Last Time:

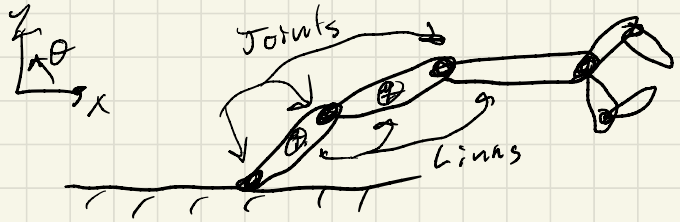
- Discrete Legendre Transform
- DEL with Constraints
- DEL with non-conservative forces

Today:

- Fixed-Base Manipulators
- Floating-Base Systems
- Least-Action with 3D Rotations

Fixed-Base Manipulators:

- Classic serial/open-chain manipulators (e.g. industrial robots (arms)).



- Note (for now) we won't allow any joints that form closed loops.

- For systems like this, we can easily compute the position + orientation of each link in the world frame from the joint angle. Called "forward kinematics".

- In 2D:

link orientation in world frame → $\theta_1 = q_1$

r_1 = link CoM position in World frame

$$r_1 = \underbrace{\begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix}}_{R(\theta)} a_1$$

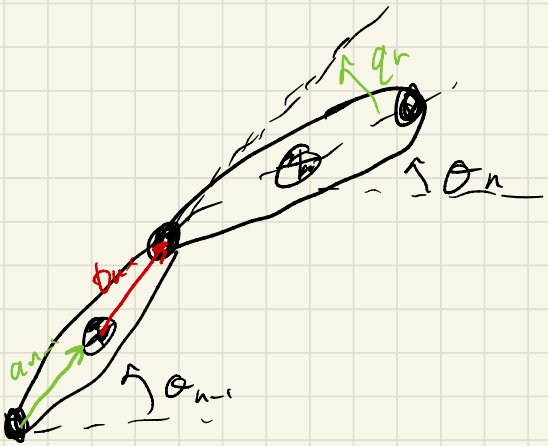
a_1 = vector from joint 1 to CoM of link 1

for $k = 2:N$

$$\theta_n = \theta_{n-1} + q_n$$

$$r_n = r_{n-1} + R_{n-1} b_{n-1} + R(\theta_n) a_n$$

b_{n-1} = vector from CoM of link $n-1$ to joint n



- In 3D:

$$ {}^{N}R_1^{B_1} = S_1(q_1) $$

$$ {}^{N}r_1 = {}^{N}R_1^{B} a_1 $$

$$ R_n = R_{n-1} S_n(q_n) $$

$$ r_n = r_{n-1} + R_{n-1} b_{n-1} + R_n a_n $$

- Let's focus on 2D for now. Let's write kinematics as a function:

$$ \begin{bmatrix} r_1 \\ \theta_1 \\ r_2 \\ \theta_2 \\ \vdots \\ r_n \\ \theta_n \end{bmatrix} = k(q) $$

$q \in \mathbb{R}^n$ vector of joint angles

- From this we can compute world-frame linear and angular velocity of each link:

$$ \frac{d}{dt} \begin{bmatrix} r_1 \\ \theta_1 \\ \vdots \\ r_n \\ \theta_n \end{bmatrix} = \frac{\partial k(q)}{\partial q} \dot{q} $$

$K(q) \in \mathbb{R}^{3n \times n}$

"Kinematic Jacobian"

- Given mass and inertia of each link, we can calculate kinetic energy:

$$T = \sum_{n=1}^{n} \frac{1}{2} m_n \dot{r}_n^T \dot{r}_n + \frac{1}{2} J_n \dot{\theta}_n^2$$

$$= \frac{1}{2} \begin{bmatrix} \dot{r}_1 \\ \dot{\theta}_1 \\ \vdots \\ \dot{r}_n \\ \dot{\theta}_n \end{bmatrix}^T \underbrace{\begin{bmatrix} m_1 I & & & 0 \\ & J_1 & & \\ & & \ddots & \\ 0 & & & m_n I \\ & & & & J_n \end{bmatrix}}_{\bar{M}} \begin{bmatrix} \dot{r}_1 \\ \dot{\theta}_1 \\ \vdots \\ \dot{r}_n \\ \dot{\theta}_n \end{bmatrix}$$

$$= \frac{1}{2} \dot{q}^T \underbrace{K(q)^T \bar{M} K(q)}_{M(q)} \dot{q} = \frac{1}{2} \dot{q}^T M(q) \dot{q}$$

- Similarly, we can compute $U(q)$ and $L(q, \dot{q})$

* Example: Forced Double Pendulum

  - Joint torques are easy in this setup
  - External forces applied to links take a little more work

- We can get this from the Lagrange-D'Alembert principle:

$$\frac{\partial}{\partial q(t)} \left[\int_{t_0}^{t_f} L(q, \dot{q}) \, dt + \int_{t_0}^{t_f} \underbrace{F(t)^T y(t)}_{\text{defined w.r.t. some coordinates } y} \, dt \right]$$

- If we take the variational derivative:

$$\int_{t_0}^{t_f} \left[\frac{\partial L}{\partial q} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right)\right]\delta q \, dt + \int_{t_0}^{t_f} F(t)^T \underbrace{\frac{\partial y}{\partial q}}_{Y(q)} \delta q \, dt$$

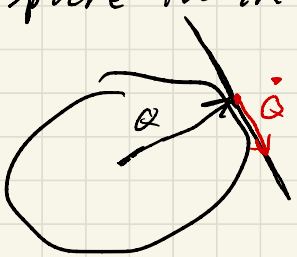
$$\Rightarrow M(q)\ddot{q} + C(q,\dot{q}) + G(q) = Y(q)^T F$$

* Example: double pendulum with external force

- To make this work in 3D we need some quaternion tricks.

Differentiating with Quaternions!

- Geometrically, a unit quaternion is confined to the unit sphere in $\mathbb{R}^4$!



- Intuitively, derivatives of q live in local 3D tangent plane, just like ω:

$$\dot{Q} = \frac{1}{2} Q * \hat{\omega} = \frac{1}{2} L(Q) H \omega$$

$\hat{\omega} = \begin{bmatrix} 0 \\ \omega \end{bmatrix}$

Left quaternion multiplication operator

- I want a 3D gradient of $f(Q) : \mathbb{H} \to \mathbb{R}$
- We can parameterize a tiny rotation with an axis-angle vector ϕ:

$$\Delta Q = \begin{bmatrix} \cos(\|\Delta\phi\|) \\ \frac{\Delta\phi}{\|\Delta\phi\|}\sin(\|\Delta\phi\|) \end{bmatrix} \approx \begin{bmatrix} 1 \\ \Delta\phi \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \Delta\phi \end{bmatrix}$$

- Now apply this to $f(Q)$:

$$f + \Delta f = f(Q * \Delta Q) \approx f\left(Q * \begin{bmatrix} 1 \\ \Delta\phi \end{bmatrix}\right) = f(Q + Q * \hat{\Delta\phi})$$

$$\approx f(Q) + \frac{\partial f}{\partial Q} L(Q) H \Delta\phi$$

$$\Rightarrow \Delta f = \frac{\partial f}{\partial Q} \underbrace{L(Q) H}_{G(Q)} \Delta\phi$$

$G(Q)$ "Attitude Jacobian"

$$\Rightarrow \frac{\partial f}{\partial \phi} = \frac{\partial f}{\partial Q} G(Q)$$

- Whenever we want to diff a function w.r.t. a quaternion input, take the standard Jacobian and right multiply by $G(Q)$
- We can play a similar trick on functions that output a quaternion:

$$Q = f(x) : \mathbb{R}^n \to \mathbb{H}$$

$$Q' = Q * \Delta Q \approx Q + Q * \hat{\Delta\phi} = Q + L(Q) H \Delta\phi$$

$$\simeq f(x+\Delta x) \simeq f(x) + \frac{\partial f}{\partial x}\Delta x$$

$$\Rightarrow \underbrace{L(Q) H \Delta\phi}_{\text{orthogonal} \Rightarrow L^T = L^{-1}} = \frac{\partial f}{\partial x}\Delta x$$

$$\Rightarrow H\Delta\phi = L(Q)^T \frac{\partial f}{\partial x}\Delta x$$

$$\Rightarrow \Delta\phi = \underbrace{H^T L^T(Q)}_{G^T(Q)} \frac{\partial f}{\partial x}\Delta x$$

- Whenever we want to diff a function with a quaternion output, take the standard Jacobian and left multiply by $G^T(q)$.
- We can now compute kinetic energy of a 3D mechanism.
- Assume our kinematics function has the form:

$$\begin{bmatrix} {}^N r_1 \\ {}^N Q_1^B \\ \vdots \\ r_n \\ Q_n \end{bmatrix} = k(q)$$

- Differentiate and apply $G^T(q)$ to the output!

$$\begin{bmatrix} \delta r_1 \\ \delta \phi_1 \\ \vdots \\ \delta r_n \\ \delta \phi_n \end{bmatrix} = \underbrace{\begin{bmatrix} I & & & 0 \\ & G^T(q_1) & & \\ & & \ddots & \\ 0 & & & I \\ & & & & G^T(q_n) \end{bmatrix}}_{\bar{G}^T(q)} \frac{\partial K}{\partial q} \delta q$$

$$\Rightarrow \begin{bmatrix} v_1 \\ \omega_1 \\ \vdots \\ v_n \\ \omega_n \end{bmatrix} = \bar{G}^T(q) K(q) \dot{q}$$

- Now we can compute kinetic energy:

$$T = \frac{1}{2} \dot{q}^T \underbrace{K^T(q) \bar{G}(q) \bar{M} \bar{G}^T(q) K(q)}_{M(q)} \dot{q}$$

$$\bar{M} = \begin{bmatrix} m_1 I & & & 0 \\ & J_1 & & \\ & & \ddots & \\ 0 & & & m_n I \\ & & & & J_n \end{bmatrix}$$

J_n is the 3×3 inertia matrix of link K.