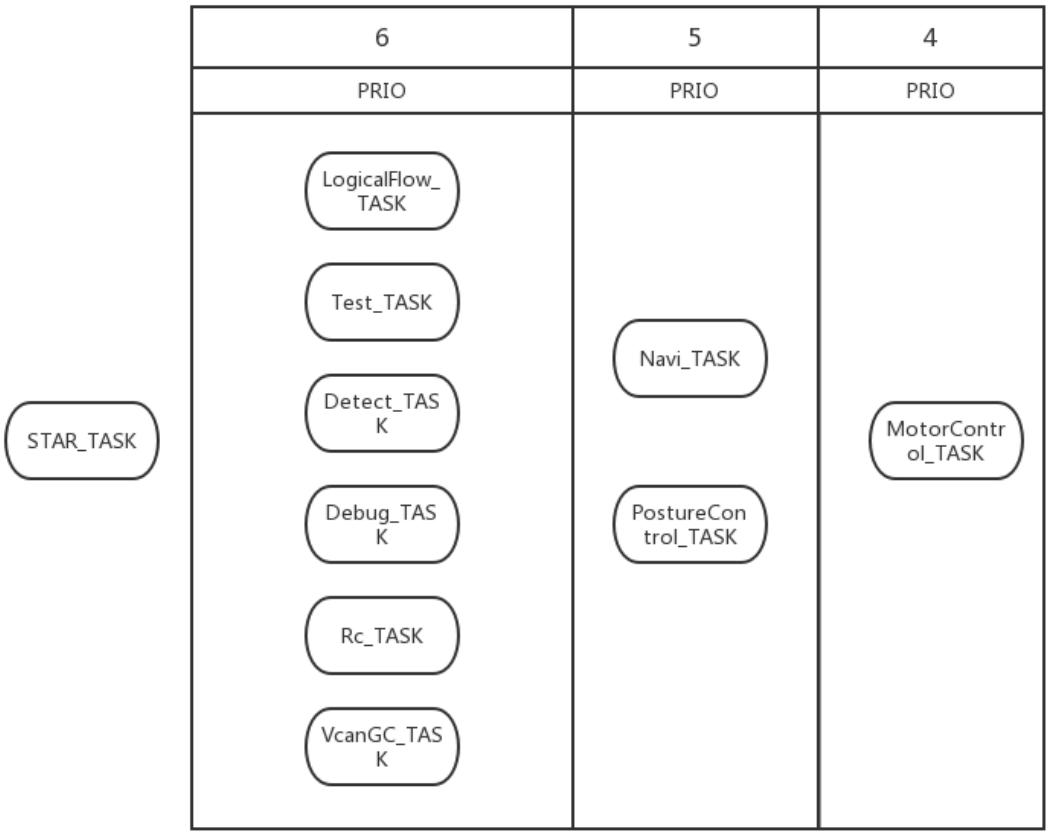

图 5.1 任务框图

5.2 正弦轨迹振荡器

5.2.1 轨迹振荡器基本概述

四足机器人运动最基本，最底层的事情是精确控制每一条腿足端的轨迹，动物在行走的时候，流畅的运动归功于流畅的足端轨迹，动物在行走的时候足端的轨迹分为两部分，分别是，支撑相，摆动相，其中支撑相是足端与地面接触时的轨迹，理论上是一条直线，摆动相是抬腿过程中足端的轨迹，我选择了使用正弦轨迹来模拟，在整段轨迹中，速度有连续变化的需求，不能有突变，而且还应该具不同的参数生成不同的轨迹的特性，参数传递功能。

5.2.2 轨迹振荡器的比较和选用

我设计了两种轨迹发生器，分别是摆线轨迹发生器，和正弦轨迹发生器。对比之后，摆线轨迹生成器的曲线拥有更好的平滑性，而在正弦轨迹中，sin 函数比较简单，大大的降低了程序的复杂程度，取其两者精华，修改了很多地方增加了一些项目特殊需要的接口，修改成了改进版的正弦轨迹振荡器。

5.2.3 正弦轨迹振荡器的程序设计

```
/**
* NAME: SinTrajectory (float t,GaitParams params, float gaitOffset)
* FUNCTION :  正弦轨迹生成器
*/
void SinTrajectory (float t,GaitParams params, float gaitOffset) {
    static float p = 0;
    static float prev_t = 0;

    float stanceHeight = params.stance_height;
    float downAMP = params.down_amp;
    float upAMP = params.up_amp;
    float flightPercent = params.flight_percent;
    float stepLength = params.step_length;
    float FREQ = params.freq;
```

```
        p += FREQ * (t - prev_t);
        prev_t = t;

        float gp = fmod((p+gaitOffset),1.0);
        if (gp <= flightPercent) {
            x = (gp/flightPercent)*stepLength - stepLength/2.0;
            y = -upAMP*sin(PI*gp/flightPercent) + stanceHeight;
        }
        else {
            float percentBack = (gp-flightPercent)/(1.0-flightPercent);
            x = -percentBack*stepLength + stepLength/2.0;
            y = downAMP*sin(PI*percentBack) + stanceHeight;
        }

    }
```

5.3 周期步态算法

5.3.1 周期步态算法

步态周期算法是在一条腿轨迹的基础上，进行足间协调控制，才能生成相位，组件协调控制通过建立 4 个腿部振荡器之间的相位耦合关系来实现，四足机器人的典型步态有了四种，walk(行走)，trot(对角小跑)，pace(同侧遛步)，gallop(跳跑)，walk 步态的特点是四足交替的起落，相位差为 0.25，负载因子为 0.75；trot 步态特点是对角足成对起落；pace 步态特点是同侧足成对起落；gallop 步态特点是前后足成对起落。后三种步态的相位差和负载因子均为 0.5。图 3-17 所示为四种步态对应的四条腿之间的相对相位关系。

5.3.2 周期步态控制实现

在程序中，stm32hal 库提供了 HAL_GetTick()函数，可以直接获取系统时间，单位为 ms。

在程序中直接使用了该库函数，作为基准的时钟脉冲，经过运算之后，输入到步态控制器里面，决定了那一条腿在何时开始动，通过频率控制了一个周期的时间长短通过占空比，控制了摆动相和支撑相的时间长短。

5.3.3 周期步态程序设计

```
/**
* NAME: void gait(  GaitParams params,float leg0_offset, float leg1_offset,float leg2_offset, float leg3_offset)
* FUNCTION : 产生时间脉冲 设定每个腿的参数 调整腿的运行方向 进行补偿
*/
void gait(    GaitParams params,LegGain gains,
                  float leg0_offset, float leg1_offset,float leg2_offset, float leg3_offset,
                  float leg0_direction, float leg1_direction,float leg2_direction, float leg3_direction) {

      if (!IsValidGaitParams(params) || !IsValidLegGain(gains)) {
            return;
      }

      float t = HAL_GetTick()/1000.0-now_time/1000.0;

      //printf("\r\n t=%f",t);

      // const float leg0_direction = 1.0;
      CoupledMoveLeg( t, params, leg0_offset, leg0_direction, 0);

      // const float leg1_direction = 1.0;
      CoupledMoveLeg( t, params, leg1_offset, leg1_direction, 1);

      // const float leg2_direction = 1.0;
      CoupledMoveLeg( t, params, leg2_offset, leg2_direction, 2);

      //    const float leg3_direction = 1.0;
      CoupledMoveLeg( t, params, leg3_offset, leg3_direction, 3);

      //改变 PD
       ChangeTheGainsOfPD(gains);
}
```

5.4 外设通信

5.4.1 mti30 陀螺仪

四足运行过程中，机器人接收 mti30 数据的通信方式为串口通信，由于 mti30 发送的是 RS232 的电平信号，固外置了 MAX2323 电平转换芯片，转换成 TTL 电平型号，接入 STM32。通讯的波特率为 115200 ，mti30 的数据是可以用上位机更改的，设置成欧拉角模式，虽然有角度突变这个问题，但相比旋转矩阵，四元素，减轻了控制器的运算量，接收到的数据有，ROLL（横滚角）,PITCH（俯仰角）,YAW（偏航角）

欧拉角输出模式下输出定义，如图 5.2 所示：

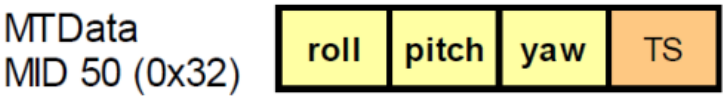

图 5.2 欧拉角输出模式定义

系统以数据包的形式发送数据，每个数据包包含 28 个字节，其中包头 2 个字节，包尾 2 个字节。 中间 24 字节为数据，以 4 个字节为单位（LSB），分别为三轴角度值，2 个坐标值，和一个角速度值。 这些数据的存储方式为单精度浮点数，在 C 语言里为 float 型数据。

数据接收流程如图 5.3 所示：

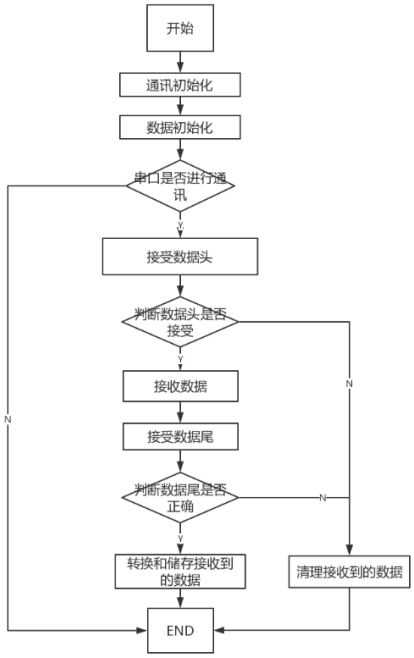

图 5.3 数据接收流程图

具体通讯流程如下：

（1）初始化 UART

（2）创建共用体。

（3）使用空闲终端进行处理，如果是遇到了一帧数据，则连续收，存入缓冲区域，知道一阵数据完毕，实现了不定长的数据收发。

（4）在串口空闲终端里面进行处理，一帧数据结束后，进入中断，判断字头是否正确，如果不正确则跳出，不进行数据跟新。

（5）如果数据正确则存入共用体。

5.4.2 C620 电调 CAN 通信

C620 电调是大疆 robomaster 出品的一款强大的无数点击调速器，里卖弄及成立，我们使用了大疆的 can 总线通信协议 c620 电调反馈回来的数据格式如图 5.4 所示

标识符：0x200 + 电调 ID
（如：ID 为 1，该标识符为 0x201）
帧类型：标准帧
帧格式：DATA
DLC：8 字节

数据域	内容
DATA[0]	转子机械角度高 8 位
DATA[1]	转子机械角度低 8 位
DATA[2]	转子转速高 8 位
DATA[3]	转子转速低 8 位
DATA[4]	实际转矩电流高 8 位
DATA[5]	实际转矩电流低 8 位
DATA[6]	电机温度
DATA[7]	Null

图 5.4 c620 反馈数据格式

我们通过发送命令控制电机，通过 can 总线发送的报文格式通信格式如图 5.5 所示

标识符：0x200　　帧格式：DATA
帧类型：标准帧　　DLC：8 字节

数据域	内容	电调 ID
DATA[0]	控制电流值高 8 位	1
DATA[1]	控制电流值低 8 位	
DATA[2]	控制电流值高 8 位	2
DATA[3]	控制电流值低 8 位	
DATA[4]	控制电流值高 8 位	3
DATA[5]	控制电流值低 8 位	
DATA[6]	控制电流值高 8 位	4
DATA[7]	控制电流值低 8 位	

图 5.5 c620 发送数据格式

5.4.3 航模遥控 PPM

PPM 信号是航模中通常用的一种信号，PPM 信号就是多路的 PWM 被合成到了一路的波上面发送机发送给了接收机之后，接收机会再次将 PPM 波分别读取成分开的 PWM 读取相应的高电平时间，帧与帧之间的时间间隔大于波与波之间的间隔。

因此我们只需要使用定时器的 pwm 捕获功能，捕获多路 pwm，储存起来解析即可。

PPM 协议格式如图 5.6 所示

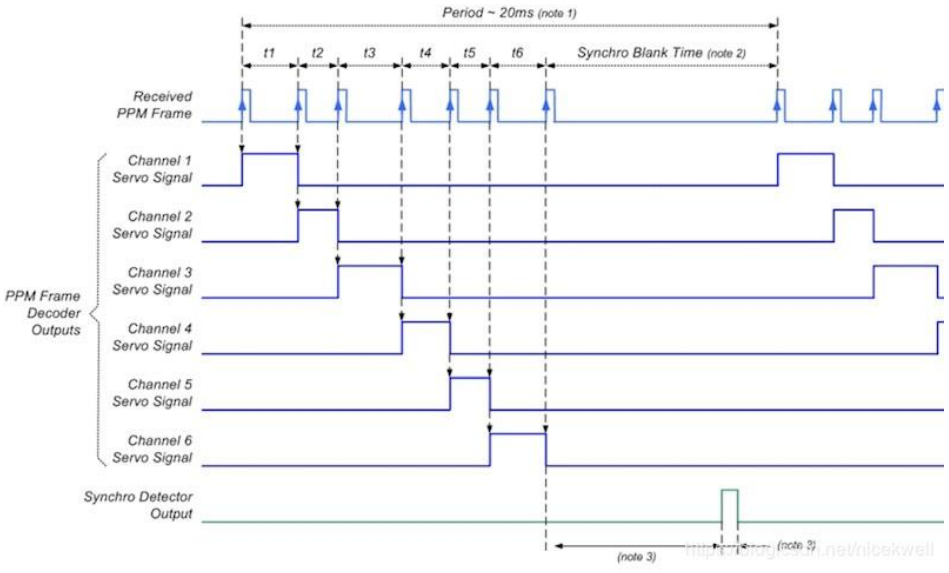

图 5.6ppm 协议

5.5 计算机视觉

视觉设备选择很多，有专业的 OPENCV、小巧的工业相机、便携且功能强大的 OPENM。这三种设备各有各的优点，也各有各的缺点。

5.5.1OPENMV 选型

OPENCV，OPENCV 是一种专业的视觉学习和应用设备，因其强大的功能，在各领域被广泛应用。OPENCV 不仅可以通过算法实现特征物体的识别，还可以通过相机上自带的测距功能实现特定物体的距离测算，从而实现机器视觉的进化，以此来实现更多复杂的功能。例如波士顿动力公司研发的四足机器人相互协助开关门，其摄像头就用了测距功能，以此来实现对目标位置、目标物体的距离，甚至大小的检测。

然而 OPENMV 这种专业的设备，虽然有一部分例程，但是要想自己能够应用到自己特殊的检测，就必须自己编写底层分析代码与算法，所以对新手有很大的难度，而且不适用于短期的开发。另外，OPENCV 摄像头体型较大，而且需要单独的 PC 进行计算，不适用于轻量化的四足信使机器人。

工业相机，工业相机有较高的像素，对于色彩的识别有更高的准确度，而且部分工业相机还具有自动调焦距，自动曝光等功能，可以实现较优秀的环境视觉反馈。但是工业相机（如图 5.7）只有相机一个部件，获取的信息直接通过数据线返回到处理器，所以这一部分和 OPENCV 有相似之处，都必须自己编写相关代码和算法，对于新手不是很友好，而且开发周期较长。除此之外，OPENCV 和工业相机都有较大的运算量，OPENCV 和工业相机都可以通过微型工业 PC 运行处理，工业相机还可以将数据发送给 STM32 进行处理，四足信使机器人已经有一个 STM32 开发板作为数据处理中心，如果再将工业相机的数据发送给该开发板处理，那可能会导致整体运算速度大幅下降，从而影响整体的效果。所以从效率角度来讲，可以再用一个 STM32 开发板单独做运算，然后再将处理结果通过串口发送给主控板，但是如此一来，增加了电路的复杂，整体工作量也增加不少，一旦出现问题，那在短时间很难定位问题所在，所以工业相机作为第二选择列入备用方案。

图 5.7 工业相机

OPENMV，OPENMV 是一种近年来慢慢兴起的视觉处理设备，其编程语言为 Python，相比于 OPENCV 的开发语言 C++以及工业相机的开发语言 C 语言，Python 在某些方面更为简洁，实现相同的目的，程序更加简短，而且易懂，所以在开发环境方面，OPENNMV 更胜一筹。除软件层面之外，OPENMV 的硬件方面也有 OPENCV 和工业相机无法与之比拟的优势，OPENMV 集成了控制芯片和摄像头于一块充电器头大小的电路板上，官方还提供了许多的扩展版，如舵机扩展板，LCD 显示屏扩展版，无线传输扩展版等，使得小巧的 OPENMV 开发板功能更加强大。官方编写的底层代码几乎满足了所有一般的视觉识别需求，比如颜色识别、形状识别、距离测算、人脸识别等，直接调用相关函数即可。由于 OPENMV 是闭源项目，所以相比于 OPENCV 少了许多灵活性，但是利用官方提供的例程代码，可以很快掌握 OPENMV 的使用，缩短了开发周期。但是 OPENMV 自带的摄像头像素较低，对于一些精确的边界或者色域值相近的颜色，OPENMV 就体现出了他的不足，很大情况会出现未检测到或者无中生有的情况。由于 OOPENMV 硬件对于环境的要求非常高，所以 OPENMV 只能在环境因素变化不大的情况下才能保证较高的识别准确率，这一点是远不如 OPENCV 和工业相机的。

综上所述，OPENMV（如图 5.8 所示）体积小，功能强大，运行稳定，是四足信使机器人视觉识别的最优选择之一。

图 5.8 openmv

5.5.2 OPENMV 调试

OPENMV 不仅可以用作颜色识别，还可以调用线性回归函数实现循迹。通过设定循迹的线的阈值，摄像头捕捉画面并将其返回给主控芯片，之后将图片转化为灰度图像，通过线性回归，将阈值内的像素回归为一条直线，然后返回该直线的斜率、与中心偏移距离，通过串口将这些信息发送给四足信使机器人主控芯片，就可以实现自动循迹，其次还可以通过设置权重来调整线性回归的准确度。然而四足信使机器人的跳跃对线性回

归有很大的影响，而且很大可能所寻的线会超出摄像头视野范围，所以循迹只能作为备用方案。

相比于巡线，颜色识别在物理位置方面的要求要小许多，所以采用颜色识别来完成机器人的转向或者跳跃等一系列动作。OPENMV 的调试软件为ＯＰＥＮＭＶ　ＩＤＥ，通过摄像头捕获目标图像，利用软件内置的阈值编辑器（如图 5.9），滑动滑块使得目标颜色调整为白色，而非目标颜色为黑色。为了提高颜色阈值调节的准确度，打印一张具有很多不同颜色色块的图作为参考，即可缩小目标颜色的阈值，从而提高识别准确度，保证阈值相近时不会出现误判的情况。

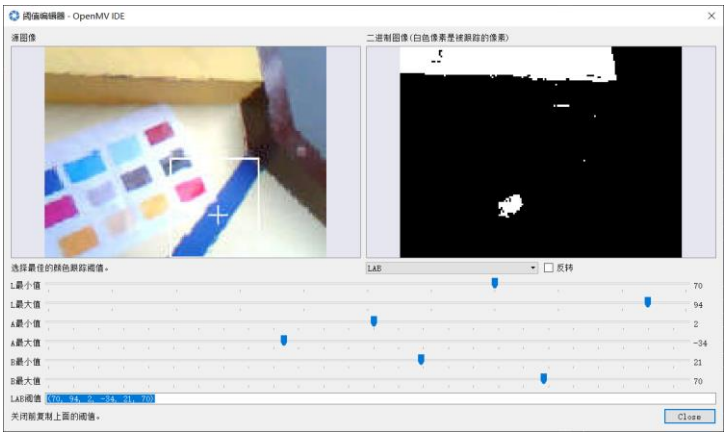

图 5.9 阈值调整界面

四足信使机器人运动速度较快，导致摄像头捕捉的画面变化也非常快，难免会出现误判的情况，特别是场地的黄色和淡黄色，像素低、运动较快都会导致部分阈值的接近，就会出现无中生有的情况，错误的判断必将导致四足信使机器人动作出错。要想彻底解决此类情况几乎不可能，但是能够通过滤波等软件层面的操作大幅降低影响。

通过设置目标色块最小的像素以及面积就可以消除因为摄像头不稳定引起的噪点。但是如果是因为周围光线变化引起颜色错误识别，那 OPENMV 就无能为力了，因为 OPENMV 自身的硬件就已经决定了其性能上限，这一切不稳定都源于 OPENMV 对环境依赖程度太高，所以 OPENMV 也只能在环境因素比较稳定的情况下才能发挥较好的作用。

5.5.3 OPENMV 与 STM32 通信

单片机之间的通信相对来说比较简单，只要以下两个方面设置正确即可，硬件方面，将两个开发板的 GND 用杜邦线连接，RXD 和 TXD 分别反接。

软件方面，OPENMV 将需要发送的信息用ＳＴＲＵＣＴ．ＰＡＣＫ打包 转换为二进制数据通过串口发送。接收方先通过串口中断将接收到的信息储存在缓冲区，再对数据进行转换、校验、处理。首先对缓冲区内数据进行转换，浮点型数据是以四个字节发送过来

的，那接收方就将这四个字节放在一个结构体，之后就可以自动转换为原浮点型数据。其次是校验，校验数据包的开头和结尾，对于数据准确度要求高的，还可以将发送的数据求和，发送方和接收方分别计算一次和，若结果一样就说明没有错误，不一样就舍弃数据，进入下一次中断重新接收。最后是处理，将所得的数据进行处理以作为系统的调节参数。

5.6 本章小结

本章是论文最重要的一章，在本章中主要对四足机器人的控制系统的软件进行了整体的介绍并对其中的核心内容进行了详细介绍。

六 经济性与可持续发展性分析

在当今资源有限的社会中，可持续发展战略是当前的全球发展趋势。机器人是未来社会科学技术发展趋势的发展趋势。环境友好型机器人主要从以下观点考虑：

（1）清洁能源，能源是一台机器的根本。四足机器所用的能源为直流电。电能主要用于给控制板、电机、驱动器等电动的部件供能。机器上的锂电池组中储存的电能可以进行多次循环。电池使用的寿命较长。在使用过程中对环境污染较小，属于清洁能源，且成本较低。

（3）能源高效使用，采用驱动马达直接连接的结构，并且在驱动器和电动机之间没有链连接，减少了摩擦力所引起的能量损失，传动链越短，机构的效率越高。因此，电机的整体效率更高，该机器更环保。

（4）节能的控制系统，在四足机器人的运动轨迹设计时，机器的运动轨迹与运动方式经过多次的优化，平滑的运动。节省了机器的能源，节约大量的能源。

综上所诉，四足机器人是一台符合可持续发展战略的机器人。四足机器人具有低成本、使用清洁能源，高效的利用能源、在运行过程中无污染等特征。四足机器人具有很好的绿色环保性。

七 总结与展望

7.1 总结

本文主从机械结构，硬件系统、软件系统这三个方面对自动抛投机器人进行设计。其中软件系统设计是本文的核心。

机械结构设计中，首先是机架的设计，在做了早期的三维建模之后，通过分析，进行了铝板的选型和加工，底盘设计中使用了全铝合金 5mm 的设计，横梁 8mm 的铝合金板，电机连接件 5mm 的铝合金连接板，腿部也使用了 5mm 的铝合金板，该结构可以在低速下保证运行的稳定。

在控制系统的硬件设计中，选取了 stm32f4 系列的芯片，保证运算能力，在最小系统的设计过程中，参考了市场上许多常用开发板的电路设计，选取了 max2323 做为电平转换模块，选取了 MPU6500 陀螺仪和 IST8310 地磁传感器组成 imu 模块，采用了独立的 LDO 供电，在主板电源供电部分，并联了 28 TVS 管，防止瞬间高压烧坏主控板，通过 PMOS Q 1 和 NMOS Q2 组成缓启动与防反接电路，防止电源接触孙瞬间打火造成的接头损坏，齐纳二极管可以在输入电压超过其击穿电压（30+-2V）时，三极管 Q3 导通，PMOS Q1 断开，保护主控板。

在软件设计中，考虑到大工程多任务的需求，引入了免费开源的嵌入式实时操作系统 FREERTOS。设计了正弦轨迹振荡器，实现了并联结构的腿部轨迹控制，在此基础上，做了足间协调控制，实现了不同步态，通过步态周期控制器，实现了不同的参数控制不同的频率，压腿高度，抬腿高度，站立高度，的变化。通过电机 PID 参数的优化，实现了四足机器人的跳跃。

7.2 展望

在毕业设计总由于时间和能力的限制。我在设计中有很多地方存在着缺陷。

结构上，由于使用的是电机直连的方式，在有巨大冲击的时候对电机的损伤非常大，已经造成好几个电机减速器损坏，应当使用一种联轴器和电机轴相连，同时设计编码器的安装位置， 安装额外的绝对式编码器以确定腿的绝对位置。

在程序设计中，设计功能的时候由于经验不足，在一开始没有一个很明确的框架，导致程序的扩展性能受限，增添修改的时候任务量十分巨大。

综合上诉两点，结构上需要很大的修改才能达到正常强度的要求，在设计中应该先进行计算仿真，然后再开始做实物。在设计程序的时候，应该先有一个明确的任务规划，再来开始着手写代码，全程要有一个大局观念。

致谢

在本次毕业设计中，我学习到了很多的东西，其中很大一部分都是来源于国内外开源的项目，四足机器人的研究现在方兴未艾，大大小小的研究团队层出不穷，新的技术在不断的涌现，这是一个充满了活力的行业，但是，目前四足机器人大多停留在实验室的阶段，往往是一个展示用的产品，或者在某些军用，和特殊领域有运用，但是这个行业，在等待一个公司，像当年大疆一统无人机行业一样，将四足机器人的价值真正的商用化，让高性能智能的机器人能够走进平常人的生活，改善人类的生活质量。这可能，才是我们这些人的目标。

感谢两位指导老师对我的帮助。感谢机器人实验室所有人的努力和付出，通过这一次比赛，我们在技术上有了很大进步，学习到很多的东西，在不断追求卓越的过程中，突破了自我。

该项目以后也会是我的主要研究对象，相关的代码已经在 github 上开源，欢迎访问、讨论、交流。https://github.com/Young144/MR2。

参考文献

［1］ 辛全彬. 小型电动直驱仿生四足机器人系统设计[D].大连理工大学,2018.

［2］ Massachusetts Institute of Technology; Mini cheetah is the first four-legged robot to do a backflip[J]. NewsRx Health & Science,2019.

［3］ 梁美彦.基于 STM32 的四足仿生机器人设计与实验研究[J].测试技术学报,2019,33(01):34-42.

［4］ 魏兵，喻全余，孙末． 机械原理［M］． 武汉：华中科技大学出版社，2011．

［5］ 王维，王建晓． 机械设计［M］． 武汉：华中科技大学出版社，2011．

［6］ 朱雅乔,陈国松,王姣姣,范广宏.四足机器人仿生关节的研究现状综述[J].机械传动,2019,43(01):159-164.

［7］ 陈榕婷,叶泳仪,杨柳娟,李金林,郑誉煌.基于 Solidworks 的四足机器人的步态分析[J].科学技术创新,2019(07):70-71.

［8］ 李岩.四足步行机器人结构设计分析[J].山东工业技术,2019(10):138.

［9］ 谭浩强． C 程序设计［M］． 北京：清华大学出版社，2010．

［10］赵大兴，高成慧，谢跃进．现代工程图学教程［M］． 武汉：湖北科学技术出版社， 2009．

［11］陈花玲． 机械工程测试技术［M］． 北京：机械工业出版社，2008．

［12］杨园园. 基于多层 CPG 神经网络的四足机器人控制方法研究[D].哈尔滨工业大学,2018.

［13］Peter Corke（彼得 科克）著，刘荣 译.机器人学、机器视觉与控制——MATLAB 算法基础［M］． 北京： 电子工业出版社，2016．

［14］金炳辰. 四足仿生机器人腿部构型设计及其缓冲运动控制的研究[D].哈尔滨工业大学,2018.

［15］王鸿舸. 小型四足机器人的奔跑和跳跃控制[D].山东大学,2018.

［16］胡胜心. 液压驱动四足机器人的系统控制仿真及初步验证[D].南京航空航天大学,2015.

［17］R C Liu,G Y Ma,Y Chen,S Han,J Gao. Four-legged robot design and gait planning[J]. Journal of Physics: Conference Series,2018,1087(6).

［18］严蔚敏，吴伟民． 数据结构：C 语言版［M］． 北京：清华大学出版社，2007．

［19］黄智伟，王兵，朱卫华． STM32F 32 位 ARM 微控制器应用设计与实践（第 2 版） ［M］． 北京：北京航空航天大学出版社，20014．

附录

一些重要的代码

```
/*********************************************************************************
*****************
    *@ 任务名称：void PostureControl_task(void *pvParameters)
    *@ 功能：姿态控制
    *@ 备注：选择状态

*********************************************************************************
*************/
void PostureControl_task(void *pvParameters)
{

//              step_len_initial =
state_detached_params[TROT].detached_params_0.step_length;
//         step_high_initial =
state_detached_params[CLIMBING].detached_params_0.stance_height;
//
      for(;;)
      {
//            _Pitch_rev=_Pitch_initial-imuinfo.ActVal[1];
//            _Roll_rev=_Roll_initial-imuinfo.ActVal[2];


            gait_params = state_gait_params[state];
            detached_params = state_detached_params[state];


            //gait_gains =state_gait_gains[state] ;
            switch(state) {

            case STOP:          //停止
                  x=0;
                  //y = 17.3205081;
                  y = 21;
                  CartesianToTheta(1.0);
                  LegGain gains = {10, 0.00, 5, 0.00};
                  //pid_spd_out_limit=400;
                  CommandAllLegs(gains);
                  break;

            case REALSE:      // 释放 什么都不做
```

```
            vTaskDelay(500);
            break;

        case CLIMBING:      //爬坡
            gait_detached(detached_params, 0.0, 0.5, 0.5, 0.0, -1.0, -1.0, -1.0, -1.0);
            break;

        case WALK:      //前进
            gait_detached(detached_params, 0.25, 0.75, 0.0, 0.5, 1.0, 1.0, 1.0, 1.0);
            break;

        case WALK_BACK:      //后退
            gait(gait_params, gait_gains, 0.0, 0.5, 0.5, 0.0, -1.0, -1.0, -1.0, -1.0);
            break;

        case BOUND:     //跳跑
            gait_detached(detached_params, 0.0, 0.0, 0.5, 0.5, 1.0, 1.0, 1.0, 1.0);
            break;

        case GALLOP:    //袭步
            gait_detached(detached_params,0.9, 0.7, 0.2, 0.0, 1.0, 1.0, 1.0, 1.0);
            break;

        case ROTAT_LEFT:        //原地左转  YAW+
            //gait_detached(detached_params, 0.0, 0.5, 0.5, 0.0, -1.0, 1.0, -1.0, 1.0);
            gait(gait_params, gait_gains, 0.0, 0.5, 0.5, 0.00, -1.0, 1.0, -1.0, 1.0); //未分离
的步态
        //          if(imuinfo.ActVal[0]>=_angle_initial+_rotate_angle)
        //              state= STOP;

            break;
        case ROTAT_RIGHT:       //原地右转  YAW-
            // gait_detached(detached_params, 0.0, 0.5, 0.5, 0.0, 1.0, -1.0, 1.0, -1.0);
            gait(gait_params, gait_gains, 0.0, 0.5, 0.5, 0.00, 1.0, -1.0, 1.0, -1.0); //未分离
的步态
        //          if(imuinfo.ActVal[0]<=_angle_initial-_rotate_angle)
        //              state= STOP;

            break;
        case JUMP:      //跳跃
            //pid_spd_out_limit=6720;
            ExecuteJump();
            break;
```

```
                case START:         //初始化
                    StartPosToMiddlePos();
                    break;

                case END:       //结束
                    MiddlePosToEndPos();
                    break;

                case TROT:          //双拍步态 对角小跑

                    if(rc_ctrl_flag == 0)
                        gait_detached(detached_params,    0.0, 0.5, 0.5, 0.0, 1.0, 1.0, 1.0, 1.0);
                    //pid_spd_out_limit=6720;
                    else if(rc_ctrl_flag == 1)
                        gait_detached(RcDetachedParam,    0.0, 0.5, 0.5, 0.0, 1.0, 1.0, 1.0, 1.0);

                    break;
                }
                vTaskDelay(10);
            }
        }

        /**
        * NAME: void gait_detached
        * FUNCTION : 四腿分离的腿部增益函数
        */
        void gait_detached(  DetachedParam d_params,
                             float leg0_offset, float leg1_offset,float leg2_offset, float
leg3_offset,
                             float leg0_direction, float leg1_direction,float leg2_direction, float
leg3_direction) {

            float t = HAL_GetTick()/1000.0-now_time/1000.0;

            // const float leg0_direction = 1.0;
            if(_leg_active[0]==YES)
                CoupledMoveLeg( t, d_params.detached_params_0, leg0_offset, leg0_direction,
0);

            // const float leg1_direction = 1.0;
            if(_leg_active[1]==YES)
                CoupledMoveLeg( t, d_params.detached_params_1, leg1_offset, leg1_direction,
1);
```

```
        // const float leg2_direction = 1.0;
        if(_leg_active[2]==YES)
            CoupledMoveLeg( t, d_params.detached_params_2, leg2_offset, leg2_direction,
2);

        //    const float leg3_direction = 1.0;
        if(_leg_active[3]==YES)
            CoupledMoveLeg( t, d_params.detached_params_3, leg3_offset, leg3_direction,
3);


    }

    /**
    * NAME: void gait(  GaitParams params,float leg0_offset, float leg1_offset,float
leg2_offset, float leg3_offset)
    * FUNCTION : 产生时间脉冲  设定每个腿的参数  调整腿的运行方向  进行补偿
    */
    void gait(    GaitParams params,LegGain gains,
                  float leg0_offset, float leg1_offset,float leg2_offset, float leg3_offset,
                  float leg0_direction, float leg1_direction,float leg2_direction, float
leg3_direction) {

    //      if (!IsValidGaitParams(params) || !IsValidLegGain(gains)) {
    //          return;
    //      }

        float t = HAL_GetTick()/1000.0-now_time/1000.0;

        //printf("\r\n t=%f",t);

        // const float leg0_direction = 1.0;
        CoupledMoveLeg( t, params, leg0_offset, leg0_direction, 0);

        // const float leg1_direction = 1.0;
        CoupledMoveLeg( t, params, leg1_offset, leg1_direction, 1);

        // const float leg2_direction = 1.0;
        CoupledMoveLeg( t, params, leg2_offset, leg2_direction, 2);

        //    const float leg3_direction = 1.0;
        CoupledMoveLeg( t, params, leg3_offset, leg3_direction, 3);

        //改变 PD
```

```
        // ChangeTheGainsOfPD(gains);
    }

    /**
    * NAME: void CoupledMoveLeg(float t, GaitParams params,float gait_offset, float
leg_direction, int LegId)
    * FUNCTION : 驱动并联腿 传递参数
    */
    void CoupledMoveLeg(float t, GaitParams params,float gait_offset, float leg_direction, int
LegId)
    {
        SinTrajectory(t, params, gait_offset);        //足端摆线轨迹生成器
        CartesianToTheta(leg_direction);       //笛卡尔坐标转换到伽马坐标


        SetCoupledPosition(LegId);        //发送数据给电机驱动函数
    //    printf("\r\nt=%f x=%f    y=%f    theta1=%f    theta2=%f    legid=%d
he%f",t,x,y,theta1,theta2,LegId,theta1+theta2);
    }

    /**
    * NAME: SinTrajectory (float t,GaitParams params, float gaitOffset)
    * FUNCTION : 正弦轨迹生成器
    */
    void SinTrajectory (float t,GaitParams params, float gaitOffset) {
        static float p = 0;
        static float prev_t = 0;

        float stanceHeight = params.stance_height;
        float downAMP = params.down_amp;
        float upAMP = params.up_amp;
        float flightPercent = params.flight_percent;
        float stepLength = params.step_length;
        float FREQ = params.freq;

        p += FREQ * (t - prev_t);
        prev_t = t;

        float gp = fmod((p+gaitOffset),1.0);
        if (gp <= flightPercent) {
            x = (gp/flightPercent)*stepLength - stepLength/2.0;
            y = -upAMP*sin(PI*gp/flightPercent) + stanceHeight;
        }
        else {
```

```
            float percentBack = (gp-flightPercent)/(1.0-flightPercent);
            x = -percentBack*stepLength + stepLength/2.0;
            y = downAMP*sin(PI*percentBack) + stanceHeight;
        }

    }


    /**
    * NAME: void CartesianToThetaGamma(float leg_direction)
    * FUNCTION : 笛卡尔坐标转换到角度坐标 也就是将 xy 转换成 theta
    */
    void CartesianToTheta(float leg_direction)
    {
        float L=0;
        float N=0;
        double M=0;
        float A1=0;
        float A2=0;

        L=sqrt(          pow(x,2)     +          pow(y,2)      );

        if(L<10) L=10;
        else if(L>30) L=30;

        // vTaskSuspend(MotorControlTask_Handler);

        N=asin(x/L)*180.0/PI;
        M=acos(    (pow(L,2)+pow(L1,2)-pow(L2,2))/(2*L1*L)     )*180.0/PI;
        A1=M-N;

        A2=M+N;

        if(leg_direction==1.0) {
            theta2=(A1-90.0);
            theta1=(A2-90.0);
        } else if(leg_direction==-1.0) {
            theta1=(A1-90.0);
            theta2=(A2-90.0);
        }

        // printf("\r\n x=%f y=%f theta1=%f theta2=%f he=%f L=%f M=%lf
N=%f     SIXI=%f",x,y,theta1,theta2,theta1+theta2,L,M,N,( (pow(L,2)+pow(L1,2)-
pow(L2,2))/(2*L1*L)));
```

```
}

bool climbing_offset_flag=NO;
float _climbing_offset_angle=15;


/**
* NAME: void CommandAllLegs(void)
* FUNCTION : 控制所有电机
*/
void CommandAllLegs(LegGain gains)
{
//      if (!IsValidLegGain(gains)) {
//            return;
//      }

//      //限位保护
//      if((theta1+theta2)>170||(theta1+theta2)<-170||theta1>140||theta1<-
140||theta2>140||theta2<-140)
//            vTaskSuspend(MotorControlTask_Handler);

      ChangeTheGainsOfPD(gains);

      SetCoupledPosition(0);
      SetCoupledPosition(1);
      SetCoupledPosition(2);
      SetCoupledPosition(3);
}

void CommandAllLegs_v(void)
{
      SetCoupledPosition(0);
      SetCoupledPosition(1);
      SetCoupledPosition(2);
      SetCoupledPosition(3);
}

/**
 *
 * 检测步态增益是否正确
 * @param    gains LegGain to check
 * @return          True if valid gains, false if invalid
 */
bool IsValidLegGain( LegGain gains) {
```

```
        // check for unstable gains
        bool bad =    gains.kp_spd < 0 || gains.kd_spd < 0 ||
                        gains.kp_pos < 0 || gains.kd_pos < 0;
        if (bad) {
            printf("Invalid gains: <0");
            vTaskDelay(500);
            return false;
        }
        // check for instability / sensor noise amplification
        bad = bad || gains.kp_spd > 50 || gains.kd_spd > 50 ||
                gains.kp_pos > 50 || gains.kd_pos > 50;
        if (bad) {
            printf("Invalid gains: too high.");
            vTaskDelay(500);
            return false;
        }
        // check for underdamping -> instability
        bad = bad || (gains.kp_spd > 50 && gains.kd_spd < 0.1);
        bad = bad || (gains.kp_pos > 50 && gains.kd_pos < 0.1);
        if (bad) {
            printf("Invalid gains: underdamped");
            vTaskDelay(500);
            return false;
        }
        return true;
    }
    /**
     *
     * 检测步态参数是否正确
     * @param    gains LegGain to check
     * @return            True if valid gains, false if invalid
     */
    bool IsValidGaitParams( GaitParams params) {
        const float maxL = 30.1;
        const float minL = 9.9;

        float stanceHeight = params.stance_height;
        float downAMP = params.down_amp;
        float upAMP = params.up_amp;
        float flightPercent = params.flight_percent;
        float stepLength = params.step_length;
        float FREQ = params.freq;
```

```
        if (stanceHeight + downAMP > maxL || sqrt(pow(stanceHeight, 2) + pow(stepLength /
2.0, 2)) > maxL) {
            printf("Gait overextends leg");
            vTaskDelay(500);
            return false;
        }
        if (stanceHeight - upAMP < minL) {
            printf("Gait underextends leg");
            vTaskDelay(500);
            return false;
        }

        if (flightPercent <= 0 || flightPercent > 1.0) {
            printf("Flight percent is invalid");
            vTaskDelay(500);
            return false;
        }

        if (FREQ < 0) {
            printf("Frequency cannot be negative");
            vTaskDelay(500);
            return false;
        }

        if (FREQ > 10.0) {
            printf("Frequency is too high (>10)");
            vTaskDelay(500);
            return false;
        }

        return true;
    }
    /**
     *改变腿部增益
     *调用了 PID_reset_kpKd 函数
     */
    void ChangeTheGainsOfPD(LegGain gains)
    {

        for (int i = 0; i < 8; i++) {
            pid_reset_kpkd(&pid_pos[i],gains.kp_pos,gains.kd_pos);
            pid_reset_kpkd(&pid_spd[i],gains.kp_spd,gains.kd_spd);

        }
```

```
}


/**
* NAME: void CycloidTrajectory (float t, GaitParams params, float gait_offset)
* FUNCTION :  摆线轨迹生成器
*/
//void CycloidTrajectory (float t, GaitParams params, float gait_offset)
//{
//      float S0 = params.step_length;
//      float H0 = params.stance_height;
//      float T = params.gait_cycle;
//      float Ty = params.swing_cycle;
//

////      printf("\r\n t=%f",t);
//      if(t>=0&&t<=Ty)       //足端摆动相
//      {
//            x=S0/(2*PI)*( (2*PI*t/Ty)-sin((2*PI*t/Ty)) );
//            if(t>=0&&t<=Ty/2)
//            {
//                  y=2*H0*( t/Ty - sin( 4*PI*t/Ty )/(4*PI) );
//            }
//            else if(t>=Ty/2&&t<=Ty)
//            {
//                  y=-2*H0*( t/Ty - sin( 4*PI*t/Ty )/(4*PI) )+2*H0;
//            }
//      }
//      else if(t>=Ty&&t<=T)       //足端支撑相
//      {
//            x=S0 - S0/(2*PI)*( (2*PI*(t-Ty)/(T-Ty))-sin((2*PI*(t-Ty)/(T-Ty))) );
//            y=0;
//      }
//      else
//            vTaskDelay(10);

//      x-=6.0;//将 0 点设置在穿过电机中心的竖直线上
//      y+=10.0*sqrt(3.0);//电机 y 初始位置  表示两条腿为水平的时

//}
```