四足机器人的足式里程计

姓 名: 夏 文 龙
院 校: 哈尔滨工业大学
学 院: 信息科学与工程学院
专 业: 自动化
邮 箱: 2473833028@qq.com

2024 年 2 月 20 日

摘 要

四足机器人在探索、救援等领域应用场景广泛。四足机器人的运动算法和导航规划等上层任务需要机器人对自身的状态有准确的估计。本文将借助四足机器人的各关节编码器、惯性测量单元 (IMU)、以及足部气压计等传感器, 通过机器人运动学和卡尔曼滤波等算法，实现对机器人的质心位置、速度和姿态的估计。通过机器人在平面运动和接触点不打滑的基本假设，建立本体系下足部速度与世界系下足部速度的映射，来估计世界系下机器人的姿态，质心的位置和速度。

ABSTRACT

Quadruped robots are widely used in fields such as exploration and rescue. The motion algorithms and navigation planning of quadruped robots require accurate estimation of their own states. This paper utilizes various sensors of quadruped robots, including joint encoders, inertial measurement units (IMU), and foot pressure sensors. Through algorithms such as robot kinematics and Kalman filtering, the estimation of the robot's position, velocity, and attitude is achieved. By making the basic assumption that the robot moves in a plane and the contact points do not slip, a mapping between foot velocity in the robot's reference frame and foot velocity in the world reference frame is established, in order to estimate the robot's position,velocity and attitude in the world reference frame.

目 录

1 概述

四足机器人凭借其良好的通过性、机动性，在工业、服务乃至军事领域拥有较为广泛的应用价值。目前研究出运动性能优越的四足机器人，已经被广泛用于执行巡检、搜救、测绘等任务。四足机器人依靠腿部关节电机的摆动生成步态以驱动机器人运动，相较于轮式机器人仅依靠轮子与地面的摩擦进行运动具有更复杂的运动特性。本文基于提供的四足机器人数据包，实现一种通过卡尔曼滤波算法融合四足机器人运动学和 IMU 的足式里程计算法，推算机器人的运动状态，包括位置、速度和姿态。

假设机器人的整个过程都在平面上运动，并且足部与地面接触时不发生打滑的现象。在提供的 ROS 数据包中，包含以下话题：

- **/aliengo/imu** 为机器人身体的 IMU 数据，“sensor_msgs/Imu” 类型，1000Hz。
- **/aliengo/joint_states** 为机器人的关节和足部数据，包括角度、角速度、扭矩等，“sensor_msgs/JointState” 类型，1000Hz。
- **/vrpn_client_node/aliendog/pose** 动作捕捉仪提供的机器人位置、姿态真值，“geometry_msgs/PoseStamped” 类型，100Hz。
- **/vrpn_client_node/aliendog/twist** 动作捕捉仪提供的机器人运动速度真值，“geometry_msgs/TwistStamped” 类型，100Hz。
- **/tf** 世界坐标系（world）到机器人坐标系（trunk/base）的 tf 变换，内容与**/vrpn_client_node/aliendog/twist** 话题一致。

其中，“aliengo_vrpn_joint_1.bag” 为较简单的运动场景，包含了前进、后退、左右平移、原地旋转等基础动作，参考图片 “trajectory_1.png”；

“aliengo_vrpn_joint_2.bag” 中的运动轨迹为环绕场地画框，参考图片 “trajectory_2.png”；

“aliengo_vrpn_joint_3.bag” 包含较为复杂的运动轨迹（8 字绕圈），参考图片 “trajectory_3.png”，是平移运动和旋转运动的随机组合。

2 四足机器人运动学建模

四足机器人可以视作是一个多刚体结构，由多个刚体和关节组成。在四足机器人运动学建模中，常使用运动学树来描述各个刚体和关节之间连接的拓扑结构。本文中十二自由度的四足机器人的运动学树结构如图1所示。

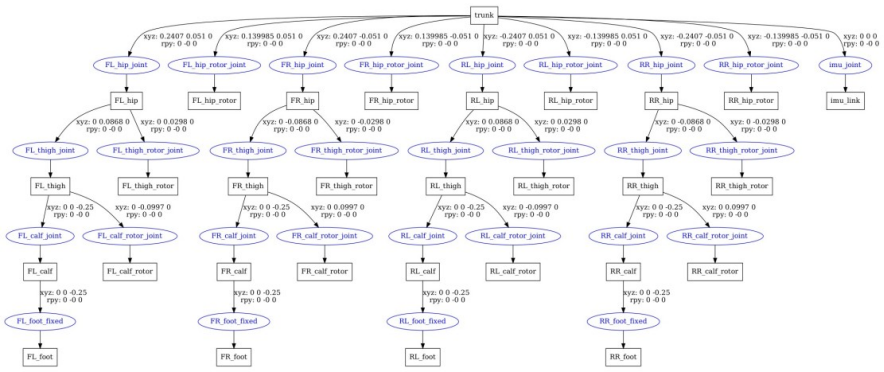


图 1: aliengo 结构

运动学模型就是表示四足机器人各个刚体在空间中的相对位置和姿态关系。可以看出，该四足机器人 aliengo 每条腿分布有三个连杆刚体和三个关节，加上躯干作为一个刚体，四足机器人的机体上总共有十三个刚体坐标系，同时还有四个足部坐标系。各坐标系和各关节的初始零位如图2示。

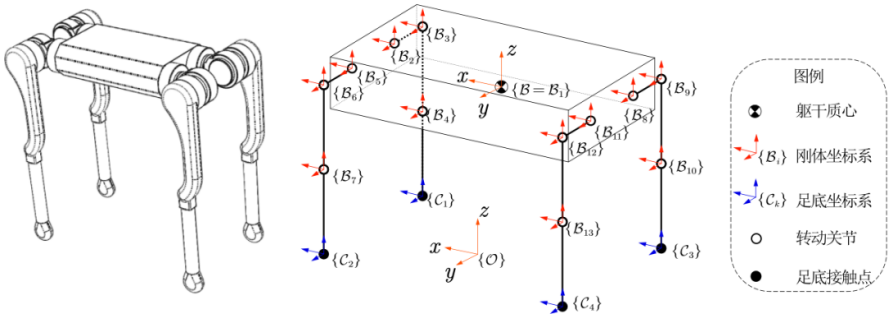


图 2: aliengo 坐标系

为了描述组成机器人的各个刚体在空间中的位置及姿态，首先定义一个刚接到外部物理环境的世界坐标系 O，简称世界系。若机器人在水平地面上开机，在启动控制器的瞬间，O 的 z 轴垂直向上且经过机器人躯干的几何中心，x 轴水平向前，y 轴水平向左，O 的原点在 z 轴与地面的交点上。图3展示了与坐标系之间相对位置有关的机器人物理尺寸，具体参数为：

$$\begin{bmatrix} h_x & h_y & l_1 & l_2 & l_3 \end{bmatrix}^T = \begin{bmatrix} 0.24m & 0.051m & 0.088m & 0.25m & 0.26m \end{bmatrix}^T$$

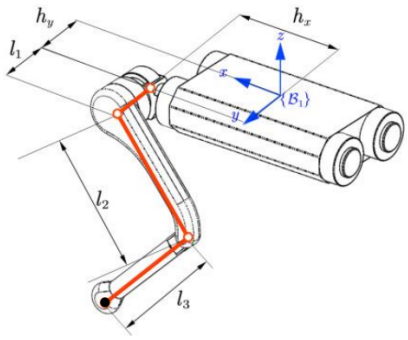


图 3: aliengo 尺寸示意图

根据关节处编码器给出的角度和角速度数据，可以利用四足机器人的运动学信息，结合足部与地面接触时不打滑的假设，来反推出机器人质心的速度。

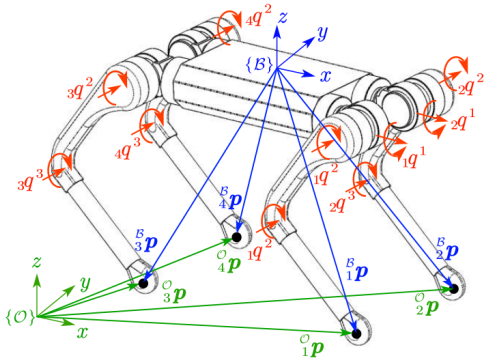


图 4: aliengo 参数示意

如图4所示，定义世界坐标系 O，定义坐标原点在四足机器人机体几何中心的机体系 B，一般 IMU 固定在四足机器人机体上，因此 IMU 的坐标系和机体系之间的空间变换矩阵是固定的，在后续的章节里，会对 IMU 的坐标系进行进一步的分析，通过坐标变换的方法使 IMU 的坐标系的原点和方向与机体系一致。因此，下文的推导过程，认为 IMU 坐标系的原点和方向与机体系一致。定义 ${}^{O}\boldsymbol{p}_i^f$ 为第 i 条腿的足端在世界系下的位置矢量，${}^{B}\boldsymbol{p}_i^f$ 为第 i 条腿的足端在机体系下的位置矢量。

以第一条腿为例，根据简单的几何变换，可以得到足端在机体系下的位置矢量为：

$$
{}^{\mathcal{B}}\boldsymbol{p}_i^f = \begin{bmatrix} {}^{\mathcal{B}}_i p_x^f \\ {}^{\mathcal{B}}_i p_y^f \\ {}^{\mathcal{B}}_i p_z^f \end{bmatrix} = \begin{bmatrix} -l_2 s_2 - l_3 s_{23} + \delta h_x \\ \zeta l_1 c_1 + l_3 s_1 c_{23} + l_2 c_2 s_1 + \zeta h_y \\ \zeta l_1 s_1 - l_3 c_1 c_{23} - l_2 c_1 c_2 \end{bmatrix} \tag{1}
$$

其中 s_i 表示 $\sin(qi)$，c_i 表示 $\cos(qi)$，s_{ij} 表示 $\sin(qi+qj)$，c_{ij} 表示 $\cos(qi+qj)$。δ 为一个符号变量，和腿的编号有关，有：

$$
\zeta = \begin{cases} 1, i \text{ 为左腿} \\ -1, i \text{ 为右腿} \end{cases}, \delta = \begin{cases} 1, i \text{ 为前腿} \\ -1, i \text{ 为后腿} \end{cases} \tag{2}
$$

得到足端在机体系下的位置矢量后，足端在世界系下的位置矢量即可表示为：

$$
{}^{\mathcal{O}}\boldsymbol{p}_i^f = {}^{\mathcal{O}}\boldsymbol{p}_{COM} + {}^{\mathcal{O}}\boldsymbol{R}_{\mathcal{B}}{}^{\mathcal{B}}\boldsymbol{p}_i^f \tag{3}
$$

其中 ${}^{\mathcal{O}}p_{COM}^f$ 表示机体中心在世界系下的位置矢量，${}^{\mathcal{O}}p_i^f$ 表示机体系世界系的旋转矩阵。

对式1求偏导即可得到关节速度和足端在机体系下的线速度之间的映射关系：

$$
{}^{\mathcal{B}}\dot{\boldsymbol{p}}_1^f = \begin{bmatrix} 0 & l_2 c_2 + l_3 c_{23} & l_3 c_{23} \\ -\delta l_1 s_1 + l_2 c_1 c_2 + l_3 c_1 s_{23} & -l_2 s_1 s_2 - l_3 s_1 s_{23} & -l_3 s_1 s_{23} \\ \delta l_1 c_1 + l_2 s_1 c_2 + l_3 s_1 c_{23} & l_2 c_1 s_2 + l_3 c_1 s_{23} & l_3 c_1 s_{23} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} \tag{4}
$$

对式3求导，可得足端在世界系下的速度：

$$
{}^{\mathcal{O}}\dot{\boldsymbol{p}}_i^f = {}^{\mathcal{O}}\boldsymbol{v}_{COM} + {}^{\mathcal{O}}R_{\mathcal{B}}\left({}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}\times}{}^{\mathcal{B}}\boldsymbol{p}_i^f + {}^{\mathcal{B}}\dot{\boldsymbol{p}}_i^f\right) \tag{5}
$$

此外，根据在与地面接触的足端不会打滑的假设，有 ${}^{\mathcal{O}}\dot{\boldsymbol{p}}_i^f = 0$，机体在世界系下的速度可以表示为：

$$
{}^{\mathcal{O}}\boldsymbol{v}_{COM} = -{}^{\mathcal{O}}R_{\mathcal{B}}\left({}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}\times} \times {}^{\mathcal{B}}\boldsymbol{p}_i^f + {}^{\mathcal{B}}\dot{\boldsymbol{p}}_i^f\right) \tag{6}
$$

3 IMU 误差模型

姿态角被视为是四足机器人重要的运动状态之一，它不仅对机器人的运动控制至关重要，而且对于正运动学的解算也必不可少。由于 IMU 可能安装到机器人躯干的任何一个地方，因此输出的原始数据并不能直接表示躯干的旋转状态。

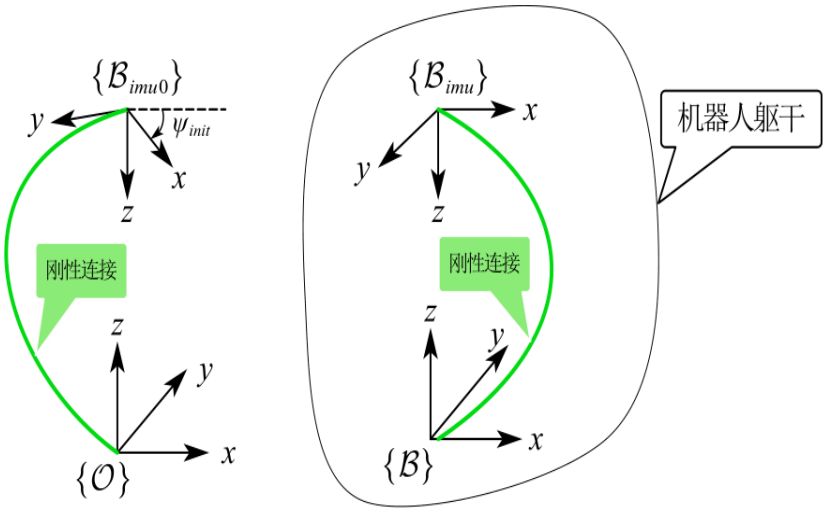


图 5: IMU 坐标关系

如上图所示，设 IMU 零系 $\mathcal{B}_{imu0}$，IMU 本体系为 $\mathcal{B}_{imu}$，本体系 B，世界系 O，当 IMU 本体系与 IMU 零系方向相同时，IMU 输出的旋转矩阵是单位阵。根据组合旋转的计算规则，陀螺仪原始旋转矩阵数据 ${}^{imu0}R_{imu}$ 与躯干旋转矩阵 ${}^{\mathcal{O}}R_{\mathcal{B}}$ 之间有如下的算术关系：

$$ {}^{\mathcal{O}}R_{\mathcal{B}} = {}^{\mathcal{O}}R_{\text{imu0}}\ {}^{imu0}R_{\text{imu}}\ {}^{imu}R_{\mathcal{B}} = {}^{\mathcal{O}}R_{\text{imu}}\ {}^{imu}R_{\mathcal{B}} \tag{7} $$

可以根据控制器启动瞬间，IMU 给出的姿态角和动作捕捉仪给出的姿态角进行比较，从而求出矩阵 ${}^{imu}R_{\mathcal{B}}$。

在四足机器人系统上，以 bag3 给出的数据为例，bag3 包含的是一些较为复杂的运动，下图展示的是 bag3 数据包中/aliengo/imu 数据包给出的 IMU 姿态角数据和/vrpn_client_node/aliendog/pose 给出的机器人姿态真值数据的对比，从左到右按照 ZYX 的欧拉角顺序，分别为 Roll,Pitch 和 Yaw 角数据。其中 Yaw 角的测量准确与否对于里程计的定位至关重要，这个量决定了机器人在 xy 平面内移动的方位角。

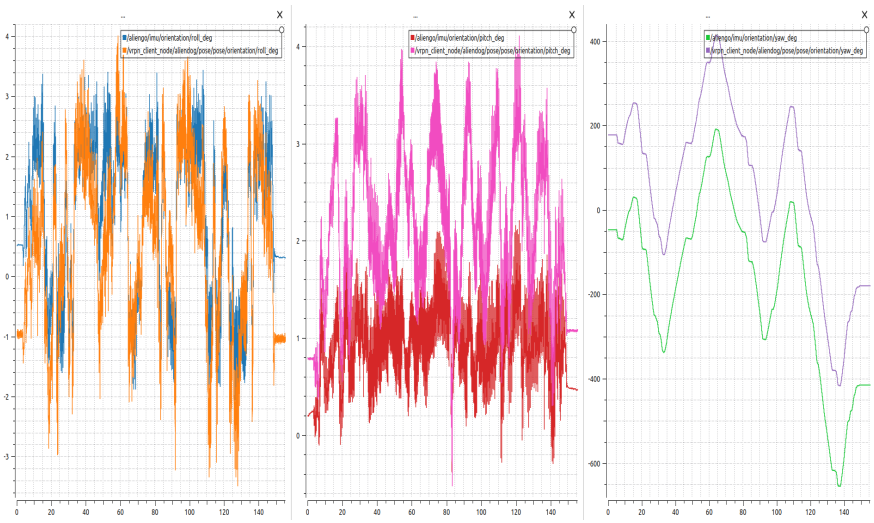


图 6: IMU 原始数据与真值对比

按照前文叙述的方法，在控制器启动的瞬间，按 IMU 给出的姿态角和动作捕捉仪给出的姿态角进行比较作差，将这一常数补偿至 IMU 给出的原始数据上，计算出 ${}^{imu}R_{\mathcal{B}}$，作为估计的姿态角数据。下图给出的即为估计的姿态角数据，在话题/estimation_body_pose 中，和/vrpn_client_node/aliendog/pose 给出的机器人姿态真值数据对比，从左到右按照 ZYX 的欧拉角顺序，分别为 Roll,Pitch 和 Yaw 角数据。

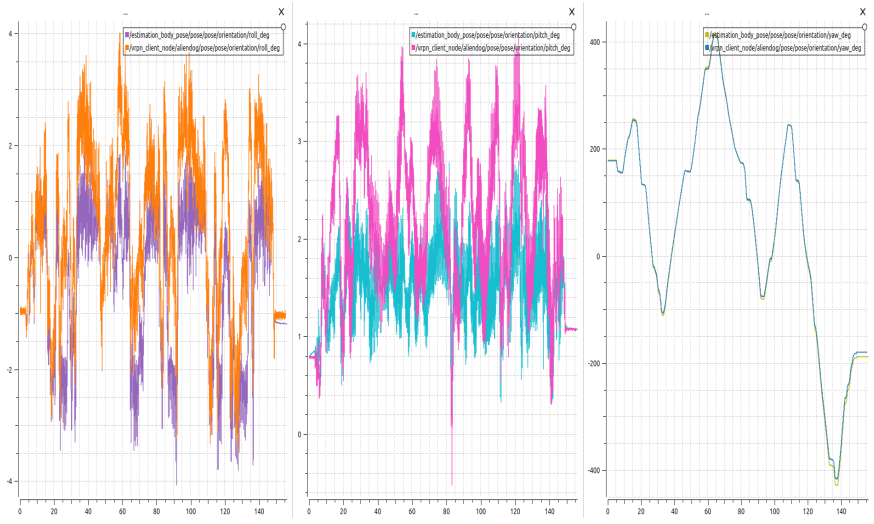


图 7: IMU 补偿后的数据与真值对比

给出补偿前后的数据，对比如下：

	Roll_Max	Roll_Min	Roll_Average
IMU 原始数据	3.44	-2.58	0.99
补偿数据	1.94	-4.08	-0.51
真值	4.01	-3.50	0.52
	Pitch_Max	Pitch_Min	Pitch_Average
IMU 原始数据	2.21	-0.48	0.90
补偿数据	2.80	0.12	1.50
真值	4.11	-0.53	2.09
	Yaw_Max	Yaw_Min	Yaw_Average
IMU 原始数据	191.52	-655.18	-172.44
补偿数据	417.50	-429.12	52.65
真值	413.35	-416.83	56.21

Roll_Max 代表整个数据包中 roll 角的最大值，Roll_Min 代表整个数据包中 roll 角的最小值，Roll_Average 代表整个数据包中 roll 角的平均值，Pitch 和 Yaw 角也如此定义。可见，在补偿后改善了 IMU 对姿态角的测量效果。

此外，加速度和角速度可以直接利用 rosbag 里/aliengo/imu 给出的角速度和加速度数据。

4 位置速度估计

4.1 状态转移方程

由于 IMU 测得的参数都是在机器人本体系下的描述，可以通过旋转矩阵变换到世界系：

$$\begin{cases} {}^{\mathcal{O}}\boldsymbol{a}_{com} = {}^{\mathcal{O}}R_{\mathcal{B}} \cdot {}^{\mathcal{B}}\boldsymbol{a}_{com} \\ {}^{\mathcal{O}}\boldsymbol{\omega}_{\mathcal{OB}} = {}^{\mathcal{O}}R_{\mathcal{B}} \cdot {}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{OB}} \end{cases} \tag{8}$$

在这里设计状态空间方程，因为需要对质心的位置和速度进行估计，因此选取的状态量为：

$$\boldsymbol{x} = \begin{bmatrix} {}^{\mathcal{O}}\boldsymbol{p}_{COM}^T & {}^{\mathcal{O}}\boldsymbol{v}_{COM}^T & {}^{\mathcal{O}}\boldsymbol{p}_f^T \end{bmatrix}^T \tag{9}$$

其中，${}^{\mathcal{O}}\boldsymbol{p}_f = \begin{bmatrix} {}^{\mathcal{O}}\boldsymbol{p}_1^{f^T} & {}^{\mathcal{O}}\boldsymbol{p}_2^{f^T} & {}^{\mathcal{O}}\boldsymbol{p}_3^{f^T} & {}^{\mathcal{O}}\boldsymbol{p}_4^{f^T} \end{bmatrix}^T$ 表示四条腿足端在世界系下的位置。定义系统的输入为加速度计的输出和重力加速度的组合：$\boldsymbol{u} = {}^{\mathcal{O}}\boldsymbol{a}_{COM}^T + {}^{\mathcal{O}}\boldsymbol{g}$。根据状态量之间的物理关系，可以写出连续状态的物理方程：

$$\begin{cases} {}^{o}\dot{\boldsymbol{p}}_{com} = {}^{\mathcal{O}}\boldsymbol{v}_{com} \\ {}^{\mathcal{O}}\dot{\boldsymbol{v}}_{com} = {}^{\mathcal{O}}\boldsymbol{a}_{com} + {}^{\mathcal{O}}\boldsymbol{g} \\ \quad {}_i\dot{\boldsymbol{p}} = 0 \quad \forall i \in \{1,2,3,4\} \end{cases} \tag{10}$$

根据连续方程，可以写出如下的离散状态转移方程：

$$\boldsymbol{x}_{k+1} = A_k\boldsymbol{x}_k + \boldsymbol{B}_k\boldsymbol{u}_k + \boldsymbol{v}_k \tag{11}$$

其中 $\boldsymbol{A}_k \in R^{18\times18}$ 表示状态转移矩阵，$\boldsymbol{B}_k \in R^{18\times3}$ 表示控制矩阵，在这里，状态转移矩阵和控制矩阵都是时不变的，因此在下文中简写为 A 和 B。$\boldsymbol{v}_k$ 表示过程噪声，可视为高斯白噪声，满足一个 N(0,Q) 的正态分布。分析系统状态和系统输入的关系，可以得到状态转移矩阵和控制矩阵的形式为：

$$\boldsymbol{A} = \begin{bmatrix} \mathbf{I}_{3\times3} & \Delta t \times \mathbf{I}_{3\times3} & \mathbf{0}_{3\times12} \\ \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times12} \\ \mathbf{0}_{12\times3} & \mathbf{0}_{12\times3} & \mathbf{I}_{12\times12} \end{bmatrix} \tag{12}$$

$$\boldsymbol{B} = \begin{bmatrix} \mathbf{0}_{3\times3} \\ \Delta t \times \mathbf{I}_{3\times3} \\ \mathbf{0}_{12\times3} \end{bmatrix} \tag{13}$$

其中，Δt 为控制周期。

4.2 观测方程

为了利用足端里程计的信息，定义系统的观测量为：

$$z = \begin{bmatrix} \left({}^{\mathcal{O}}\boldsymbol{p}_{COM} - {}^{\mathcal{O}}\boldsymbol{p}_1^f\right)^T \\ \left({}^{\mathcal{O}}\boldsymbol{p}_{COM} - {}^{\mathcal{O}}\boldsymbol{p}_2^f\right)^T \\ \left({}^{\mathcal{O}}\boldsymbol{p}_{COM} - {}^{\mathcal{O}}\boldsymbol{p}_3^f\right)^T \\ \left({}^{\mathcal{O}}\boldsymbol{p}_{COM} - {}^{\mathcal{O}}\boldsymbol{p}_4^f\right)^T \\ \left({}^{\mathcal{O}}\boldsymbol{v}_1^f\right)^T \\ \left({}^{\mathcal{O}}\boldsymbol{v}_2^f\right)^T \\ \left({}^{\mathcal{O}}\boldsymbol{v}_3^f\right)^T \\ \left({}^{\mathcal{O}}\boldsymbol{v}_4^f\right)^T \\ p_{z,1}^f \\ p_{z,2}^f \\ p_{z,3}^f \\ p_{z,4}^f \end{bmatrix}^T \tag{14}$$

其中 $\left({}^{\mathcal{O}}\boldsymbol{p}_{COM} - {}^{\mathcal{O}}\boldsymbol{p}_i^f\right)$ 为机体中心到第 i 个足端的位置矢量在世界系下的描述，$\left({}^{\mathcal{O}}\boldsymbol{v}_i^f\right)^T$ 为第 i 个足端对机体在世界系下速度的观测，$p_{z,i}^f$ 表示第 i 个足端在世界系下的离地高度。

将之前的公式整合到一起，可以得到观测量为

$$z = \begin{bmatrix} -{}^{\mathcal{O}}R_{\mathcal{B}} \cdot {}^{\mathcal{B}}_1\boldsymbol{p} \\ -{}^{\mathcal{O}}R_{\mathcal{B}} \cdot {}^{\mathcal{B}}_2\boldsymbol{p} \\ -{}^{\mathcal{O}}R_{\mathcal{B}} \cdot {}^{\mathcal{B}}_3\boldsymbol{p} \\ -{}^{\mathcal{O}}R_{\mathcal{B}} \cdot {}^{\mathcal{B}}_4\boldsymbol{p} \\ -{}^{\mathcal{O}}R_{\mathcal{B}} \cdot \left({}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{OB}\times} \cdot {}^{\mathcal{B}}_1\boldsymbol{p} + {}^{\mathcal{B}}_2\dot{\boldsymbol{p}}\right) \\ -{}^{\mathcal{O}}R_{\mathcal{B}} \cdot \left({}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{OB}\times} \cdot {}^{\mathcal{B}}_2\boldsymbol{p} + {}^{\mathcal{B}}_2\dot{\boldsymbol{p}}\right) \\ -{}^{\mathcal{O}}R_{\mathcal{B}} \cdot \left({}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{OB}\times} \cdot {}^{\mathcal{B}}_3\boldsymbol{p} + {}^{\mathcal{B}}_3\dot{\boldsymbol{p}}\right) \\ -{}^{\mathcal{O}}R_{\mathcal{B}} \cdot \left({}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{OB}\times} \cdot {}^{\mathcal{B}}_4\boldsymbol{p} + {}^{\mathcal{B}}_4\dot{\boldsymbol{p}}\right) \\ {}^{\mathcal{O}}_1p^z \\ {}^{\mathcal{O}}_2p^z \\ {}^{\mathcal{O}}_3p^z \\ {}^{\mathcal{O}}_4p^z \end{bmatrix} \tag{15}$$

据此可以得到观测矩阵 H 有如下形式：

$$\boldsymbol{H}=\begin{bmatrix} & \mathbf{I}_{12\times3} & \mathbf{0}_{12\times3} & \mathbf{I}_{12\times12} & & & \\ & \mathbf{0}_{12\times3} & \mathbf{I}_{12\times3} & \mathbf{0}_{12\times12} & & & \\ \mathbf{0}_{1\times6} & 0 & 0 & 1 & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} \\ \mathbf{0}_{1\times6} & \mathbf{0}_{1\times3} & 0 & 0 & 1 & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} \\ \mathbf{0}_{1\times6} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & 0 & 0 & 1 & \mathbf{0}_{1\times3} \\ \mathbf{0}_{1\times6} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & 0 & 0 & 1 \end{bmatrix} \tag{16}$$

最终系统的离散观测方程为：

$$\boldsymbol{z}_k=\boldsymbol{H}\boldsymbol{x}_k+\boldsymbol{w}_k \tag{17}$$

其中 w_k 为观测噪声，可视为高斯白噪声，满足一个 $N(0,R)$ 的正态分布。由于观测量主要是通过关节编码器反馈的数据计算得到的，因此此处观测噪声的大小取决于编码器测量噪声。根据足端里程计的原理，摆动腿足端的速度对机体在世界系下的速度没有观测意义，因此当足部在与地面没有接触的情况下，将观测噪声大大增加即可。

由于足底里程计求解质心位置是个积分过程，必然存在累计误差。在 x 方向与 y 方向的累计位置误差不会对控制器造成大的影响；但是在高度方向若有误差，就会直接体现在机器人行走时质心高度与抬腿高度的控制上。因此假设所有支撑足在世界系下高度名义上为 0。在这一假设下，可以通过足底在本体系下的位置，反过来求解质心在世界系下的高度。即观测值中的足底高度项

$$ {}^{o}_{i}p^{z}=0 \tag{18}$$

4.3 卡尔曼滤波融合

本文所述四足机器人使用的传感器，包括测量关节角度 ${}_{i}q^{j}$ 与转速 ${}_{i}\dot{q}^{j}$ 的电机编码器，其中 i 为腿的序号，为关节序号；以及测量质心加速度 ${}^{\mathcal{B}}\boldsymbol{a}_{com}$、躯干旋转矩阵 ${}^{\mathcal{O}}R_{\mathcal{B}}$、躯干角速度 ${}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{OB}}$ 的 IMU。机器人躯干的转动状态（旋转矩阵与角速度）可以直接从 IMU 读取，还需要融合多种传感器数据来获取质心在世界系下的位置 ${}^{\mathcal{O}}\boldsymbol{p}_{com}$ 与速度 ${}^{\mathcal{O}}\boldsymbol{v}_{\mathrm{com}}$。

由前文，四足机器人系统的状态方程和观测方程均已给出，在状态转移方程和观测方程的基础上，本文使用线性卡尔曼滤波器对其进行融合估计。

所谓卡尔曼滤波器，实际上就是一个状态观测器，它利用传感器融合、信息融合来提高系统的精度。在四足机器人系统上，我们通过 IMU 和编码器两种传感器对机器人的状态 x 进行估计，这两种方式都有各自的不确定

性，卡尔曼滤波可以将这两者做到最优结合（加权平均），使得我们估计的状态的不确定性小于其中任何一种。

以下为卡尔曼滤波器的方程，包含此前定义在状态转移方程和观测方程中的矩阵，可以直接将它们带入：

$$\begin{cases} \hat{x}_{k|k-1} = A\hat{x}_{k-1} + Bu_k \\ P_{k|k-1} = AP_{k-1}A^T + Q \\ K_k = \frac{P_{k|k-1}H^T}{HP_{k|k-1}H^T+R} \\ \hat{x}_k = \hat{x}_{k|k-1} + K_k\left(z_k - H\hat{x}_{k|k-1}\right) \\ P_k = \left(I - K_kH\right)P_{k|k-1} \end{cases} \tag{19}$$

其中

$$Q = \begin{bmatrix} \frac{Idt\text{noise}_p}{20} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{9.8Idt\text{noise}}{20} & 0 & 0 & 0 & 0 \\ 0 & 0 & Idt\text{noise}_{p1} & 0 & 0 & 0 \\ 0 & 0 & 0 & Idt\text{noise}_{p2} & 0 & 0 \\ 0 & 0 & 0 & 0 & Idt\text{noise}_{p3} & 0 \\ 0 & 0 & 0 & 0 & 0 & Idt\text{noise}_{p4} \end{bmatrix}$$

表示过程噪声的协方差矩阵，

$$R = \begin{bmatrix} I_{12\times12}\ \text{noise}_{\ \text{pfoot}} & \cdots & 0 \\ \vdots & I_{12\times12}\ \text{noise}_{\ vfoot} & \vdots \\ 0 & \cdots & I_{4\times4}\ \text{noise}_{\ zfoot} \end{bmatrix}_{28\times28}$$

表示观测噪声的协方差矩阵，P_k 为先验估计协方差矩阵，$P_{k|k-1}$ 为后验估计协方差矩阵，K_k 为卡尔曼滤波增益。通过卡尔曼滤波来对先验估计结果进行修正，最终得到一个协方差最小的新状态分布。

通过增大摆动腿对应的过程噪声和观测噪声，削弱摆动腿的运动学信息对四足机器人位置速度估计结果的影响。在判断四足机器人的足端与地面的接触状态这个问题上，由于已经给到了足部压力传感器的数据，因此便于对足部接触情况进行判断，当/effort 数据大于 50 时，就认为足端与地面发生不打滑的充分的接触。

经调试，最终确定的参数为：

参数	值
$noise_p$	0.35
$noise_v$	0.06
$noise_{pi}$	0.01
$noise_{pfoot}$	0.01
$noise_{vfoot}$	0.005
$noise_{zfoot}$	0.1

5 结果分析

为了验证算法的性能，在给到的三个 ros 数据包中进行数据验证，将动作捕捉仪采集到的位置、速度和姿态数据作为状态估计的真值，实验基于 Linux 下的机器人操作系统（ROS），在个人电脑（Intel i5-1240P）上运行。

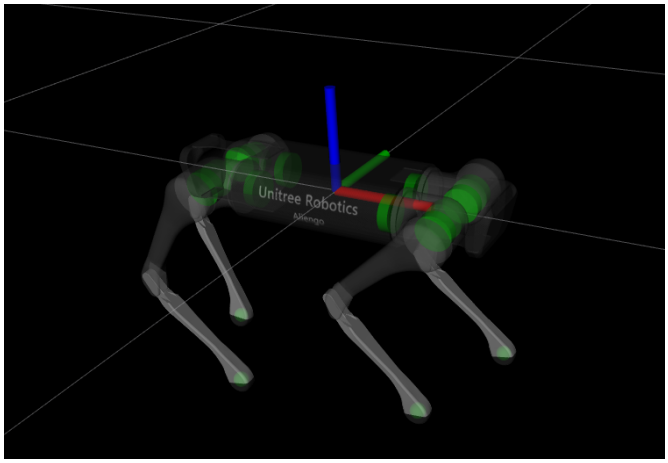


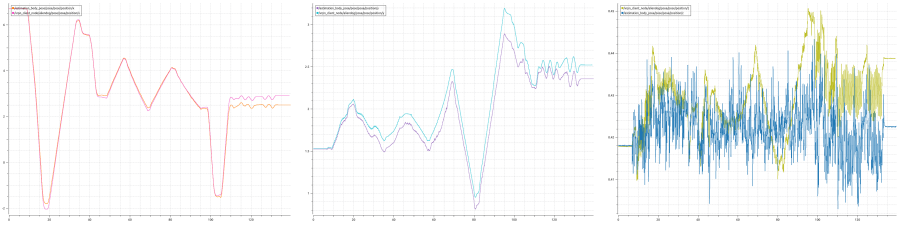
图 8: 四足机器人实验平台

笔者已将输出数据存于三个 ros 包中便于数据复现与分析，见 output-bag 文件夹。

5.1 位置分析

给到的三个数据包用于进行算法测试，分别包含了较简单的运动场景（前进后退、平移、旋转）、环绕场地画方框、较复杂的运动轨迹（8 字绕圈、平移旋转的随机组合）。下面有图片形式给出预测值和真值的数值对比，分别以 x,y,z 三个方向和 xy 平面轨迹图的方式给出。

以下是 bag1 的数据，一张图里的两个曲线分别是估计值和真值。



(a) bag1_x (b) bag1_y (c) bag1_z

图 9: bag1 位置估计

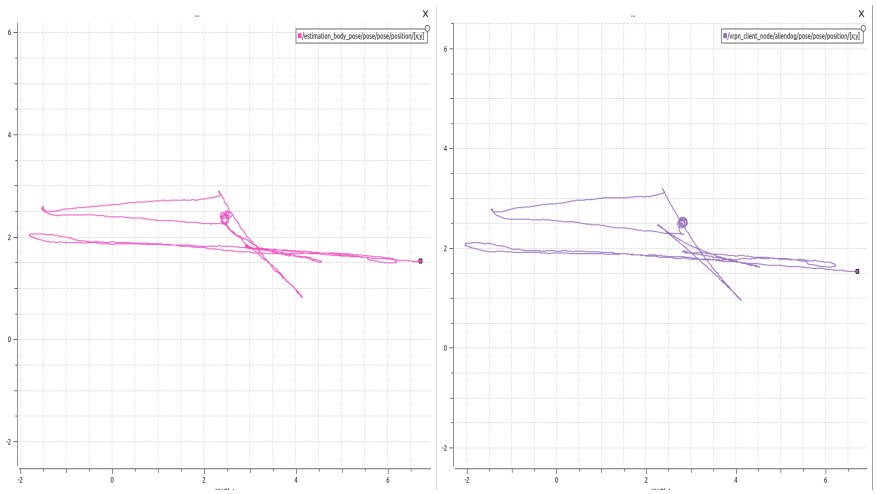
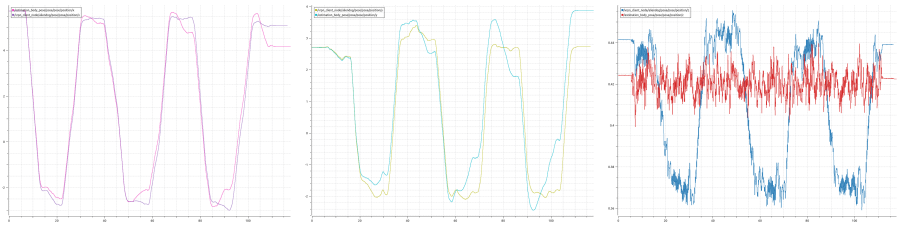


图 10: bag1 轨迹图



(a) bag2_x (b) bag2_y (c) bag2_z

图 11: bag2 位置估计

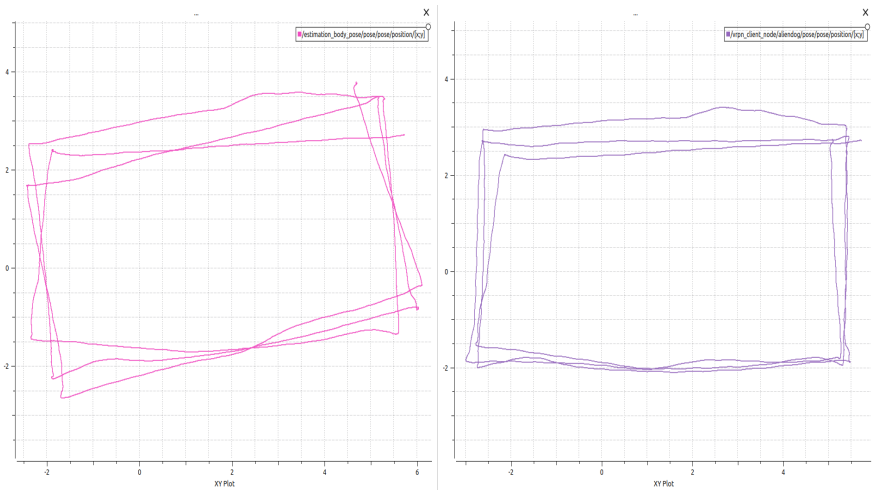
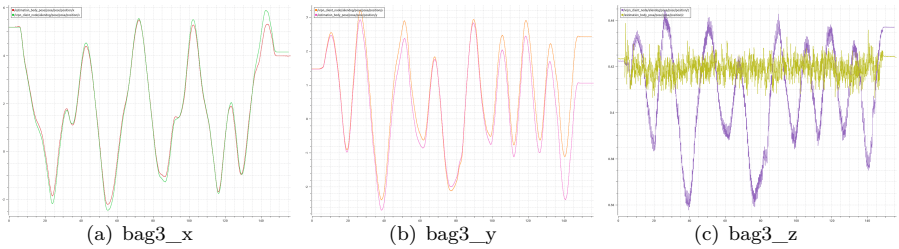


图 12: bag2 轨迹图



(a) bag3_x (b) bag3_y (c) bag3_z

图 13: bag3 位置估计

图 14: bag3 轨迹图

从上图可以看出，随着机器人的行驶距离的增加，误差逐渐变大，这是因为在真实物理世界中，机器人运行时存在物理参数的微小变化，同时与地面的接触也会存在滑动，且各种传感器的测量都会存在有随机误差，这些因素会造成累计误差的存在，因此随着行驶距离增加，误差也会越来越大。

下面对足式里程计的估计值进行精度评价，下图是 bag1-3 的误差曲线。

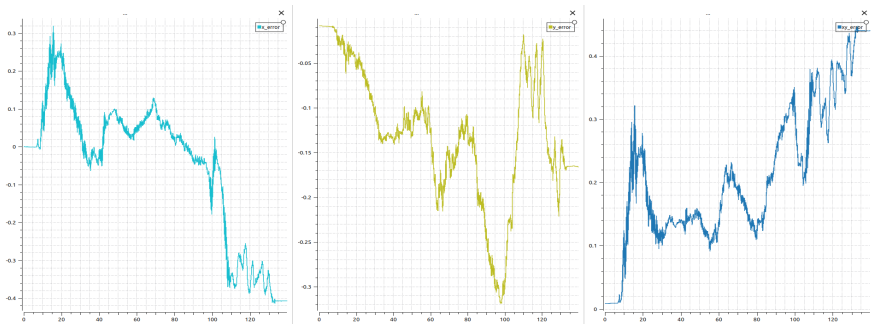


图 15: bag1 位置误差

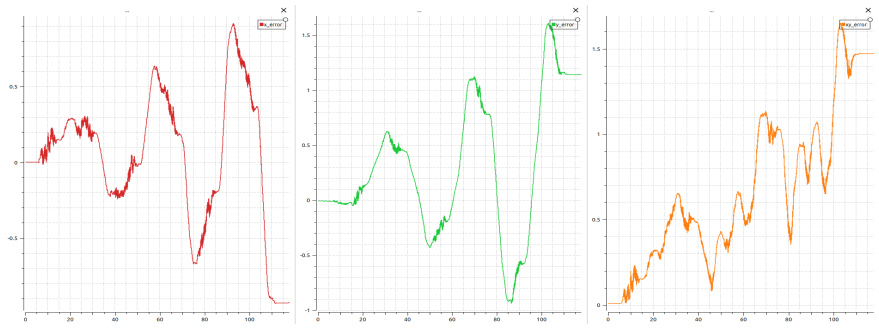


图 16: bag2 位置误差

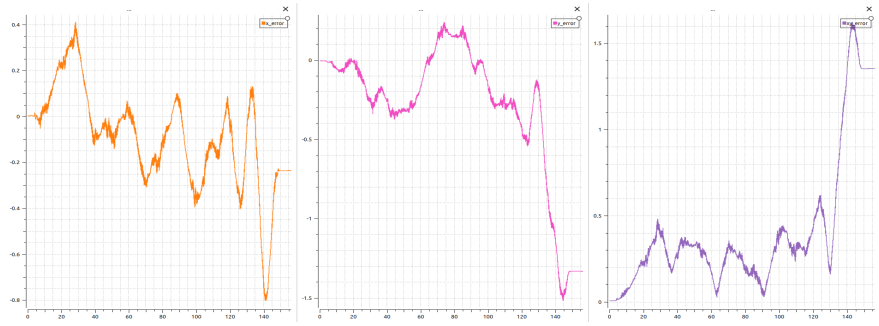


图 17: bag3 位置误差

下表罗列了多个评价指标，包括最大值（max），最小值（min），均值（mean），均方根误差（rmse）构成。

表 1: 位置误差数据（单位 m）

	x 方向误差最大值	y 方向误差最大值	xy 平面误差最大值	x 方向平均误差
bag1	0.41	0.31	0.44	0.14
bag2	1.33	1.46	1.69	0.41
bag3	0.93	1.61	1.65	0.34
	y 方向平均误差	x 方向均方根误差	y 方向均方根误差	
bag1	0.13	0.44	0.15	
bag2	0.51	0.73	0.65	
bag3	0.52	0.66	0.68	

通过对于估计误差的分析，结果表明，足部里程计的均方根误差在 0.6m 左右，具有一定的可信度，可以比较准确地估计机器人的位置。

5.2 速度分析

类似位置分析里的步骤，将 x 和 y 方向的速度展示如下，从左到右以此为 x 方向速度和 y 方向速度：

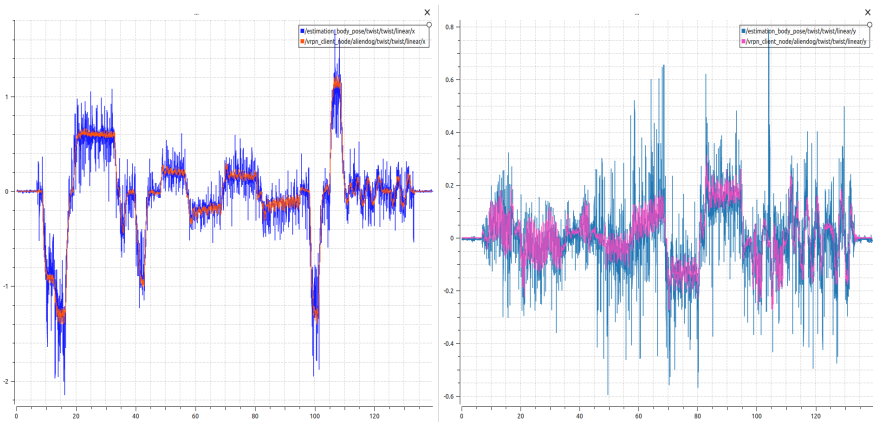


图 18: bag1 速度估计

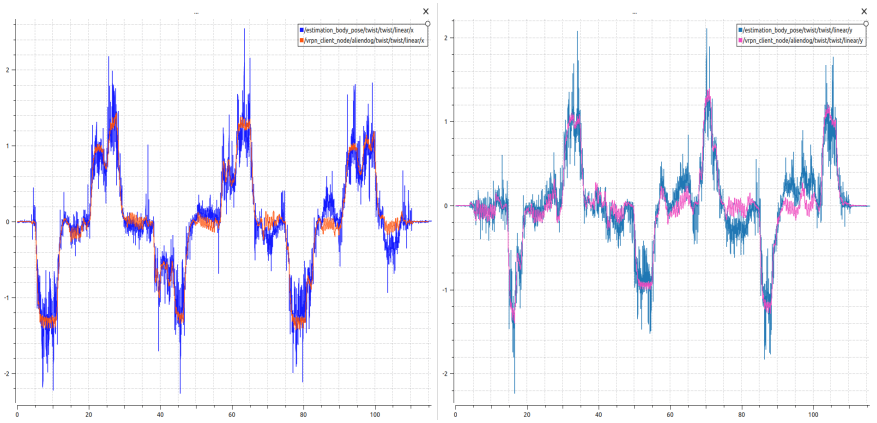


图 19: bag2 速度估计

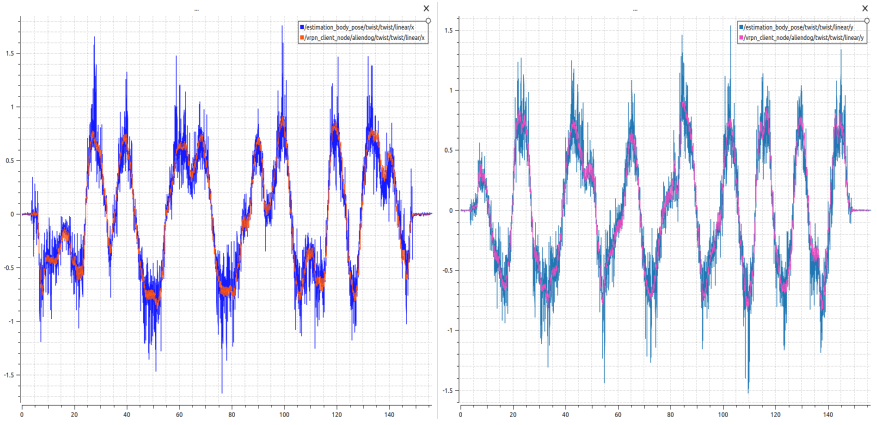


图 20: bag3 速度估计

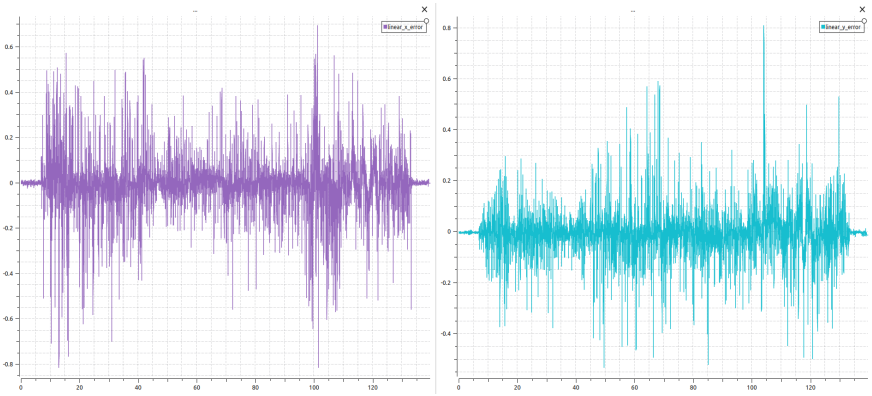


图 21: bag1 速度估计误差

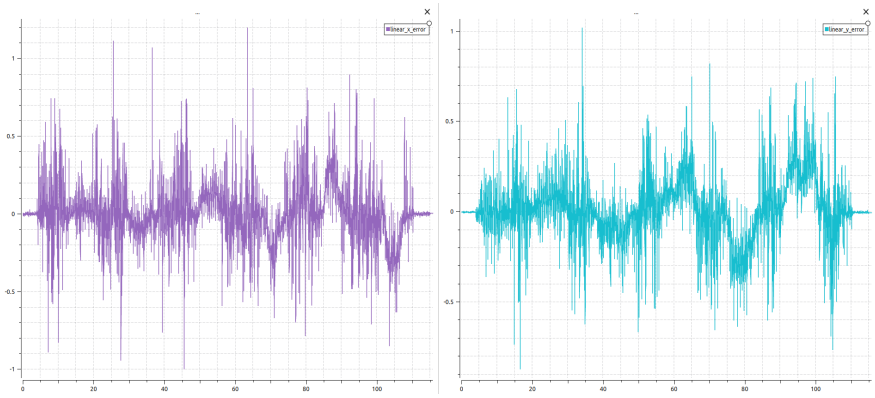


图 22: bag2 速度估计误差

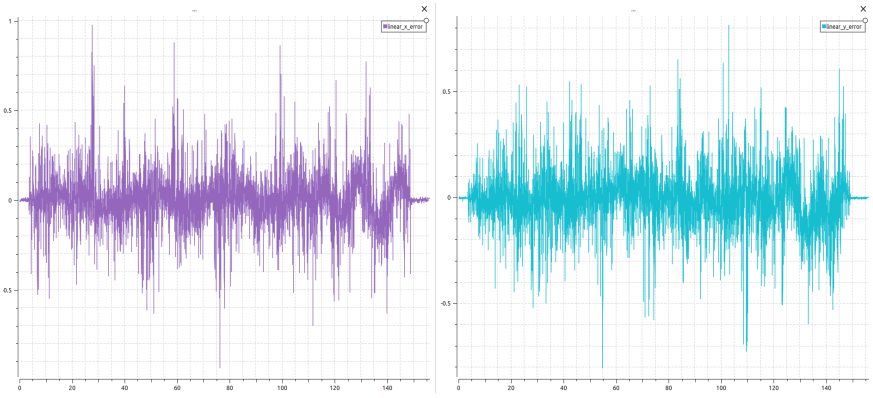


图 23: bag3 速度估计误差

下表罗列了多个评价指标，包括最大值（max），最小值（min），均方根误差（rmse）构成。

表 2: 速度误差数据（单位 m/s）

	x 方向速度最大误差	x 方向速度平均误差	x 方向速度均方根误差
bag1	0.82	-0.0041	0.10
bag2	1.20	-0.0018	0.17
bag3	0.97	-0.0023	0.12
	y 方向速度最大误差	y 方向速度平均误差	y 方向速度均方根误差
bag1	0.81	-0.0056	0.073
bag2	1.02	0.0082	0.16
bag3	0.81	-0.0089	0.11

可以看到在 xy 平面内的速度测量上，误差都比较小，均方根误差也比较小，估计效果较好。

5.3 姿态角分析

由于前文已展示姿态角估计与真值对比，此处仅将 RPY 三个姿态角的估计误差展示如下，从左到右依次为 Roll,Pitch,Yaw：

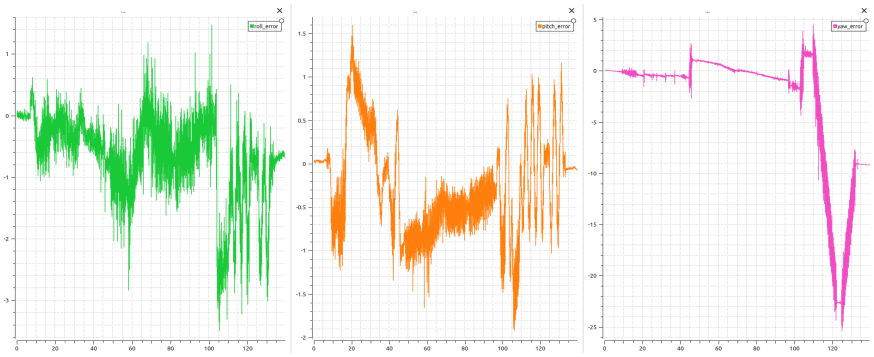


图 24: bag1 姿态角估计误差

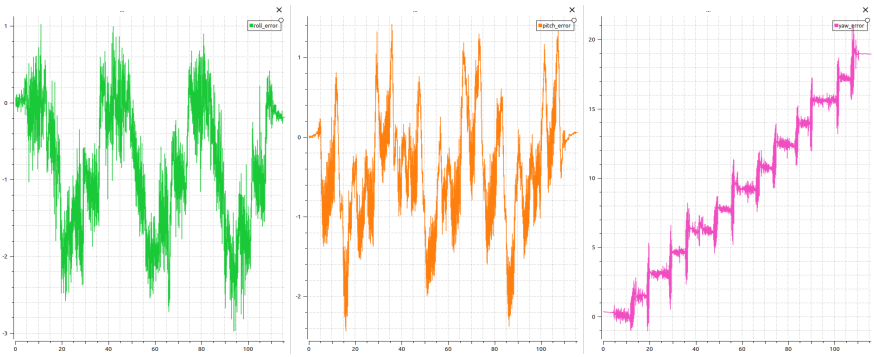


图 25: bag2 姿态角估计误差

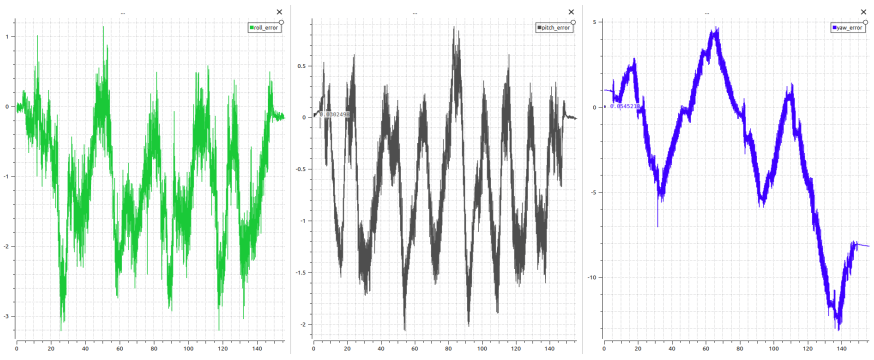


图 26: bag3 姿态角估计误差

表 3: 姿态角误差数据（角度单位：度）

	Roll 最大误差	Pitch 最大误差	Yaw 最大误差
bag1	3.50	1.93	25.41
bag2	2.98	2.44	21.11
bag3	3.22	2.07	13.17
	Roll 平均误差	Pith 平均误差	Yaw 平均误差
bag1	0.69	0.52	3.14
bag2	0.87	0.61	8.93
bag3	1.05	0.64	3.70
	Roll 均方根误差	Pitch 均方根误差	Yaw 均方根误差
bag1	0.94	0.51	6.54
bag2	1.12	0.77	10.67
bag3	0.82	1.29	4.99

可以看到误差主要产生在 yaw 轴上，Roll 轴和 Yaw 轴的均方根误差比较小，Yaw 轴由于无法观测的原因，均方根误差达到 10 度左右，后续可通过增加更多传感器的方式解决。

6 后续展望

机器人是一个多学科交叉的方向。本文对于四足机器人的研究仅仅只是停留在了皮毛的阶段，在平地的假设下完成了较为简易的四足机器人足式里程计，在精度和鲁棒性上都还存在很大的提升空间。但由于笔者的时间和精力有限，这一工作仍存在许多的不足之处可以改进，经笔者归纳反思，在以下几个方面可以进一步探索：

（1）建立更精准的动力学模型。本文动力学模型中主要的误差来自多方面，一是四足机器人各刚体机械参数的变化无法精准测量，二是在模型预测控制中对动力学模型进行了大量的近似和简化。未来可以对于机器人物理参数做到更好的估计，能够考虑在运行中变形的脚和滚动接触会引起微小的腿长变化等运动时不断变化的因素。

（2）增加环境感知设备。仅利用 IMU 和编码器无法对三轴的绝对位置和偏航角做到精准的估计，考虑加入更多的传感器，如深度相机等，或利用多种基于本体的定位方法，提高定位精度。

（3）提高程序的鲁棒性。目前程序的算法仅适用于平面地面，但对于崎岖、柔软、或泥泞的地面，本文提出的算法恐难以达到比较好的效果。需要进一步设计崎岖路面、爬坡、泥泞路面等实验，验证算法的适应性和局限性，并改进算法。

7 结语

一个月的控制之旅时光飞逝，转瞬即逝。在这一个月中，笔者对四足机器人状态估计这一问题进行了深入的学习和研究，开阔了眼界，熟练使用了机器人操作系统（ROS）最终坚持了下来，也实现了基本的状态估计功能。通过此次控制之旅，笔者也对机器人有了更加清晰深刻的认识，也希望未来能够有机会继续在机器人领域深入研究下去；同时，笔者也相信，在无数机器人研究者的努力下，机器人与人类之间也终会筑起一座飞架南北的大桥，从此“天堑变通途”。

参考文献

[1] Bloesch M, Hutter M, Hoepflinger M H,et al.State Estimation for Legged Robots - Consistent Fusion of Leg Kinematics and IMU[J]. 2012.DOI:10.15607/RSS.2012.VIII.003.

[2] 沈雅阁. 针对复杂地形的四足机器人状态估计和运动控制算法研究与应用 [D]. 电子科技大学, 2022.DOI:10.27005/d.cnki.gdzku.2022.002080.

[3] B. E. Jackson, K. Tracy, and Z. Manchester, "Planning with attitude," IEEE Robot. Automat. Lett., vol. 6, no. 3, pp. 5658-5664, Jul. 2021.

[4] S. Yang, H. Choset and Z. Manchester, "Online Kinematic Calibration for Legged Robots," in IEEE Robotics and Automation Letters, vol. 7, no. 3, pp. 8178-8185, July 2022, doi: 10.1109/LRA.2022.3186501.

[5] S. Yang, Z. Zhang, Z. Fu and Z. Manchester, "Cerberus: Low-Drift Visual-Inertial-Leg Odometry For Agile Locomotion," 2023 IEEE International Conference on Robotics and Automation (ICRA), London, United Kingdom, 2023, pp. 4193-4199, doi:10.1109/ICRA48891.2023.10160486.

[6] 姜春丽. 基于误差状态卡尔曼滤波的液压四足机器人状态估计研究 [D]. 哈尔滨工业大学, 2022.

[7] 于宪元. 基于稳定性的仿生四足机器人控制系统设计 [D]. 北京航空航天大学,2021.

[8] 李晨晨, 周志峰. 融合 IMU 的四足机器人腿部里程计 [J]. 农业装备与车辆工程, 2023,61(04):104-108+117.

附录 A　部分代码

部分代码如下，整个四足机器人里程计由 C++ 在机器人操作系统（ROS）下完成。

```
void Ekf::update_estimation(CtrlState &state,double dt)
{
    A.block<3, 3>(0, 3) = dt * eye3;
    B.block<3, 3>(3, 0) = dt * eye3;
    //u = Ra + ag
    Eigen::Vector3d u = state.root_rot_mat * state.imu_acc +
        Eigen::Vector3d(0, 0, -9.81);
    if (state.movement_mode == 0)
    {  //stand状态下，四个足端都是触地的
        for (int i = 0; i < NUM_LEG; ++i)
            estimated_contacts[i] = 1.0;
    }
    else
    {
        for (int i = 0; i < NUM_LEG; ++i)
        {
            estimated_contacts[i] = std::min(std::max(
                (state.foot_force(i)) / (100.0 - 0.0), 0.0), 1.0);
        }
    }
    //update Q
    Q.block<3, 3>(0, 0) = PROCESS_NOISE_PIMU * dt / 20.0 * eye3;
    Q.block<3, 3>(3, 3) = PROCESS_NOISE_VIMU * dt * 9.8 / 20.0 * eye3;
    // update Q R for legs not in contact
    for (int i = 0; i < NUM_LEG; ++i)
    {
        // foot position transition
        Q.block<3, 3>(6 + i * 3, 6 + i * 3) = (1 + (1 -
            estimated_contacts[i]) * 1e3) * dt * PROCESS_NOISE_PFOOT *
            eye3;
        //摆动相加大噪声
        R.block<3, 3>(i * 3, i * 3) = (1 + (1 - estimated_contacts[i]) *
            1e3) * SENSOR_NOISE_PIMU_REL_FOOT * eye3;
        R.block<3, 3>(NUM_LEG * 3 + i * 3, NUM_LEG * 3 + i * 3) = (1 + (1 -
            estimated_contacts[i]) * 1e3) * SENSOR_NOISE_VIMU_REL_FOOT *
            eye3;
        if (assume_flat_ground)
        {
            R(NUM_LEG * 6 + i, NUM_LEG * 6 + i) = (1 + (1 -
                estimated_contacts[i]) * 1e3) * SENSOR_NOISE_ZFOOT;
        }
    }
    xbar = A*x + B*u ;
    Pbar = A * P * A.transpose() + Q;
    //测量
```

```
    yhat = C*xbar;
    for (int i=0; i<NUM_LEG; ++i)
    {
        Eigen::Vector3d fk_pos = state.foot_pos_rel.block<3,1>(0,i);
        // 世界坐标系下足端的位置fk estimation
        y.block<3,1>(i*3,0) = state.root_rot_mat*fk_pos;
        //state.root_rot_matrix是将机体坐标系转换到世界坐标系的矩阵
        Eigen::Vector3d leg_v = -state.foot_vel_rel.block<3,1>(0,i) -
            skew(state.imu_ang_vel)*fk_pos;
        y.block<3,1>(NUM_LEG*3+i*3,0) =
            (1.0-estimated_contacts[i])*x.segment<3>(3) +
            estimated_contacts[i]*state.root_rot_mat*leg_v;
        y(NUM_LEG*6+i) = (1.0-estimated_contacts[i])*(x(2)+fk_pos(2)) +
            estimated_contacts[i]*0;
    }
    S = C * Pbar *C.transpose() + R;
    S = 0.5*(S+S.transpose());
    error_y = y - yhat;
    //这一项是（z-Hx）*（H×P×HT + R）卡尔曼增益的分母
    Serror_y = S.fullPivHouseholderQr().solve(error_y);
    //后验状态估计
    x = xbar + Pbar * C.transpose() * Serror_y;
    SC = S.fullPivHouseholderQr().solve(C);
    P = Pbar - Pbar * C.transpose() * SC * Pbar;
    P = 0.5 * (P + P.transpose());
    //减少位置漂移
    if (P.block<2, 2>(0, 0).determinant() > 1e-6) {
        P.block<2, 16>(0, 2).setZero();
        P.block<16, 2>(2, 0).setZero();
        P.block<2, 2>(0, 0) /= 10.0;
    }
    //大于50N认为接触
    for (int i = 0; i < NUM_LEG; ++i)
    {
        if (estimated_contacts[i] < 0.5)
        {
            state.estimated_contacts[i] = false;
        }
        else
        {
            state.estimated_contacts[i] = true;
        }
    }
    //质心位置与速度
    state.estimated_root_pos = x.segment<3>(0);
    state.estimated_root_vel = x.segment<3>(3);
    state.root_pos = x.segment<3>(0);
    state.root_lin_vel = x.segment<3>(3);
}
```