目录

SRSP

1. Introduction to Rigid Body Dynamics

基于牛顿-欧拉动量守恒定律建立运动方程

结合系统内的约束条件，得到整体系统的运动方程

Concerning the Simulation Model:

1.properties of each individual Rigid Body

2.kinematic coupling between Rigid Bodies

3.additional actors/sensors

Simulator:solvers/integrators to simulate the resulting behavior of the interacting Rigid Bodies over time.

解DAE方程

2.Rigid Body Dynamics Simulation in short

Maximal Coordinates（最大坐标法） vs Generalized Coordinates（广义坐标法）

因为本题目主要围绕碰撞，所以使用Maximal Coordinates。

结构更灵活、适合接触/碰撞仿真，但计算更复杂。

Newtons laws of motion for Rigid Body

$\underline{p}_i = m_i \cdot \underline{v}_i$

$\underline{L}_i = \underline{\Theta}_i \cdot \underline{\omega}_i$

$$\frac{d\underline{p}_i}{dt} = \underline{f}_{\text{ext},i} = m_i \cdot \dot{\underline{v}}_i$$

$$\frac{d\underline{L}_i}{dt} = \underline{\tau}_{\text{ext},i} = \underline{\Theta}_i \cdot \dot{\underline{\omega}}_i + \underline{\omega}_i \times \underline{\Theta}_i \underline{\omega}_i$$

$\Leftrightarrow$

Change of velocity

$$\dot{\underline{v}}_i = \frac{1}{m_i} \cdot \underline{f}_{\text{ext},i}$$

$$\dot{\underline{\omega}}_i = \underline{\Theta}_i^{-1} \cdot \left(\underline{\tau}_{\text{ext},i} - \underline{\omega}_i \times \underline{\Theta}_i \underline{\omega}_i\right)$$

Equation of motion of a system of Rigid Bodies

$$\underbrace{\begin{bmatrix} \dot{\underline{v}}_1 \\ \dot{\underline{\omega}}_1 \\ \vdots \\ \dot{\underline{v}}_n \\ \dot{\underline{\omega}}_n \end{bmatrix}}_{\ddot{\underline{x}}} = \underbrace{\begin{bmatrix} m_1^{-1}\underline{I} & \underline{0} & & \underline{0} & \underline{0} \\ \underline{0} & \underline{\Theta}_1^{-1} & & \underline{0} & \underline{0} \\ & & \ddots & & \\ \underline{0} & \underline{0} & & m_n^{-1}\underline{I} & \underline{0} \\ \underline{0} & \underline{0} & & \underline{0} & \underline{\Theta}_n^{-1} \end{bmatrix}}_{\underline{M}^{-1}} \cdot \underbrace{\begin{bmatrix} \underline{f}_{\text{ext},1} \\ \underline{\tau}_{\text{ext},1} - \underline{\omega}_1 \times \underline{\Theta}_1 \underline{\omega}_1 \\ \vdots \\ \underline{f}_{\text{ext},n} \\ \underline{\tau}_{\text{ext},n} - \underline{\omega}_n \times \underline{\Theta}_n \underline{\omega}_n \end{bmatrix}}_{\underline{f}_{\text{ext}}}$$

计算机进行仿真，需要使用差分离散法去模拟连续的微分方程：

$$\ddot{\underline{x}} = \underline{M}^{-1} \cdot \underline{f}_{\text{ext}}$$

Euler积分：

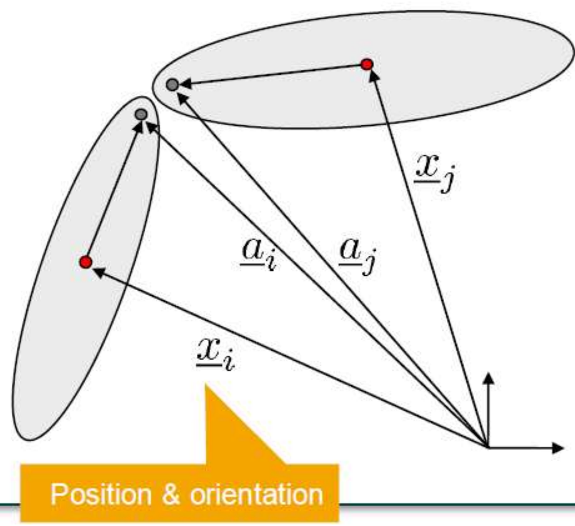
$$\frac{\underline{\dot{x}}(t+dt)-\underline{\dot{x}}(t)}{dt}=\underline{M}^{-1}(t)\cdot\underline{f}_{\text{ext}}(t)$$

$$\underline{\dot{x}}(t+dt)=\underline{\dot{x}}(t)+dt\cdot\underline{M}^{-1}(t)\cdot\underline{f}_{\text{ext}}(t)$$

约束条件：

位置约束：等式约束：

$$C(\underline{x}(t),t)=\underline{0}$$



$$C(\underline{x}(t),t)=\underline{a}_i(t)-\underline{a}_j(t)=\underline{0}$$

之后可以推导一阶导数和二阶导数，也就是所谓的速度和加速度

$$\frac{d}{dt}(C(\underline{x}(t),t))=\underline{0}$$

$$\frac{\partial C}{\partial \underline{x}}\underline{\dot{x}}+\frac{\partial C}{\partial t}=\underline{0}$$

$$\underline{J}\cdot\underline{\dot{x}}+\underbrace{\frac{\partial C}{\partial t}}_{=\underline{0}}=\underline{0}$$

雅可比矩阵：Describes the “direction of the constraints”

$$\underline{J} = \frac{\partial C}{\partial \underline{x}}$$

达朗贝尔原理：约束力不做功

$$\underline{f}_c = \underline{J}^T \cdot \underline{\lambda}$$

系统状态向量 → 限制方程 → 反向推导 →正向推导 → 系统状态向量

3.Recap: Properties and Kinematics of Rigid Bodies

Rigid Body = deformation .

质心： The Centre of Mass

$$\iiint_{\mathbb{B}} \rho(\underline{p}) \cdot \left(\underline{p} - \underline{p}_{\text{cog}}\right) \, dV = 0$$

$$\Leftrightarrow \underline{p}_{cog} = \frac{1}{m} \iiint_{\mathbb{B}} \rho(\underline{p}) \cdot \underline{p} \, dV$$

惯性张量：

$$\underline{L}_B = \underline{\Theta}_{B,\text{cog}} \cdot \underline{\omega}_B$$

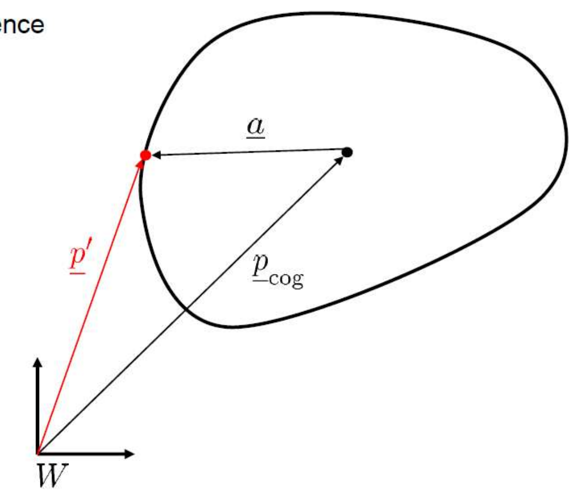
$$^{W}\underline{\Theta}_{cog} = \iiint\limits_{\mathbb{B}} \rho(\underline{p}) \cdot \begin{pmatrix} p_y^2 + p_z^2 & -p_y p_z & -p_x p_z \\ -p_x p_y & p_x^2 + p_z^2 & -p_y p_z \\ -p_x p_z & -p_y p_z & p_x^2 + p_y^2 \end{pmatrix} \, dV$$

对称性 and 正定性

惯性张量的平行轴定理（Parallel Axis Theorem），用于在刚体动力学中将惯性张量从质心位置转换到任意参考点。

The Parallel Axis Theorem allows to change the reference point of the Inertia Tensor.

$$^{W}\underline{\Theta}_{p'} = {}^{W}\underline{\Theta}_{\underline{p}_{\text{cog}}} + m \cdot \begin{pmatrix} a_y^2 + a_z^2 & -a_y a_z & -a_x a_z \\ -a_x a_y & a_x^2 + a_z^2 & -a_y a_z \\ -a_x a_z & -a_y a_z & a_x^2 + a_y^2 \end{pmatrix}$$



如何在不同坐标系之间转换惯性张量（Inertia Tensor）的方向。它是惯性张量旋转变换的公式，并不会改变参考点（如质心），只改变表示的坐标系方向。

$$^{W_2}\underline{\Theta}_{\text{cog}} = {}^{W_2}\underline{R}_{W_1} \cdot {}^{W_1}\underline{\Theta}_{\text{cog}} \cdot \left({}^{W_2}\underline{R}_{W_1}\right)^T$$

四元数可以表示“旋转”或“朝向”，可以和旋转矩阵相互转换。

$$ {}^W\underline{T}_B = \begin{pmatrix} {}^W\underline{R}_B & {}^W\underline{p}_{\text{cog}} \\ \underline{0} & 1 \end{pmatrix} $$

$$ \begin{pmatrix} {}^W\underline{p}_{\text{cog}} \\ \\ {}^W\underline{q}_B \end{pmatrix} $$

四元数的加减法以及求导的相关法则：

The sum of two quaternions x and y:

$$ x + y = (x_0 + y_0) + (x_1 + y_1)i + (x_2 + y_2)j + (x_3 + y_3)k $$

The product of two quaternions x and y:

$$ \begin{aligned} x \cdot y = & \,(x_0y_0 - x_1y_1 - x_2y_2 - x_3y_3) \\ & + (x_0y_1 + x_1y_0 + x_2y_3 - x_3y_2)i \\ & + (x_0y_2 - x_1y_3 + x_2y_0 + x_3y_1)j \\ & + (x_0y_3 + x_1y_2 - x_2y_1 + x_3y_0)k \end{aligned} $$

The inverse quaternion:

$$ q^{-1} = \frac{\bar{q}}{q\bar{q}} $$

The conjugated quaternion:

$$ \bar{q} = q_0 - q_1 i - q_2 j - q_3 k $$

Quaternions are able to represent rotations in three dimensional space:

Each unit quaternion $q \in \mathbb{H}$ can be unambiguously represented in polar representation usin
a $\in \mathbb{H}_{\text{pure}}$ and a polar angle $\alpha \in \mathbb{R}$:

$$ q = \cos\frac{\alpha}{2} + a \cdot \sin\frac{\alpha}{2} $$

The mapping $r_q\colon \mathbb{H} \rightarrow \mathbb{H}$ represents a rotation defined by q, i.e. a rotation about a with angle

$$ r_q(x) = qxq^{-1} \quad \text{with} \quad x = x_1 \cdot i + x_2 \cdot j + x_3 \cdot k \quad , \quad \underline{x} = \big(x_1 \quad x_2 \quad x_3 $$

The concatenation of two rotations r_{q1} and r_{q2} is equivalent to the multiplication of the quate
representing the individual rotations before applying the rotation:

$$ r_{q1} \circ r_{q2} = r_{q1 \cdot q2} $$

The inverse rotation is represented by the inverse quaternion:

$$ r_q^{-1} = r_{q^{-1}} $$

$$\dot{\underline{q}} = \frac{1}{2} \cdot \begin{pmatrix} 0 \\ \underline{\omega} \end{pmatrix} \circ \underline{q}$$

四元数和旋转矩阵的相互转换：

Quaternions are directly related to rotation matrices:

Convert Quaternions to rotation matrix (Euler-Rodrigues formula):

$$R_{q(\underline{a},\alpha)} = \begin{pmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & -2q_0q_3 + 2q_1q_2 & 2q_0q_2 + 2q_1q_3 \\ 2q_0q_3 + 2q_1q_2 & q_0^2 - q_1^2 + q_2^2 - q_3^2 & -2q_0q_1 + 2q_2q_3 \\ -2q_0q_2 + 2q_1q_3 & 2q_0q_1 + 2q_2q_3 & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{pmatrix}$$

For special cases the rotation matrix can be converted into a quaternion (see also [Shuster1993]):

$$q(\underline{a},\alpha) = \frac{\sqrt{1 + r_{11} + r_{22} + r_{33}}}{2} + \frac{r_{32} - r_{23}}{4 \cdot q_0} i + \frac{r_{13} - r_{31}}{4 \cdot q_0} j + \frac{r_{21} - r_{12}}{4 \cdot q_0} k$$

for $\quad 1 + r_{11} + r_{22} + r_{33} > 0$

对于刚体上的某个点，不属于质心，其动量计算公式为：

$$\underline{v}_p = \underline{v}_{\text{cog}} + \underline{\omega} \times \underline{r}$$

$$\underline{\omega}_p = \underline{\omega}_{\text{cog}} = \underline{\omega}$$

$$\dot{\underline{v}}_p = \dot{\underline{v}}_{\text{cog}} + \underline{\omega} \times (\underline{\omega} \times \underline{r}) + \dot{\underline{\omega}} \times \underline{r}$$

向心加速度和欧拉加速度

tip:解法：

$$\dot{\boldsymbol{v}}_p = \frac{d}{dt}(\boldsymbol{v}_{\text{cog}}) + \frac{d}{dt}(\boldsymbol{\omega} \times \boldsymbol{r})$$

$$\Rightarrow \dot{\boldsymbol{v}}_p = \dot{\boldsymbol{v}}_{\text{cog}} + \dot{\boldsymbol{\omega}} \times \boldsymbol{r} + \boldsymbol{\omega} \times \underbrace{\dot{\boldsymbol{r}}}_{=\boldsymbol{\omega} \times \boldsymbol{r}}$$

第三步：解释 $\dot{\boldsymbol{r}} = \boldsymbol{\omega} \times \boldsymbol{r}$

由于点 p 是**刚体内固定点**，即在刚体坐标系下不动：

- 其相对质心的运动完全来自于刚体的旋转
- 因此在世界坐标下的 $\dot{\boldsymbol{r}}$ 表达为：

$$\dot{\boldsymbol{r}} = \boldsymbol{\omega} \times \boldsymbol{r}$$

4.Constraints

约束为两种：Holonomic constraints（完整约束） = 等式约束

Non holonomic constraints = 不等式约束，用于检查是否存在穿模

1.Holonomic constraints

3) Quaternion for orientatio
for Rigid Body 1 ($\underline{q}_1$, he

$${}^{W}\underline{T}_{A_1}(\underline{p}_1,\underline{q}_1) = {}^{W}\underline{T}_{B_1}(\underline{p}_1,\underline{q}_1)\cdot {}^{B_1}\underline{T}_{A_1} = \begin{pmatrix} {}^{W}\underline{R}_{B_1}(\underline{q}_1)\cdot {}^{B_1}\underline{R}_{A_1} & {}^{W}\underline{R}_{B_1}(\underline{q}_1)\cdot \underline{r}_{A_1}+\underline{p}_1 \\ \underline{0} & 1 \end{pmatrix}$$

旋转

平移

平移部分应该约束为0，so:

$$\boxed{C(\underline{x}(t),t) = {}^{W}\underline{R}_{B_1}(\underline{q}_1)\cdot \underline{r}_{A_1} + \underline{p}_1 - {}^{W}\underline{R}_{B_2}(\underline{q}_2)\cdot \underline{r}_{A_2} - \underline{p}_2 = \underline{0} \in \mathbb{R}^3}$$

然后求导，得到速度约束

二次求导，得到加速度约束

5.Simulation of free-floating Rigid Bodies

线动量➕角动量

动量守恒

外力 = 动量对时间求导 = m ✖️ 线速度的导数

外力矩 = 角动量的导数 = （这里存在一个特殊的陀螺项）

$$\underline{f}_{\text{ext}} = \underline{\dot{p}} = m \cdot \underline{\dot{v}}$$

$$\underline{\tau}_{\text{ext}} = \underline{\dot{L}} = \underline{\Theta} \cdot \underline{\dot{\omega}} + \underline{\omega} \times \underline{\Theta} \cdot \underline{\omega}$$

如果外力没有作用在质心，则需要重新计算外力矩：

$$\underline{\tau}'_{\text{ext}} = \underline{r} \times \underline{f}'_{\text{ext}}$$

陀螺项是由于惯性张量的导数引起的，陀螺项会导致刚体的翻滚运动

$$\underline{\tau}_{\text{ext}} = \underline{\dot{L}} = \underline{\Theta} \cdot \underline{\dot{\omega}} + \boxed{\underline{\omega} \times \underline{\Theta} \cdot \underline{\omega}}$$

陀螺项（陀螺力矩）

如果ω与惯性张量Θ的特征向量共线，则陀螺矩为零。刚体绕一个主惯性轴旋转。

如果ω与惯性张量Θ的特征向量不共线，则陀螺矩不为零。刚体翻滚（例如，失去平衡的轮胎）。

仿真设计的循环：

当前系统的状态 （p,q,v,w） → 计算系统状态的微分 （离散） → 最后重新计算变化后的惯性张量

然后回到最初，重新进行循环，计算下一个时间状态。

6.Constraint forces and simulation（约束力和仿真）

1.达朗贝尔定理：约束力不做功

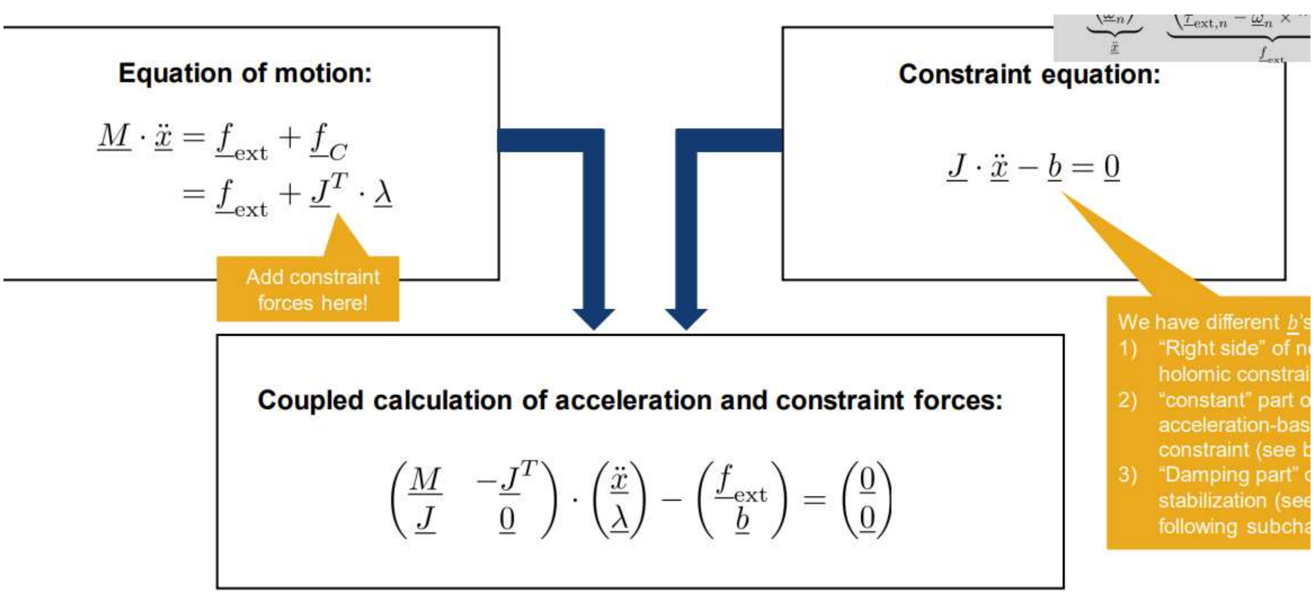
$$
\begin{aligned}
\delta W &= \sum_{i=1}^{N} \underline{f}_{C,i}^{T} \cdot \delta \underline{x}_i \\
&= \sum_{i=1}^{N} \underline{f}_{C,i}^{T} \cdot \dot{\underline{x}}_i \delta t \\
&= \sum_{i=1}^{N} \left(\underline{J}_i^{T} \cdot \underline{\lambda}_i \right)^{T} \cdot \dot{\underline{x}}_i \delta t \\
&= \sum_{i=1}^{N} \underline{\lambda}_i^{T} \cdot \underbrace{\underline{J}_i \cdot \dot{\underline{x}}_i}_{=0} \delta t \quad = 0
\end{aligned}
$$

2.约束力计算：

$$
\underline{f}_c = \underline{J}^T \cdot \underline{\lambda}
$$

计算约束力两种方法

1.方程法：



b 是约束对系统产生的“额外加速度项”，通常由约束随时间变化或非线性速度项引起，是为了保持约束条件恒成立所必须的补偿项。

（通过对位置进行二次求导，可以得出b也就是对加速度的约束项）

2.JMJT-approach

$$\underline{\ddot{x}} = \underline{M}^{-1}\left(\underline{f}_{\text{ext}} + \underline{J}^{T} \cdot \underline{\lambda}\right)$$

使用这个替换

维度更少，计算量会变小

仿真流程（JMJT方法对于受约束刚体）:

系统状态 → 计算雅可比矩阵 → 计算受约束力 →正向推导出积分方程去计算下一个状态的速度和加速度 → 更新惯性张量　　最后重新回到第一个，进行计算下一个的仿真

对于最大化坐标法存在着数据飘逸的问题，所以我们需要用到一些稳定项。

8.Impulse-based simulation of constrained Rigid Bodies

(基于冲量的受约束刚体系统仿真方法)

冲量（Impulse） 是力在短时间内的积分，表示瞬时作用产生的“动量变化”:
J=∫Fdt=Δp=m·Δv.（瞬间发生的碰撞、接触反弹、反作用力)

Impulse-based vs Force-based 方法区别

方法	施力方式	是否逐步积分	是否适合碰撞/瞬时反应	典型用途
Force-based	持续施加力 F	✅ 需要微分方程数值积分	❌ 不适合刚体硬碰撞	常规仿真、控制建模
Impulse-based	瞬时施加冲量 J	❌ 不需要积分过程	✅ 非常适合瞬间碰撞反应	游戏物理、接触响应

9.MLCP（混合线性互补问题）

10.Simulation of constrained Rigid Bodies with friction

Coulomb friction model

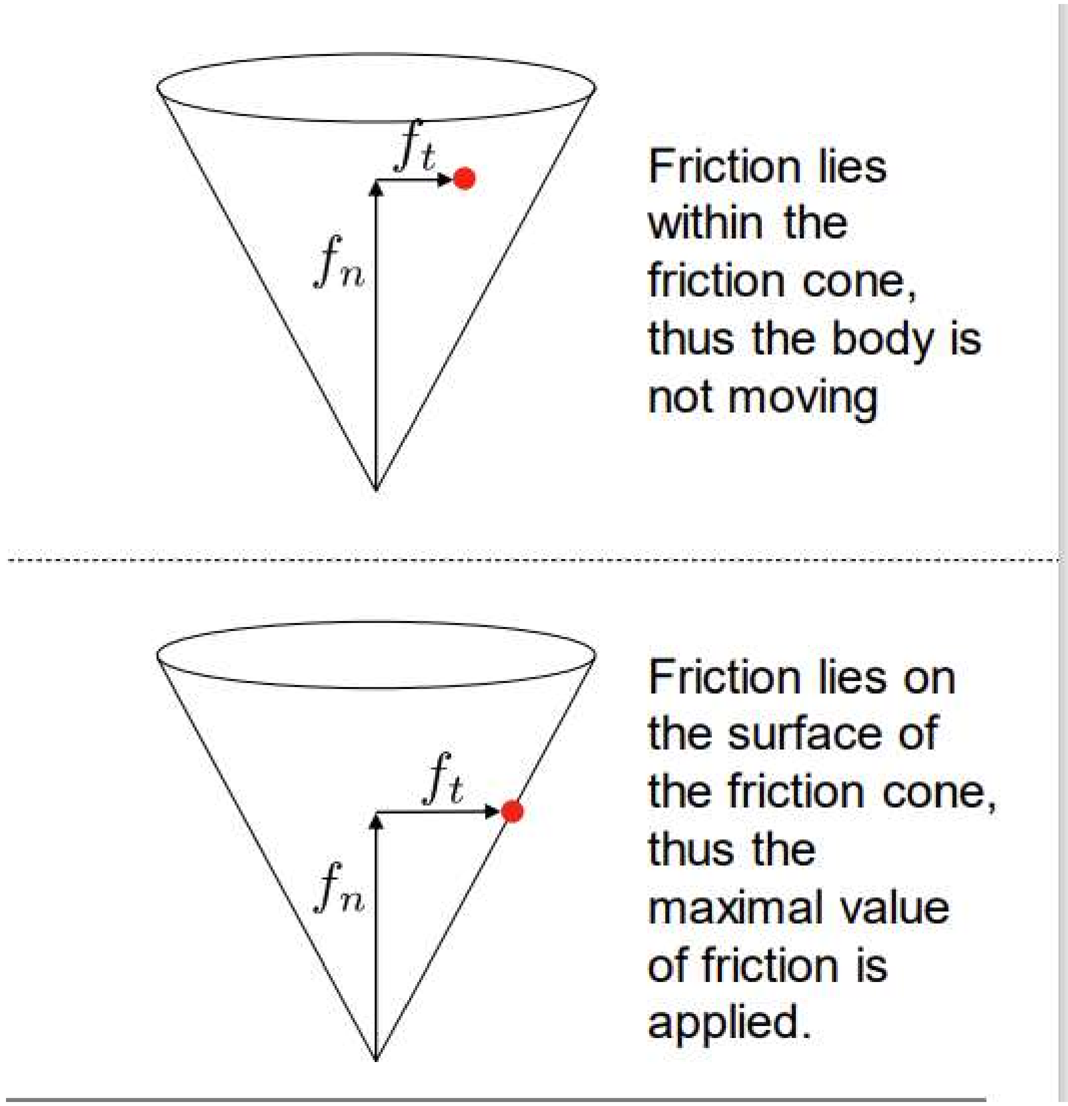
$$f_t \leq \mu \cdot f_n$$

- ft：切向摩擦力（t = tangential）
- fn：法向接触力（n = normal）
- μ：摩擦系数（Coefficient of Friction）

摩擦力大小不能超过“最大静摩擦力” μfn，否则物体就会滑动。

“摩擦锥（Friction Cone）” 的概念：

在 **3D 接触建模中**，摩擦是**向量**，方向可以指向任何切平面方向。因此：这个“模长不超过上限”的几何表示就是一个锥体，叫做 **摩擦锥**。



Friction lies **within** the friction cone, thus the body is **not moving**.
摩擦力位于摩擦锥内，所以物体处于静止状态。

Friction lies **on the surface** of the friction cone, thus the **maximal value** of friction is applied.
摩擦力位于摩擦锥表面，因此摩擦力达到最大值。

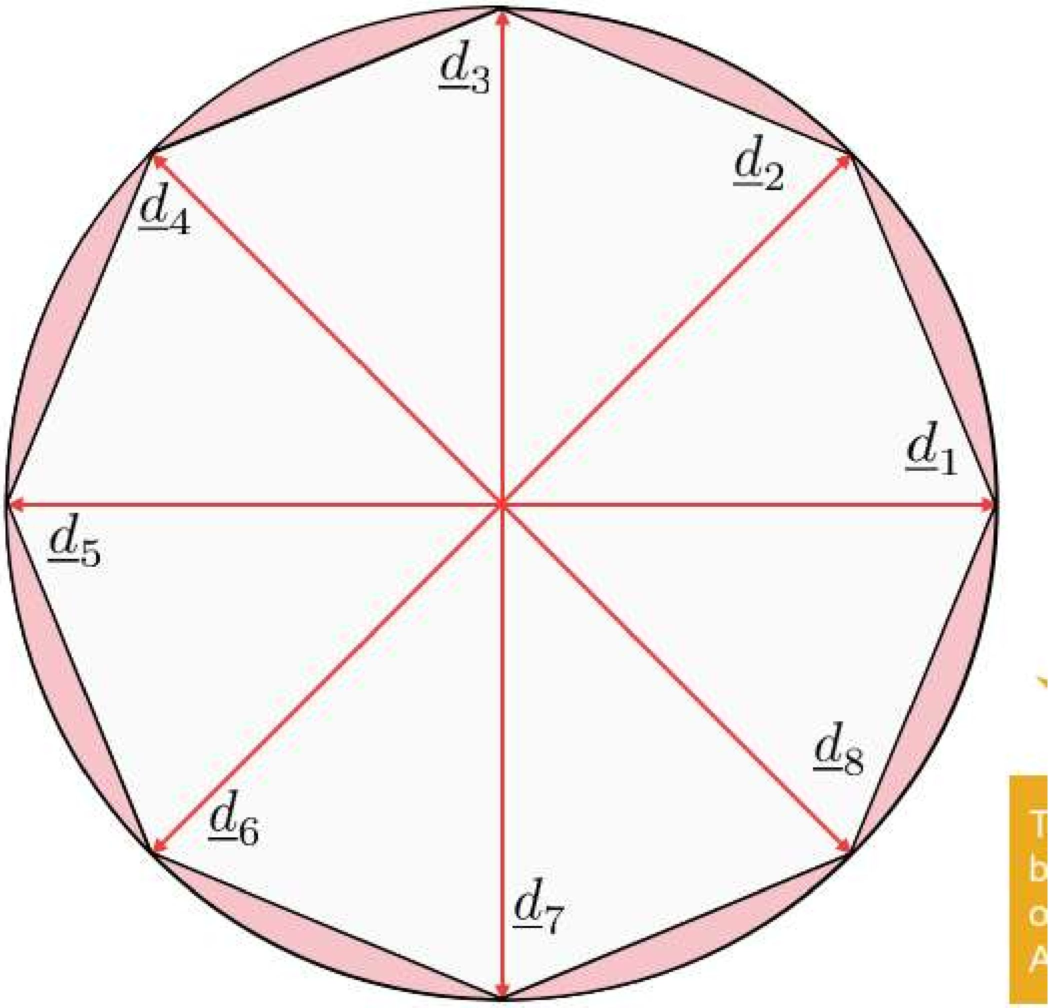
动静摩擦的区别

Friction Cone Approximation – Approximation Error

摩擦锥的近似建模与误差问题

在真实物理中，摩擦力的限制是一个**连续的圆锥形区域（摩擦锥）**，但在计算机中**无法直接处理圆锥体的无限方向**，因此我们通常做如下近似将摩擦锥**离散化成有限数量的基向量d1,d2,...,d8**如图中的红线,实际计算中就只允许摩擦力沿这些方向的**线性组合**.

“摩擦多面体”（Friction Pyramid 或 Friction Cone Approximation）



> **摩擦锥在仿真中通常被近似为摩擦多面体，用有限个方向 di替代无限锥面。方向越多逼近越准，但代价是计算变慢，精度与性能之间需要权衡。**

我的理解：计算机需要把连续的东西进行离散化，所以把摩擦锥这个练习的圆锥模拟成多面体，并用不同的向量之间的线形组合来表示摩擦力的方向。

通过一组离散切向方向 di，并用拉格朗日乘子 λdi 决定每个方向的摩擦力大小，我们就可以逼近任意方向上的摩擦行为，同时便于求解器处理这些不等式约束。

基于约束的摩擦建模：互补约束的建模方法。

要精确的描述摩擦，需要两个互补的约束（complementarity constraints）。

该建模方式通过两个互补条件共同描述摩擦行为：一个约束摩擦大小（是否超过锥体），一个约束摩擦方向（是否滑动），它们之间通过 β·aaux=0 互补性条件连接，实现对静摩擦与动摩擦的统一建模。

✅ 第一条约束：

速度约束 + 摩擦方向

$$J_d \cdot v + \beta \cdot \mathbf{e} = 0$$

- J_d：摩擦方向上的约束 Jacobian（多个离散方向）
- v：相对速度
- β：辅助变量（辅助乘子，用于判定状态）
- $\mathbf{e} = (1, 1, \ldots, 1)^T$：单位向量，长度等于摩擦方向数

✅ 第二条约束：

摩擦大小约束 + 法向力

$$\mu \cdot \lambda_n - \mathbf{e}^T \cdot \lambda_d = a_{\text{aux}} \geq 0$$

- μ：摩擦系数
- λ_n：法向拉格朗日乘子（正向力）
- λ_d：每个方向上摩擦力分量（多个方向）
- a_{aux}：辅助变量，表示当前是否满摩擦力限制

下面为我自己写的计算流程：

摩擦约束整合进动力学系统方程

$\longrightarrow$ Mixed LCP

一. 基础动力学方程（系统动力学）

$$M \cdot \dot{v} = f_{ext} + J^T \cdot \lambda$$

时间离散化，变成冲量形式：

$$M \cdot (v(t+dt) - v(t)) = dt \cdot f_{ext} + dt \cdot J^T \cdot \lambda$$

- λ 包括所有约束的拉格朗日乘子（例如 $\lambda_e, \lambda_n, \lambda_d$）
- $J = [J_e, J_n, J_d]$：对应完整的约束，法向接触约束以及切向摩擦约束

二. 引入约束

J_e 完整约束（关节） $J_e \cdot v(t+dt) = 0$

J_n 法向非穿透约束 $J_n \cdot v(t+dt) = a_n \geq 0$

J_d 摩擦方向约束 $\underbrace{J_d \cdot v(t+dt) + \beta \cdot e = 0}_{\text{用于判断静/动摩擦}}$

三. 摩擦力的互补性建模

1. 滑动状态判断：

$$J_d \cdot v(t+dt) + \beta \cdot e = 0$$

若滑动 $\beta > 0$，表示存在相对速度.

若静止 $\beta = 0$，代表 0 相对滑动.

2. 摩擦力大小限制 + 互补关系

$$\mu \cdot \lambda_n - e^T \cdot \lambda_d = a_{aux} \geq 0$$

$$a_{aux} \cdot \beta = 0$$

(表示总摩擦力不能超过最大静摩擦力，是否滑动由 β 控制)

四、总系统组装为混合 LCP

$$\begin{pmatrix} M & -J_e^T & -J_n^T & -J_d^T & 0 \\ J_e & 0 & 0 & 0 & 0 \\ J_n & 0 & 0 & 0 & 0 \\ J_d & 0 & 0 & 0 & E \\ 0 & 0 & \mu & -E^T & 0 \end{pmatrix} \cdot \begin{pmatrix} v(t+dt) \\ dt \cdot \lambda_e \\ dt \cdot \lambda_n \\ dt \cdot \lambda_d \\ \beta \end{pmatrix} - \begin{pmatrix} M \cdot v(t) + dt \cdot f_{ext} \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$= \begin{pmatrix} 0 \\ 0 \\ a_n \\ 0 \\ a_{aux} \end{pmatrix}$$

五、补充不等式与互补条件。

$$\begin{pmatrix} a_n \\ a_{aux} \end{pmatrix} \geq 0 \qquad \begin{pmatrix} \lambda_n \\ \beta \end{pmatrix} \geq 0 \qquad \left(\begin{pmatrix} a_n \\ a_{aux} \end{pmatrix}\right)^T \cdot \begin{pmatrix} \lambda_n \\ \beta \end{pmatrix} = 0$$

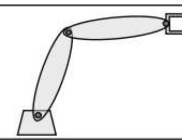
1. $a_n > 0$：没有接触，$\lambda_n = 0$
2. $a_{aux} > 0$：摩擦力未满，$\beta = 0$，即静摩擦
3. 若 $\beta > 0$：正在滑动，则 $a_{aux} = 0$，即已达最大摩擦

11.Generalized vs. maximal coordinates formalism

什么是广义坐标?

广义坐标是对刚体系统进行建模时的核心变量，代表系统的最小独立自由度集，其时间导数构成速度与加速度，广义力是与之对应的驱动力。

Lagrange formalism

Step 1	**Define generalized coordinates**	$q_1, \ldots, q_n$ 
Step 2	**Calculate kinetic energy & potential energy**	$L = E_{\text{kin}}(\underline{q}, \dot{\underline{q}}) - E_{\text{pot}}(\underline{q})$
Step 3	**Calculate Lagrange equation**	$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\underline{q}}}\right) - \frac{\partial L}{\partial \underline{q}} = Q$
Step 4	**Calculate Equation of motion**	$\underline{H}(\underline{q})\,\ddot{\underline{q}} + C(\underline{q}, \dot{\underline{q}}) = \underline{\tau}$

正向动力学（Forward Dynamics）**和**逆向动力学（Inverse Dynamics）：

正向动力学是：已知广义坐标 q、速度 q̇、输入力/力矩 τ，求加速度 q̈
即：你给力，它算运动。

逆向动力学是：已知系统姿态 q、速度 q˙、加速度 q¨，求所需力/力矩 τ
即：你给运动轨迹，它算需要的驱动力。

逆向动力学中的经典算法:**Recursive Newton-Euler Algorithm（递归牛顿欧拉算法）**

🔹 【Step 1】Forward Iteration（前向迭代）

作用：从根部（base）出发，依次计算每个刚体的状态：

1️⃣ 计算每个 link 的位置、速度、加速度

- 使用关节位置 q_i、速度 $\dot{q}_i$、加速度 $\ddot{q}_i$
- 通过变换矩阵和雅可比递推，得到刚体在世界坐标下的：
  - 平动速度 v_i
  - 角速度 ω_i
  - 加速度 a_i

2️⃣ 计算为产生这些加速度所需的合力与合力矩

- 使用刚体方程：

$$F_i = m_i \cdot a_i, \quad \tau_i = I_i \cdot \dot{\omega}_i + \omega_i \times (I_i \cdot \omega_i)$$

🔹 【Step 2】Backward Iteration（反向迭代）

作用：从末端（end-effector）回溯，计算每个关节的驱动力/力矩

3️⃣ 从最末端 link 开始，依次向根部回传：

- 合力、合力矩通过连接点传递
- 在每个关节处，将这些力变换为关节力矩 τ_i：

$$\tau_i = J_i^T \cdot \begin{bmatrix} F_i \\ \tau_i \end{bmatrix}$$

- 也可以理解为"控制该关节所需的力"

合成刚体算法(CRBA)，这是用来高效计算机器人正向动力学中的**关节空间惯性矩阵 H(q)** 的经典方法。

Forward dynamics – Composite Rigid Body algorithm

Goal of the Composite Rigid Body Algorithm

$$\ddot{\underline{q}} = FD\left(\underline{q}, \dot{\underline{q}}, \underline{\tau}\right)$$

i.e. calculate the joint space inertia matrices of the motion equation

?

$$\boxed{\underline{H}\left(\underline{q}\right)}\ddot{\underline{q}} + \boxed{\underline{C}\left(\underline{q}, \dot{\underline{q}}\right)} = \underline{\tau} \qquad \text{with} \qquad \boxed{\underline{C}\left(\underline{q}, \dot{\underline{q}}\right) = ID\left(\underline{q}, \dot{\underline{q}}, \underline{0}\right)}$$

Calculate the generalized accelerations using the motion equation

$$\ddot{\underline{q}} = \underline{H}\left(\underline{q}\right)^{-1}\left(\underline{\tau} - \underline{C}\left(\underline{q}, \dot{\underline{q}}\right)\right)$$

The joint space inertia matrix needs to have full rank, i.e. no redundant configurations are allowed

Composite Rigid Body Algorithm 的作用：

> 🔧 **计算惯性矩阵 H(q)**
> 这是最难、最慢的一项，因此我们用 CRBA 做这件事。

H(q)：关节空间惯性矩阵

- 是正向动力学中加速度对力的“质量”映射
- 与机器人几何结构和质量分布有关
- CRBA 目标就是高效求它

Forward Dynamics – Articulated Body Algorithm（ABA）

关节刚体算法（也叫递归牛顿欧拉正向解法），是机器人动力学中用于正向动力学求解 q¨ 的一种**高效且线性时间复杂度（O(n)）** 的算法。

🔹 Step 1:

从 base 向末端计算：每个 link 的位置、速度和非线性偏置力（bias forces）

- 给定 $q, \dot{q}$
- 递推计算每个 link：
  - 空间速度 v_i
  - 空间加速度（无驱动项）$\dot{v}_i^{\text{bias}}$
  - 重力、科氏、离心项合成的 bias force p_i

🔹 Step 2:

从末端向 base 反向递推：每个 link 的关节惯性 & 偏置力（Articulated Inertia & Bias Force）

- 对于每个关节：
  - 计算**组合后的刚体惯性** $\hat{I}_i$
  - 计算**组合后的偏置项**（考虑连接子 link 的影响）：

$$\hat{p}_i = p_i + \text{变换子节点传来的反力}$$

🔹 Step 3:

从 base 向末端再递推：计算每个关节的加速度

使用下面形式递推解出 $\ddot{q}_i$：

$$\ddot{q}_i = \frac{1}{\hat{a}_i}\left(\tau_i - \hat{b}_i\right)$$

其中：

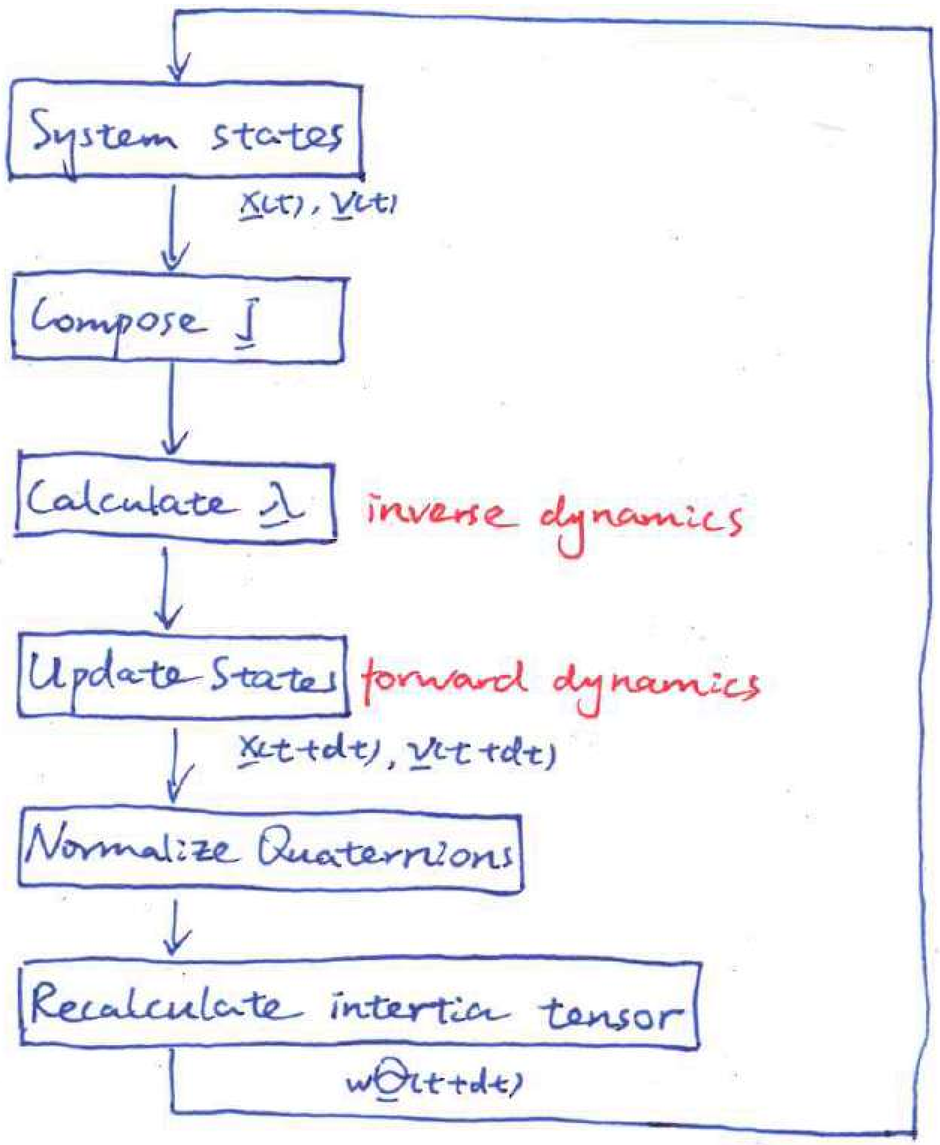
- $\hat{a}_i$：关节的有效惯量
- $\hat{b}_i$：非线性偏置项（来自 bias force 和父节点加速度）

什么是 Maximal Coordinates（最大坐标）：

最大坐标法是**分别为每一个刚体单独定义其完整的位姿（位置 + 姿态）**，而不是只用最小自由度数。

个人总结

纯刚体（无约束）


System states
X(t), V(t)
Compose I
Calculate λ
inverse dynamics
Update States
forward dynamics
X(t+dt), V(t+dt)
Normalize Quaternions
Recalculate intertia tensor
wΘ(t+dt)

🟪 5. Normalize Quaternions

📌 **为什么?**

- 四元数表示旋转，但会在数值积分中逐渐**漂离单位模**
- 非单位模四元数不再表示纯粹的旋转

🔣 **操作:**

$$\mathbf{q} \leftarrow \frac{\mathbf{q}}{\|\mathbf{q}\|}$$

✅ **说明:**

- 每次更新完四元数后都要归一化以维持稳定性

题目理解：

Implementation of an Accelerated Projected Gradient Descent Solver for Multibody Dynamics Simulation

实现一种新的数值求解方法（APGD），用于加速仿真中刚体接触 + 摩擦 + 约束的问题。

在**多体动力学仿真**中（比如机器人、车辆、卫星），你会经常碰到：

- 刚体之间的**接触**（例如轮子接地）
- 接触产生的**摩擦力**
- **约束**（如铰链、轨道）
- 这些都是非光滑动力学（non-smooth dynamics）问题

这些问题数学上形成了一个非常难求解的非线性系统，特别是：

> **NSC/CCP 问题**：非光滑接触问题 / 锥互补问题（Cone Complementarity Problem）

什么是 APGD?

APGD = Accelerated Projected Gradient Descent

- 是一种数值优化方法

- 基于 Nesterov 加速梯度法
- 适合求解受限凸优化问题（如接触力、摩擦力的计算）

优点：

- 比 Gauss-Seidel 更快收敛
- 可并行化
- 更适合大规模系统仿真

在 C++ 多体仿真平台中集成 APGD 方法，并与 Gauss-Seidel 方法比较，提升计算效率。