Cassie HZD双足控制方法说明

此文档同时关联以下文档：

Pinocchio库使用说明

以及以下project:

https://git.ddt.dev:9281/rbt/alg/pinnocchio_bipedal_example

将搭建一个简易的双足模型，并使用Cassie的HZD方法进行控制，最终效果参考

Pinocchio库使用说明

关于双足控制的一些背景

在双足的模型控制领域，我个人将其分为两个流派：简化模型派和全阶模型派。这个观点出自于Jessy Grizzle的这个视频。视频中提到，Jessy不太懂怎么用简化模型方法做控制，因而选择了硬刚动力学控制的方法。其思路非常的传统控制，因而Cassie的控制方法也显得非常地直接。下面给大家简单介绍一下这两个流派。

简化模型派主要以MPC+简化模型为主。有关于MPC方面的具体介绍可以参考世成的这一篇。MPC控制比较有名的应用还是在四足上面（比如MIT Cheetah和后来的一系列四足），一般的做法是把身体简化为一个单刚体，然后把四足机器人的运动控制简化为对单刚体质心的控制。这一步会把四足机器人复杂的动力学模型，简化为基座（单刚体）的动力学模型（状态空间方程）。状态空间方程的A矩阵则是一个12x12的矩阵。简化后，利用该模型，预测机器人未来**N**步的状态，构建优化问题，并进行优化，再取最近一步的支撑力作为当前的控制器输出，这也就是最常用的MPC控制框架。此方法的优点是规避了显式的矩阵求逆，降低了运算成本。（求解库里即便有求逆，也是做过相应优化，速度会比较快）。由于**N**的数值越大，所构建的矩阵规模也越大（比如，优化10步。矩阵规模就会变为120*120），因而，在不同成本的平台上，可以调整N的数值，在成本和效果上做权衡和取舍。缺点则是由于模型过于简化，比如，忽略了腿部本身的质量以及转动惯量，当腿部惯量比较大的时候，控制效果会打折扣。因此，又诞生出了MPC+WBC（WBIC），NMPC等方法，用来解决更复杂的运动控制问题。

以Cassie为代表的全阶模型派则选择了**更暴力**的方法，把机器人的动力学模型（一般是浮动基座模型）直接求解了出来，并将控制的重点由**支撑腿力分配**，转化为**摆动腿的轨迹跟随+其他目标控制量的跟踪**。由于求解出来了动力学模型（主要是M,C,G矩阵），因而可以采用反馈线性化控制的方法，让目标量更好地被跟踪。更重要的是，全阶模型派还**证明了这种控制方法的稳定性**，能通过构建李雅普诺夫函数，证明机器人的行走是能够维持在机器人的零动态(Zero Dynamics)内，或者说是周期性稳定

的状态。 可见，全阶模型控制具备理论的完备性，而非像简化模型一样，丢失了大量信息。 也因此，在面对腿部惯量较大等非线性因素的时候，具备更好的处理能力。 缺点则是需要涉及到若干处大矩阵的求逆，同时对控制的实时性又有要求。

然而，随着各种计算平台算力的提升，以及动力学算法的优化，求解全阶动力学模型的门槛正在逐步下降。近年也渐渐出现一些将全阶模型控制从双足往四足上推广的案例。

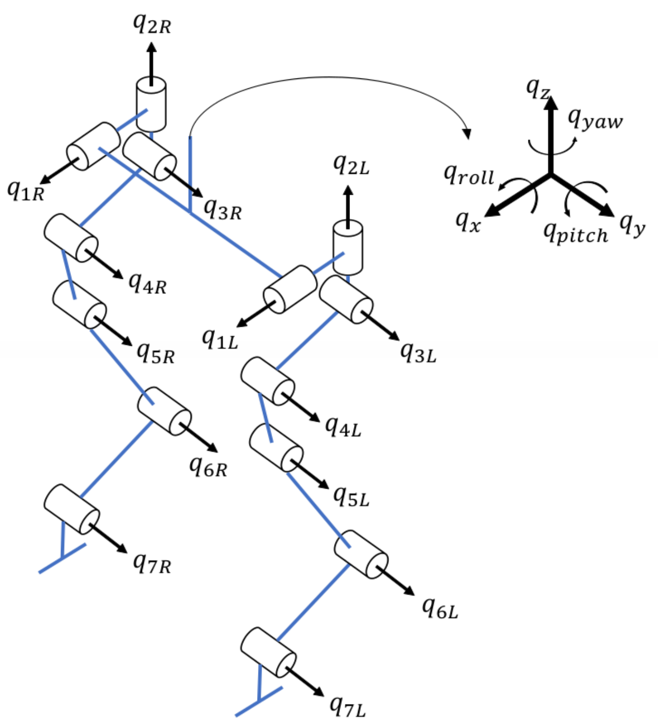
本文会着重介绍Cassie中应用到的全阶模型控制方法，希望能给大家带来一些帮助。

浮动基座模型

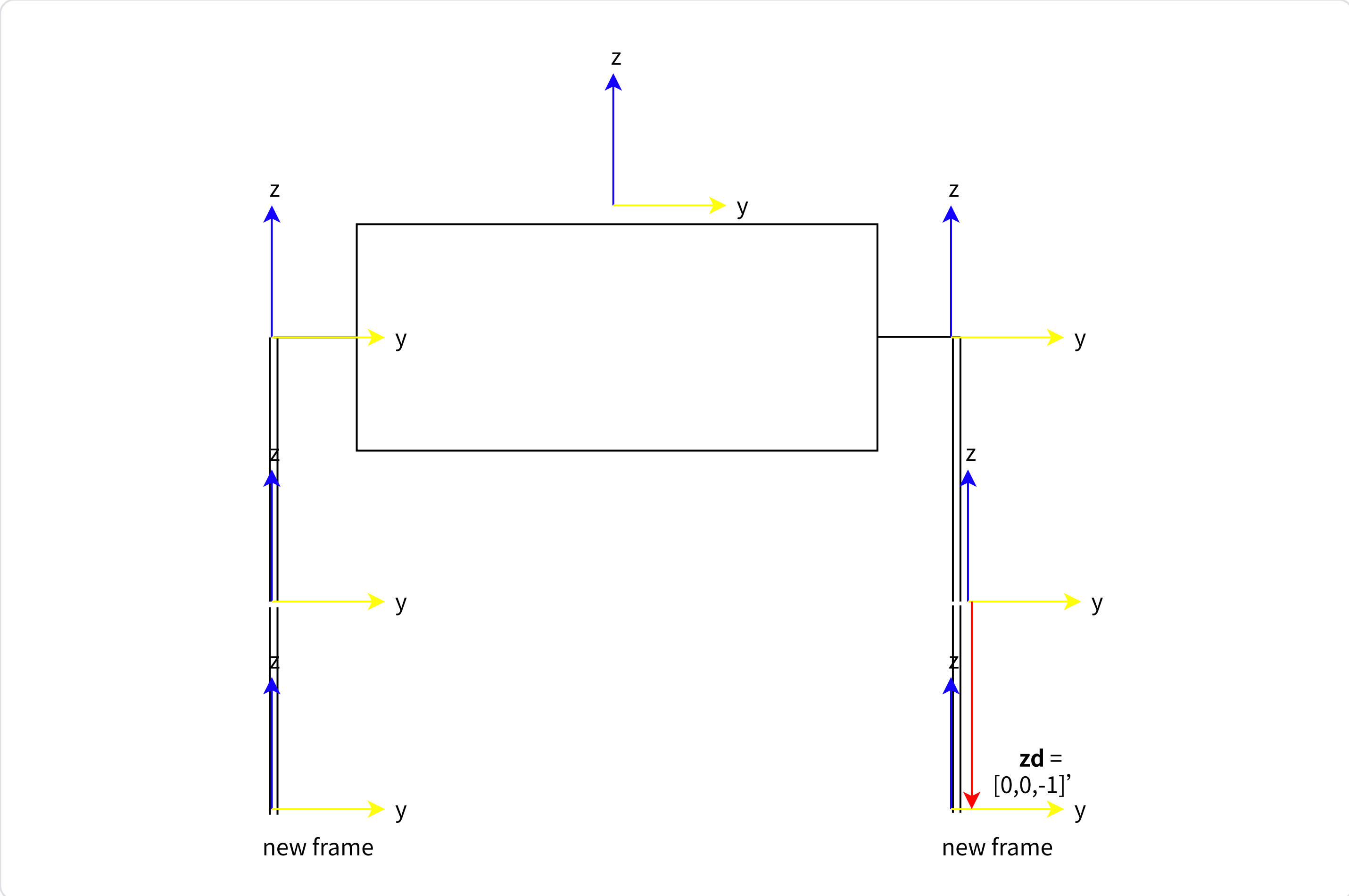
一般的机械臂模型里面，一般使用的是固定基座的动力学模型，即认为机器人的基座是固定的，并由此展开相关的运动学和动力学的推算。

但在足式机器人领域，机器人的基座，也就是身体，在受到地面的反作用力后，是会发生移动的。由此诞生了浮动基座模型，浮动基座模型的含义是，认为基座本身含有6个欠驱动的自由度，即x,y,z,roll,pitch,yaw，因此浮动基座模型比传统的固定基座模型多6个自由度。引入浮动基座模型，能让我们把在设计控制器的时候，把基座本身的运动也一并考虑进去。

下图是Cassie的浮动基座模型示意图，Cassie身上一共有14个自由度，其中包含12个主动驱动自由度和2个欠驱动自由度。再加上6个浮动基座自由度，一共是20个自由度。



下图是easy robot的坐标示意图，一共4个关节，再加上6个浮动基座自由度，一共是10个自由度。



浮动基座的动力学模型可表达为如下**式(1)**：

$$M(q) + C(q,\dot{q}) + G(q) = B\tau + J_c^T F_c$$

$$J_c\ddot{q} + \dot{J}_c\dot{q} = 0$$

固定基座的动力学模型则一般表示为**式(2)**：

$$M(q)\ddot{q} + C(q,\dot{q}) + G(q) = \tau$$

可以看到，比起固定基座模型的动力学方程，浮动基座模型的动力学模型里面包含了 $J_c(q)^T F_c$ 这个外力项，同时还多了一个加速度约束 $J_c\ddot{q} + \dot{J}_c\dot{q} = 0$。正是因为存在外力项和加速度约束，浮动基座才可以在地面反作用力的作用下发生位移和旋转。B矩阵为选择矩阵，用来筛掉欠驱动关节。**注意，如果没有B矩阵的存在，那么在求解的时候，会默认所有的自由度都是全驱的（包括新加的6个自由度），这显然是不对的。**

> 举个例子：如果我们直接用浮动基座的雅格比矩阵，通过末端外力来反算各关节的输出力矩，即直接使用以下公式
>
> $\tau = J_c^T F$
>
> 会发现，转矩的前六项，也就是 $Fx_x, F_y, F_z, \tau_r, \tau_p, \tau_y$ 占比会很大。但这几个关节又是欠驱动的，与实际情况矛盾。
>
> 真正的做法应该是要求要用以下方式求雅格比矩阵：
>
> $J = \dfrac{\partial r_b}{\partial q_r}$

> 其中qr是主动关节的广义坐标。这样构建的雅格比矩阵和固定基座下构建是不一样的，因为末端位置的原点是浮动基座的原点，而不是固定基座的。此时的雅格比矩阵应当满足
>
> $v = J\dot{q_r}$
>
> 之后通过虚功原理，依然可以得到：
>
> $\tau = J^T F$
>
> 相关内容可以参考ETH动力学讲义中关于Quasi-Static Control (准静态控制)部分的内容。

一般还会把这个外力项增广到惯量矩阵M里面：

$$M_e = \begin{bmatrix} M(q) & -J_g^T \\ J_g & 0 \end{bmatrix}, \quad \ddot{q_e} = \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \\ \ddot{r} \\ \ddot{p} \\ \ddot{y} \\ \ddot{q_{l1}} \\ \ddot{q_{l2}} \\ \ddot{q_{r1}} \\ \ddot{q_{r2}} \\ F_x \\ F_z \end{bmatrix}, \quad J_g = \begin{bmatrix} Jc_{st}(0,0,1,10) \\ Jc_{st}(2,0,1,10) \end{bmatrix}, \begin{bmatrix} vx \\ vy \\ vz \\ dr \\ dp \\ dy \end{bmatrix} = Jc_{st} * \dot{q}$$

并同步地增广科氏矩阵C和重力矩阵

$$C_e = \begin{bmatrix} C(q,\dot{q}) \\ 0 \end{bmatrix} \qquad G_e = \begin{bmatrix} G(q) \\ 0 \end{bmatrix} \qquad B_e = \begin{bmatrix} 0 \\ B_e \end{bmatrix}$$

其中， M_e 为惯量矩阵 $M(q)$ 的增广矩阵，上三角部分增广了支撑腿的雅格比矩阵中关于外力Fx和Fz的部分（即 JF_{st} 的第0行和第2行），下三角部分增广了加速度约束。此外，在Cassie的算法里还有个取巧的地方，即认为：

$$J_c\ddot{q} + \dot{J}_c\dot{q} \approx \ J_c\ddot{q} = 0$$

q_e 为广义坐标q的增广后广义坐标，增加了外力项 F_x 和 F_x 。完成增广后，动力学模型就会变成式(2)的形式，即下列**式(3)**：

$$M_e(q)\ddot{q} + C_e(q,\dot{q}) + G_e(q) = B_e\tau$$

可以看到，式(3)已经很接近固定基座的动力学方程了，这意味着一些我们可以套一些常用的机械臂非线性控制算法了。

混合零动态

混合零动态（Hybrid Zero Dynamics）是对零动态（Zero Dynamics）的延伸。零动态的概念则常出现于非线性控制理论中。（SISO和MIMO也有），关于零动态的具体概念可具体参考这篇文章。我们在分析线性系统稳定性的时候，是用零极点的方式去进行分析，零极点也包含了系统的输出具体如何收敛到某一状态的信息。 但在非线性系统，系统输出收敛到稳定状态的动态过程是不唯一的，有的可

能是指数型收敛，有的可能是极限环。因而，产生了零动态这一概念，**它是对系统稳定的动态过程的描述**。

设系统的输出为y，y是一个值或一个向量，零动态的意思则是输出y的输出结果在0附近波动（或者在初始值的一组集合范围内波动）。混合零动态则是混合模型（Hybrid Model）的零动态，双足便是一个很好的例子。在双足模型，存在两个模型，即摆动腿模型和双腿支撑模型，如下图所示：

3) Hybrid Model: The overall model is given as follows:

$$
\Sigma : \left\{ \begin{array}{rl}
D(q)\ddot{q} + H(q,\dot{q}) = Bu + J_{sp}(q)^\top \tau_{\mathrm{sp}} + J_{\mathrm{R}}(q)^\top \lambda & \\
J_{\mathrm{R}}(q)\ddot{q} + \dot{J}_{\mathrm{R}}(q,\dot{q})\dot{q} = 0 & (q;\dot{q}^-) \notin \mathcal{S}_{\mathrm{R}\to\mathrm{L}} \\
\dot{q}^+ = \Delta_{\mathrm{R}\to\mathrm{L}}(\dot{q}^-) & (q;\dot{q}^-) \in \mathcal{S}_{\mathrm{R}\to\mathrm{L}} \\
& \\
D(q)\ddot{q} + H(q,\dot{q}) = Bu + J_{sp}(q)^\top \tau_{\mathrm{sp}} + J_{\mathrm{L}}(q)^\top \lambda & \\
J_{\mathrm{L}}(q)\ddot{q} + \dot{J}_{\mathrm{L}}(q,\dot{q})\dot{q} = 0 & (q;\dot{q}^-) \notin \mathcal{S}_{\mathrm{L}\to\mathrm{R}} \\
\dot{q}^+ = \Delta_{\mathrm{L}\to\mathrm{R}}(\dot{q}^-) & (q;\dot{q}^-) \in \mathcal{S}_{\mathrm{L}\to\mathrm{R}}.
\end{array} \right. \tag{11}
$$

混合零动态只是一种状态，进入到双足的混合零动态，意味着机器人已经进入到其运行的稳态。

反馈线性化控制

反馈线性化控制是一种非线性控制的手段。其基本思路是在保证系统的最小线性相位是稳定的情况下，消除系统非线性项带来的影响。在机械臂控制中，反馈线性化帮助我们让机械臂更好地**跟踪给定的轨迹**。

我们在简介中提到过，在简化模型派中，其运算重点在于支撑腿的力分配，而全阶模型派中，则更加关注摆动腿的末端位置控制。反馈线性化便是为了让摆动腿更好地跟踪给定的轨迹，使得通过LIP模型计算的下一时刻的落足点不会有太大的变动，**从而机器人的运行不会超出零动态的范围**。 如果没有使用反馈线性化（比如使用传统的PID+前馈）,使得摆动腿跟踪的轨迹差距过大，会使得LIP模型实时计算的落足点发生较大的浮动，进而使机器人超出零动态范围。

下面介绍双足机器人中的反馈线性化控制器设计。可以参考这篇文章，在该文章中，尽管其模型是一个平面机器人的模型，但用到的反馈线性化方法和Cassie基本是相同。

我们先定义系统的输出，即我们的目标控制量。定义输出y为：

$$y = h(q) = [\theta, z_{st}, x_{sw}, z_{sw}]^T$$

其中q为原始浮动基座动力学模型的广义坐标，θ 位机器人基座pitch角，z_{st} 为支撑腿的z坐标，x_{sw} 为。关于y和广义坐标之间的雅格比矩阵，由雅格比矩阵的定义，可得

$$J_h = \frac{\partial h(q)}{\partial q}$$

雅格比矩阵应该满足**式(4)**：

$$\dot{y} = J_h \dot{q}$$

$$\ddot{y} = J_h * \ddot{q} + \dot{J}_h \dot{q}$$

我们已经完成了对浮动基座模型的建立，如

$$M_e(q)\ddot{q_e} + C_e(q_e, \dot{q_e}) + G_e(q_e) = B_e \tau$$

变形可得：

$$\ddot{q_e} + M_e(q_e)^{-1}(C_e(q_e, \dot{q_e}) + G_e(q_e)) = M_e(q_e)^{-1} B_e \tau$$

由式(4)，为了能让矩阵的维度对的上，我们给矩阵 J_h 增广两列0向量，或者说右乘一个增广矩阵S，从而我们我们得到

$$J_h S \ddot{q_e} + J_h S M_e(q_e)^{-1}(C_e(q_e, \dot{q_e}) + G_e(q_e)) = J_h S M_e(q_e)^{-1} B_e \tau$$

进而

$$\ddot{y} - \dot{J}_h S \dot{q_e} + J_h S M_e(q_e)^{-1}(C_e(q_e, \dot{q_e}) + G_e(q_e)) = J_h S M_e(q_e)^{-1} B_e \tau$$

令 $D = J_h S M_e^{-1} B_e$，且按照前述，省略掉 $\dot{J}\dot{q}$,则：

$$u = D^{-1}\ddot{y} + D^{-1} J_h S M_e^{-1}(C_e(q, \dot{q}) + G_e(q)) = \alpha(x) + \beta(x) v$$

$$\alpha(x) = D^{-1}(C_e + G_e), \beta(x) = D^{-1}, v = \ddot{y}$$

为了更好地跟踪给定的轨迹，将 v 设定为PD控制率，即

$$v = kp * \Delta y + kd * \Delta \dot{y} + \ddot{y}$$

由此实现了摆动腿末端轨迹的跟踪控制。

我们来具体的看一下Jh的具体表达式，实际上就是去看 $\dot{y}$ 和 $\dot{q}$ 之间的关系。对于y(0)，实际就是q(4)，也就是浮动基座的pitch坐标。对于y(1)，就是支撑腿的的末端z坐标，（这里也可以选择使用支撑腿的腿长，乘一个旋转矩阵即可）。对于y(2)，就是摆动腿的x坐标。对于y(3)，就是摆动腿的z坐标。 由此，Jh的表达式如下。

$$J_h = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & \ldots & 0 \\ 0 & \ldots & 0 & J_{cst}(2,6,1,4) & & & & \\ 0 & \ldots & 0 & J_{csw}(0,6,1,4) & & & & \\ 0 & \ldots & 0 & J_{csw}(2,6,1,4) & & & & \end{bmatrix}$$

其中， J_{cst} 是浮动基座模型下，支撑腿末端的雅格比矩阵， J_{csw} 是摆动腿末端的雅格比矩阵。Jcst(2,6,1,4)的含义是，以Jcst矩阵的第2行第6列为起点，选1行4列。这是因为根据矩阵乘法：

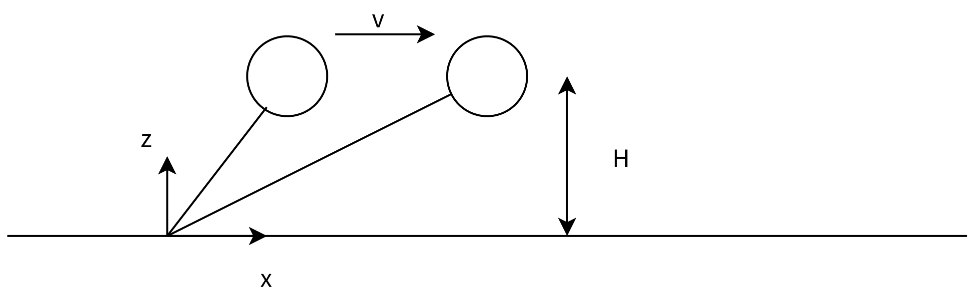
$$v(2) = v_z = J_{cst}(2,6) * \dot{q(6)} + \ J_{cst}(2,7) * q(7) \ + \ J_{cst}(2,8) * q(8) \ + \ J_{cst}(2,9) * q(9)$$

v_z 是支撑腿沿浮动坐标系下沿z轴的速度。 J_{csw} 部分以此类推。

由此，我们得到了一个关于腿部的线性控制器。

LIP模型与末端落点选择

LIP模型，即线性倒立摆模型，描述的是一个**z轴高度不变**、腿长可变的线性倒立摆，如下图所示。LIP模型能根据输入的质心目标速度，机身高度、步频、机身质量等信息，输出下一时刻足端的落点。获得足端落点后，我们就可以规划足端轨迹，并输入到反馈线性化控制中。



注意，上述模型不涉及到动力学，只涉及到运动学的结算。该运动学模型的微分方程为：

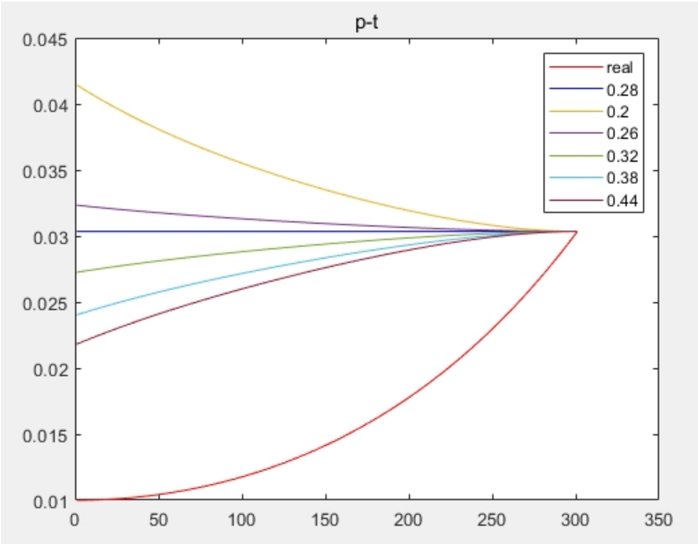
$$\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ g/H & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$$

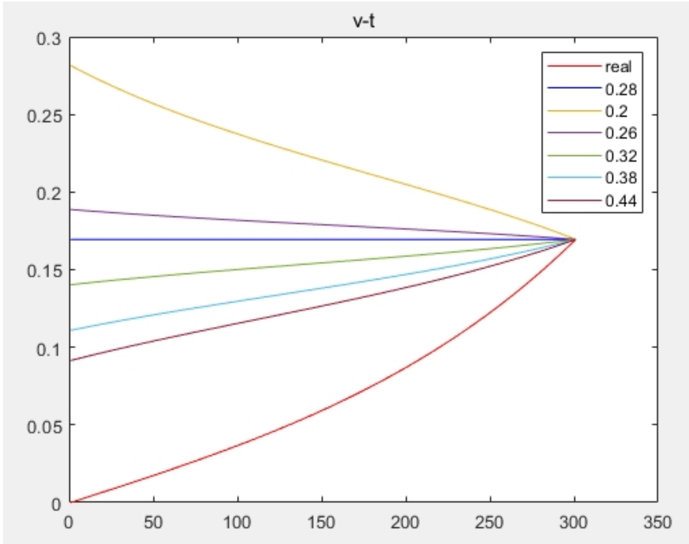
LIP模型描述的是在高度不变的杆下，质心的位置、速度和加速度之间的关系。这里有一个很重要的前提是，高度是一个恒定值，而不是一个时变量。否则上述矩阵就变成了一个时变矩阵，上述微分方程的通解也会变得更加复杂。

上述微分方程的通解为一个带指数函数的形式，**经过离散化后**，会变为以下形式：

$$\begin{bmatrix} x(T) \\ \dot{x}(T) \end{bmatrix} = \begin{bmatrix} cosh(l*(T-t)) & \frac{1}{l}sinh(l(T-t)) \\ lsinh(l(T-t)) & cosh(l(T-t)) \end{bmatrix} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix}$$

该微分方程解的含义是，已知t时的位置和速度，可以预测出T时刻质心的位置和速度。尽管该微分方程的前提假设是H不变，但假定我们的H发生了变化，此时微分方程预测出的质心位置和速度将会发生如下变化，其中0.28m为目标的高度值。





在高度为0.28时其预测到的T时刻的x轴位置和速度均为一条直线。 当输入高度范围从 0.26m到0.44m时，预测到的T时刻位置值会发生大约1cm左右的偏差，速度会发生约0.1m/s的偏差，对实际的控制影响并不大。因而控制器稍微有些跟不上，也有一定的容忍范围。

在Cassie的算法中，计算落足点的方法是根据2T时刻的角动量（质心速度），反算T时刻的落足点。对于T时刻的机器人，2T时刻的质心速度为：

$$v_{k+1} = lsinh(lT)p_k + \ cosh(lT)v_k$$

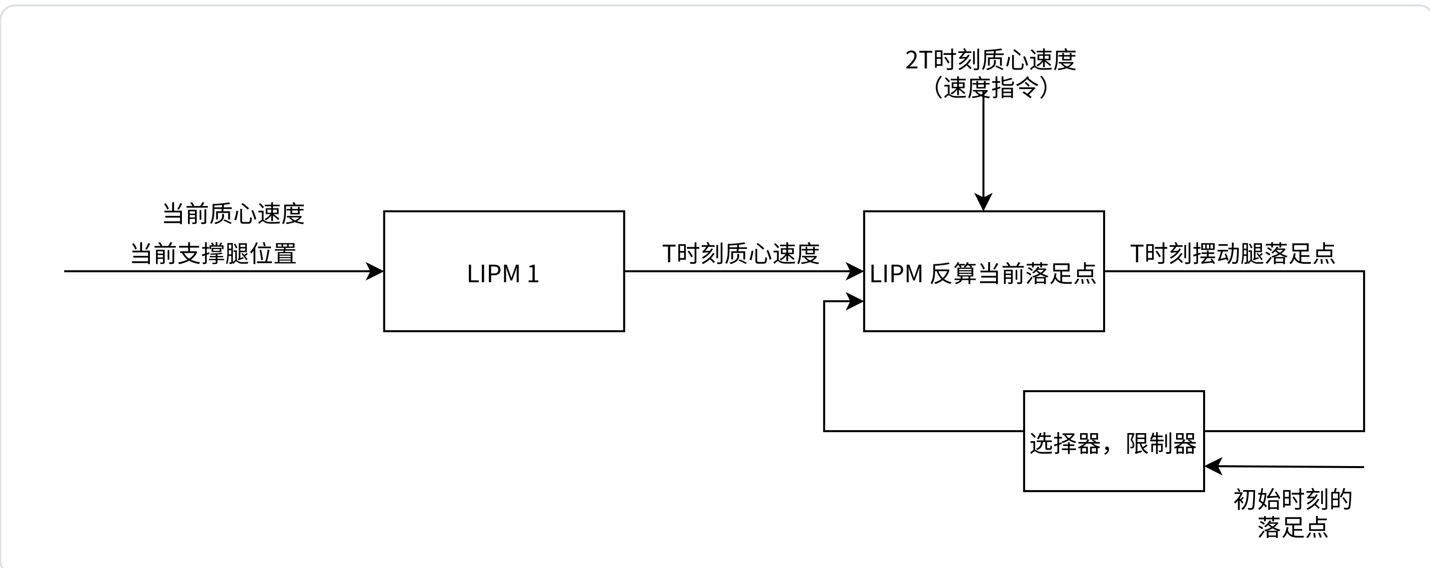
算出的落足点可表示为：

$$p_k = \frac{v_{k+1} - cosh(lT)v_k(t)}{lsinh(lT)}$$

其中 p_k 指的是T时刻摆动腿的落足点。 v_{k+1} 是2T时的质心速度，也是机器人的目标速度。 $v_k(t)$ 指的是机器人摆动相在整个摆动周期过程中预测到的T时刻的质心速度，他满足

$$v_k = lsinh(lT)p_{k-1} + \ cosh(lT)v_{k-1}$$

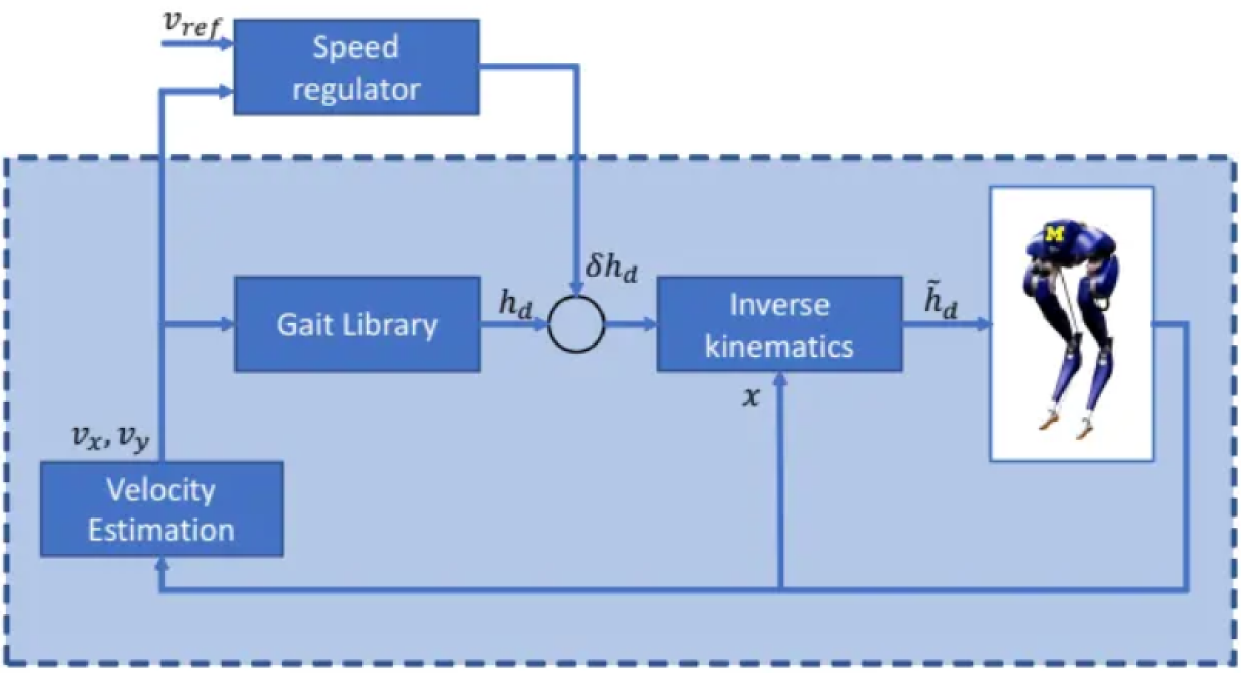
下列框图更直观地表示推算落足点的过程：



当我们拿到了T时刻摆动腿的落足点后，即可在摆动腿的**起始时刻**对摆动轨迹进行规划，如正弦曲线、贝赛尔曲线等。在Cassie中则是集成了一整个gait library。

控制框图

基于以上介绍，基于HZD方法的控制框图如下图所示：



其中，Inverse Dynamics即浮动基座模型和反馈线性化控制部分。Velocity Estimate则是机器人质心速度估计的部分，在easy robot中，质心速度由支撑腿相对于浮动基座坐标原点的速度反算而来。Gait Library则包含了基于LIP模型的落点计算，以及轨迹规划。

关于该控制器的稳定性证明，同样可以参考这篇的3.2章节。需要注意的是，**在该证明中，包含了反馈线性化控制器部分，因而反馈线性化控制、动力学模型的建模，在HZD框架中是不可或缺的存在。**

参考文献：

ETH动力学讲义

比较详细地推导双足中反馈线性化控制方法

Angular Momentum about the Contact Point for Control of Bipedal Locomotion: Validation in a LIP-based Controller.pdf

Feedback Control of a Cassie Bipedal Robot: Walking, Standing, and Riding a Segway.pdf