**KINEMATICS** : Describe motion without considering forces

DIRECT : $r = f(q)$ → Numerical / → Differential

## Direct Kinematics

Homogeneous Transform : GOAL $_0T_e$ $\quad T = \begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix}$

↳ (1) Describe pose of a frame wrt another

(2) Transform the representation of a geometric vector from a frame to another

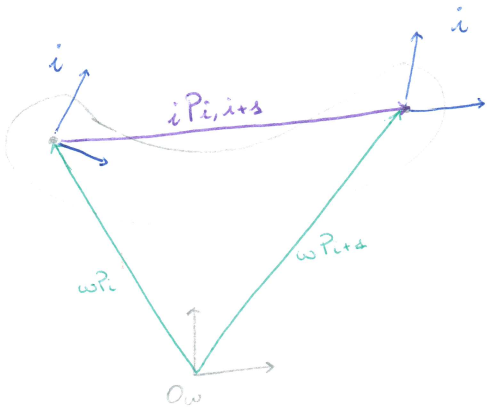
$^AP = {}^AP_{B} + {}^AR_B{}^BP$

## Direct Differential Kinematics

Express end-effector motion velocity as a function of joint velocity

$$\dot r = \frac{\partial f(q)}{\partial q}\dot q = J_A(q)\dot q \quad \text{JOINT SPACE} \to \text{TASK SPACE}$$

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = J(q)\dot q \quad \text{JOINT SPACE} \to \text{CARTESIAN SPACE}$$




LINEAR VELOCITY OF LINK $i+1$ IN TERMS OF LINEAR/ANGULAR OF $i$

Position: $_wP_{i+1} = {}_wP_i + {}_wR_i\,{}_iP_{i,i+1}$

$\frac{d}{dt}$ ↳ $\underbrace{{}_w\dot P_{i+1}}_{{}_wv_{i+1}} = \underbrace{{}_w\dot P_i}_{{}_wv_i} + \underbrace{{}_w\dot R_i\,{}_iP_{i,i+1}}_{{}_w\omega_i \times {}_wP_{i,i+1}} + \underbrace{{}_wR_i\,{}_i\dot P_{i,i+1}}_{{}_wv_{i,i+1}}$

$$\boxed{{}_wv_{i+1} = {}_wv_i + {}_w\omega_i \times {}_wP_{i,i+1} + {}_wv_{i,i+1}}$$

ANGULAR VELOCITY

composition of rot. matrix : ${}_wR_{i+1} = {}_wR_i\,{}_iR_{i+1}$

$\frac{d}{dt}$ ↳ $S({}_w\omega_{i+1})\,{}_wR_{i+1} = S({}_w\omega_i)\,{}_wR_i\,{}_iR_{i+1} + {}_iR_{i+1}\,S({}_i\omega_{i,i+1})\,{}_wR_i$

POST-MULTIPLY by ${}_{i+1}R_w$ ↳ $S({}_w\omega_{i+1}) = S({}_w\omega_i) + S({}_w\omega_{i,i+1})$

$$\boxed{{}_w\omega_{i+1} = {}_w\omega_i + {}_w\omega_{i,i+1}}$$

VELOCITY OF END-EFFECTOR (GOAL) $\begin{bmatrix} {}_wv_e \\ {}_w\omega_e \end{bmatrix} = \begin{bmatrix} \Sigma\,{}_wv_i \\ \Sigma\,{}_w\omega_i \end{bmatrix} = \begin{bmatrix} J_P(q) \\ J_O(q) \end{bmatrix}\dot q$

- PRISMATIC JOINT ATTACHED TO EE : ${}_wv_{i,i+1} = \dot d_{i+1} z_{i+1}$ $\quad J_{P_{i+1}} = z_{i+1}$
- REVOLUTE JOINT ATTACHED TO EE : $J_{P_{i+1}} = z_{i+1} \times ({}_wP_e - {}_wP_{i+1})$

$$J = \begin{bmatrix} J_{P_i} \\ J_{O_i} \end{bmatrix} = \begin{cases} \begin{bmatrix} z_i \\ 0 \end{bmatrix} & \text{Prismatic} \\ \begin{bmatrix} z_i \times ({}_wP_e - {}_wP_{i+1}) \\ z_i \end{bmatrix} & \text{Revolute} \end{cases}$$

INVERSE . find $q$ from EE → SYMBOLIC (geometric intuition) / → NUMERICAL

- Multiple solutions
- $\infty$ solutions (redundant robots)
- No solutions

## SYMBOLIC IK

(1) write DK $\begin{cases} P_x = \\ P_y = \end{cases}$ equations of coordinates of EE

(2) compute IK → find $q$ in function of $P_x$ and $P_y$

## NUMERICAL IK

INPUT : $P_e^d$ $\quad$ OUTPUT : $q$

(1) sequence of $q_i$

(2) compute error $\quad e_i = P_e^d - P_e^i(q_i)$

(3) Make $e_i \to 0$ as $i \to \infty$

$$q^* = \underset{q}{\operatorname{argmin}} \underbrace{\tfrac{1}{2}\|e(q)\|^2}_{C(q)} \to \text{NON-CONVEX}$$

**GAUSS-NEWTON APPROACH**

1. Start with initial guess $\bar q_0$
2. compute linear approximation of problem
3. Solve it to find a new guess
4. Iterate until convergence (error is small enough)

$r(q) = r(\bar q) + J^T J\Delta q = 0$

$J^Te + J^TJ\Delta q = 0 \Rightarrow J^TJ\Delta q = -J^Te$

↳ $\Delta q = -\underbrace{(J^TJ)^{-1}J^T}_{J^\#} e(\bar q)$ $\quad$ NEWTON STEP

USING DAMPED PSEUDO-INVERSE : $-J^T\lambda I$ → make $J^TJ$ always invertible / → make $(J^TJ)^{-1}$ pos. def. → $\Delta q$ is descent direction for $C(q)$ $\forall q$

LINE SEARCH TO FIND LEARNING RATE $\alpha$ :

- $\bar q_{test} = \bar q_i + \alpha\Delta q_i$
- reduction $= \|e(\bar q_i)\| - \|e(\bar q_{test})\|$
- if ![ (reduction > 0) ∨ (reduction > $\gamma\alpha\|e\|$) ]

  $\alpha = \beta\alpha$

  $\bar q_{test} = \bar q_i + \alpha\Delta q_i$

  reduction $= \|e(\bar q_i)\| - \|e(\bar q_{test})\|$
- RETURN $\bar q_{i+1} = \bar q_{test}$

## Redundancy

More DoF than you need. $\infty$ solutions. $J_{FAT}$.

Null space ($N(J)$): $\exists$ joint space velocity $\neq 0$ that doesn't effect EE position.

Row space ($Row(J)$): subspace spanned by the rows of J. $J^\#$ maps always onto the row space.

RANK NULLITY THEOREM
- $\dim(Row(J)) + \dim(N(J)) = n$ (domain)
- $\dim(R(J)) + \dim(N(J^\#)) = m$ (codomain)

Redundant robots can do a secondary task without affects EE velocity:

NULL-SPACE METHOD: $\dot{q} = \underbrace{J^\# v}_{Row(J)} + \underbrace{[I - J^\# J]}_{N(J) \rightarrow \text{secondary task}} \dot{q}_0$

$$\begin{bmatrix} J_{FAT} \rightarrow J^\# = J^T(J^TJ)^{-1} \\ J_{SKINNY} \rightarrow J^\# = (J^TJ)^{-1}J^T \end{bmatrix}$$

SKINNY $\rightarrow$ Least square $\rightarrow$ $q^* = \underset{\dot{q}}{\operatorname{argmin}} \frac{1}{2}\|J\dot{q} - v\|$ $\quad \dot{q}^* = J^\#_{SKINNY} v$

FAT $\rightarrow$ Norm minimization
- $\rightarrow$ full rank $\rightarrow$ $\dot{q} = \underset{\dot{q}}{\operatorname{argmin}} \|\dot{q}\|^2$ s.t. $J\dot{q} = v$
- $\rightarrow$ NOT full rank $\rightarrow$ $\dot{q} = \underset{q \in S}{\operatorname{argmin}} \|\dot{q}\|^2$ $\quad S = \{q \in \mathbb{R}^n : \|J\dot{q} - v\| \text{ is minimum}\}$

GENERALIZED PSEUDO-INVERSE: $A^\#_W = WA^T(A_WA^T)^{-1}$

## Singularity

Less mobility than you need

Singular values of A: $\sigma_i = \sqrt{\lambda_i}$ $\quad \lambda$: eigenvalues of $A^TA$

In a singular configuration $\bar{q}_s$:
(1) EE mobility loss
(2) Possibly $\infty$ solutions to IK
(3) Jacobian loses rank $\rho(J) < m$
  $\rightarrow \rho(J) + \dim(N(J)) = ○ \rightarrow$ null space increase
(4) Big $\dot{q}$ to achieve small EE velocities

SINGULAR CONFIGURATION
- $\rightarrow$ redundant robots $\det(J(\bar{q}_s))J(\bar{q}_s)^T = 0$
- $\rightarrow$ NON-redundant robots $\det(J(\bar{q}_s)) = 0$

TO COMPUTE JOINT VELOCITY FROM TASK VELOCITY $\rightarrow$ $\dot{q} = J^{-1}(q)v$ $\rightarrow$ singular/redundant J is not invertible

$\rightarrow$ use $J^\#$, $J^\# v$ finds the solution $\dot{q}$ closest to v.

DAMPING LEAST SQUARE: $\min_{\dot{q}} \underbrace{\frac{1}{2}\|J\dot{q} - v\|}_{\text{least square}} + \underbrace{\frac{\lambda}{2}\|\dot{q}\|^2}_{\text{Damped}}$
- $\rightarrow$ Always pos. def.
- $\rightarrow$ small joint velocity

# STATICS

Equilibrium: we have forces but not motion

PRINCIPLE OF VIRTUAL WORK: $\underbrace{\tau^T dq}_{\text{work of joints}} - \underbrace{F^T\left(\frac{dP}{d\Phi}\right)}_{\text{work of end effector}} = \tau^T \underbrace{\delta q}_{\text{virtual displacement}} - F^T J dq = 0 \Rightarrow \underbrace{\tau}_{\text{forces of joints}} = J^T \underbrace{F}_{\text{forces on EE}}$

## $J^T$ maps $F$ to $\tau$

$R(J^T)$ = Joint Torques that can balance end-effector forces

$N(J^T)$ = Forces that don't need to be balanced (they're balanced by reaction forces of the structure)

### NON-REDUNDANT MANIPULATORS

1.1 $\text{Rank}(J/J^T) = m$ NO SINGULARITY

$\nexists N(J^T)$ $\nexists N(J)$

Each joint movement produce an unique EE movement

1.2 $\text{Rank}(J/J^T) < m$ ROBOT IS AT SINGULARITY

$\exists N(J) \to \exists$ joint movements that don't produce EE movement

$\exists N(J^T) \to \exists$ EE forces that don't need joint forces

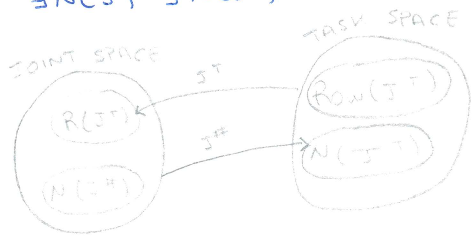
### REDUNDANT MANIPULATORS ($J^T$ SKINNY, $J$ FAT)

2.1 $\text{RANK}(J/J^T) = m$

$\exists N(J)$ $\nexists N(J^T)$

2.2. $\text{RANK}(J/J^T) < m$

$\exists N(J)$ $\exists N(J^T)$




## FORCES TRANSFORMATION

$$\begin{bmatrix} {}^B f \\ {}^B m \end{bmatrix} = {}^A A_B^T \begin{bmatrix} {}^A f \\ {}^A m \end{bmatrix} \qquad \begin{bmatrix} {}^B R_A & 0 \\ [{}^B P_{BA}]_\times {}^B R_A & {}^B R_A \end{bmatrix}$$

# DYNAMICS

Studies the cause of motion (forces are involved).

DYNAMIC MODEL provides a relationship between the sources of motion and resulting motion.

## Direct Dynamics

Compute movements given forces. $u(t) = \begin{bmatrix} \tau_1 \\ \vdots \\ \tau_n \end{bmatrix} \Rightarrow \text{ROBOT} \Rightarrow q(t) = \begin{bmatrix} q_1 \\ \vdots \\ q_n \end{bmatrix}$

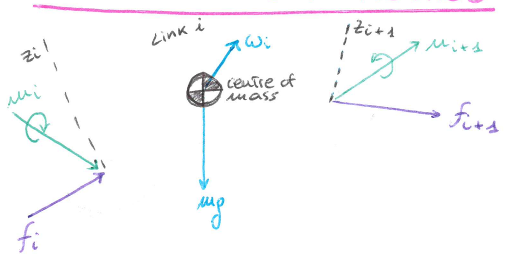
The solution is founded by integrating numerically the diff. equations of the model. Done by simulation

## Inverse Dynamics

Given a desired motion (trajectory) compute necessary forces to achive that movement.

- EXPERIMENTAL SOLUTION: iterative learning by trials
- ANALYTICAL SOLUTION: use dynamic model to compute algebrically $u^d(t)$ at every $t$.

## NEWTON - EULER METHOD




NEWTON EQUATION: $\underbrace{f_i - f_{i+1} + mg}_{\Sigma f_i} = m_i a_{ci}$

EULER EQUATION: $\underbrace{m_i - m_{i+1} + P_{c,i} \times f_i + P_{c,i+1} \times (-f_{i+1})}_{\Sigma m_i} = \underbrace{I_i \dot\omega_i + \omega_i \times (I_i \omega_i)}_{\text{Angular momentum}}$

### Recoursive Newton Euleur Algorithm (RNEA)

INPUT:

- base-link $\omega_0, \ddot{P}_0 - g_0, \dot\omega_0$
- Joint $q, \dot q, \ddot q$
- Last-link $f_{n+1}$ $m_{n+1}$

FWD PASS: Propagate velocities and acc. from base to EE with DK.

- $\omega_i = \omega_{i-1} + \dot q_i z_i$
- $\dot\omega_i = \dot\omega_{i-1} + \ddot q z_i + \dot q_i \omega_{i-1} \times z_i$
- $v_i = v_{i-1} + \omega_{i-1} \times P_{i-1,i}$
- $a_i = a_{i-1} + \dot\omega_{i-1} \times P_{i-1,i} + \omega_{i-1} \times (\omega_{i-1} \times P_{i-1,i})$

BWD PASS: compute forces / moments for each link going from EE to base using Newton-Euler equations.

$f_i = f_{i+1} + m_i(a_{ci} - g_i)$ FROM NEWTON

$m_i = m_{i+1} - P_{c,i} \times f_i + P_{c,i+1} \times f_{i+1} + I_i \dot\omega_i + \omega_i \times I_i \omega_i$ FROM EULER

OUTPUT: $\tau_i = \begin{cases} z_i^T f_i & \text{Prismatic joints} \\ z_i^T m_i & \text{revolute joints} \end{cases}$

- Inverse Dynamics $u = \text{RNEA}(0g, q, \dot q, \ddot q^d) = H(q)\ddot q^d + C(q, \dot q) + g(q)$
- Direct Dynamics compute $H$ and $h(q, \dot q)$ with RNEA $\ddot q = H(q)^{-1}(u - h) \to$ integrate

## Joint Space Dynamic Model

$H(q)\ddot q + C(q, \dot q) + g(q) = u$ The model found is in the form $\Phi(q, \dot q, \ddot q) = u$

SCALAR VERSION: $\sum_{j=1}^{n} m_{kj}(q)\ddot q_j + \sum_{i,j}^{n} c_{kij}(q)\dot q_i \dot q_j + \frac{\partial U}{\partial q_k} = u_k \quad k \in [1, n]$

# ROBOT CONTROL (JOINT SPACE)

## EQUILIBRIUM

$M(q)\ddot{q} + h(q,\dot{q}) = u$ an equilibrium point is where this equation is equal to zero.

state-space form: $\dot{x} = \underbrace{f_x(x_1, x_2)}_{\text{Drift}} + \underbrace{f_u(x_1)u}_{\text{Forcing term}}$ $\quad x_1 = q \quad x_2 = \dot{q}$ $\quad \dot{x} = \begin{bmatrix} x_2 \\ -M^{-1}(x_1)[h(x_1,x_2)] \end{bmatrix} + \begin{bmatrix} 0 \\ M^{-1}(x_1) \end{bmatrix} u$

UNFORCED EQUILIBRIUM

$f_u(\bar{x})u = 0$

$g(\bar{x}_1) = 0$

FORCED EQUILIBRIUM

$u = u(x)$

$u(\bar{x}) = g(\bar{x}_1)$ Joint Torques balance gravity

## PD Control

INPUTS: joint reference trajectories $q^d(t)$, $\dot{q}^d(t)$

joint position/velocities

CONTROL LAW: $u(t) = K_p(q^d(t) - q(t)) + K_d(\dot{q}^d(t) - \dot{q}(t))$

↳ If we replace this $u$ in dynamics equation we found: $g(q) = K_p(q^d - q)$ AT EQUILIBRIUM STEADY STATE ERROR

Joint distant has less error because it has to carry less weight.

## PD + GRAVITY COMPENSATION

IDEA: add a non-linear cancellation of gravity in the control law $u = K_p e(t) - K_d \dot{e}(t) + g(q)$

At equilibrium $q \to q^d$ $\dot{q} \to 0$ NO STEADY STATE ERROR

we need to identify $g(q)$

## PID CONTROL

we consider gravity as a constant disturbance acting on the system $u(t) = K_p(e(t)) + K_d(\dot{e}(t)) + \overbrace{K_i \int_0^t (e(\tau))\, d\tau}^{u_I}$

Integral component grows magnifying the error till the disturbance is not compensated → $e(t) > 0$ $u_I \uparrow$ ; $e(t) < 0$ $u_I \downarrow$

Integral action compensate gravity at equilibrium $g(q) = K_I \int_0^t (e(t))\, d\tau$

REGULATORS

TRAJECTORY PLANNER

## INVERSE DYNAMICS - DECENTRALIZED CONTROL

COUPLING EFFECTS

→ DECENTRALIZED CONTROL: irregularities caused by the movement of the other joint

→ CENTRALIZED CONTROL: compensate non-linearity to cancel the coupled dynamics and achieve a linear system with low gains → FEEDBACK LINEARIZATION

## Feedback Linearization

Methodology to control non-linear systems $\dot{x} = \alpha(x)u + \beta(x)$ defining a control action as: $u = \alpha(x)^{-1}(v - \beta(x))$

I obtain a linearized system in the new control input $v$.

The system becomes linear and DECOUPLED wrt any input

$$\dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 \\ M(x_1)^{-1} \end{bmatrix} u + \begin{bmatrix} x_2 \\ -h(x_1, x_2) \end{bmatrix}$$

$$\dot{x} = \alpha(x)u + \beta(x)$$

## INVERSE DYNAMIC COMPENSATION

reference trajectory is used to compute the torques that compensate coupling effects.

Given trajectory $q^d(t), \dot{q}^d(t), \ddot{q}^d(t)$ FEED-FORWARD action should give the exact reproduction of desired motion but disturbances and unmodelled dynamics cause divergences from desired trajectory.

The FEEDBACK term to make the control scheme more robust: $u = \underbrace{u_d}_{\text{open loop}} + \underbrace{K_p(e(t)) + K_p(\dot{e}(t))}_{\text{closed-loop}}$

## JOINT SPACE COMPUTE TORQUE CONTROL

compute linearizing controller using the measurements. Special case of feedback linearization.

① Evaluate $M, h$ at the current state $q, \dot{q}$

② Add an additional feedback

FINAL CONTROL LAW: $u = \underbrace{\hat{M}(q)v + \hat{h}(q,\dot{q})}_{\text{compensation terms computed online}}$ Non-linear state feedback

$M(q)\ddot{q} + \cancel{h} = Mv + \cancel{h} \Rightarrow \ddot{q} = v$

Our goal is $\ddot{e} + K_d\dot{e} + K_p e = 0 \Rightarrow$ suitably select $K_d, K_p$

Linear control law to stabilize the tracking error to zero! $v = \ddot{q}^d + K_p(q^d - q) + K_d(\dot{q}^d - \dot{q})$

# TASK SPACE CONTROL
Given a reference trajectory for the end-effector $p^d(t)$, compute $u(t)$ such that $p(t) \approx p^d(t)$

Ⓐ INVERSE KINEMATICS compute $q^d(t)$ such that $x^d(t) = f(q^d(t))$ then apply joint motion control.

Ⓑ DIRECT CARTESIAN SPACE CONTROL Instead of defining a motion for the joints, directly control forces that act on EE.

cartesian PD + gravity compensation: $u = J^T(q)[K_P(p^d - p) + K_d(\dot p^d - \dot p)] + g(q)$ no redundancy ($J^T$ square)

$$u = J^T f^d + [I - J^T J^{T\#}]\, \tau_0 \quad \text{postural task for redundant manipulators}$$

$$\underbrace{f^d = K_P(p^d - p) + K_d(\dot p^d - \dot p)}_{\text{cartesian task}} \qquad \underbrace{\tau_0 = K_q(q_0 - q) - K_{\dot q}(\dot q)}_{\text{postural task}}$$

Ⓒ COMPUTED TORQUE IN CARTESIAN SPACE: Joint space dynamics $M\ddot q + h = u$ ①

EE dynamics $\ddot q = J^{-1}(\ddot p - \dot J \dot q)$ ②

$\Rightarrow \Lambda(q)\ddot p + \mu(q, \dot q) = F$ Dynamics reflected at EE

FINAL CONTROL LAW: $u = J^T[\hat\Lambda(q)(\underbrace{\ddot p^d + K_P e_x + K_d \dot e_x}_{\text{terms of error correction from PD}}) + \hat\mu(q, \dot q)]$ CARTESIAN BIAS TERM

↳ Inertia depends on q and is variable in different cartesian directions

For redundant robots, redundancy will remain an internal dynamic that should be stabilized by a postural task in the null-space of $J^{T\#}$

↳ $u = J^T[\hat\Lambda v + \hat h] + [I - J^T J^\#]\, u_P$
↳ postural task to stabilize dynamics

Ⓓ Inverse Kinematics + Joint computed torque

Solve IK at the acceleration level to get $\ddot q^d$ then use joint space inverse dynamics (JSID) to get $u$.

REDUNDANT ROBOTS: $u = M\ddot q^d + h = M[\underbrace{J^\#(v - \dot J \dot q) + (I - J^\# J)\ddot q_0}_{\ddot q^d \text{ from } \ddot p = J\ddot q + \dot J \dot q}] + h$ (more general)

# ORIENTATION CONTROL OF EE
$u = [u_x, u_y, u_z] \in \mathbb{R}^3 \rightarrow J_o^T \rightarrow \tau \in \mathbb{R}^n$

$u = J_o^T(K_o e_o + D_o(\underbrace{\omega^d - J_o \dot q}_{\dot e}))$ momentum proportional to error: $u = K_o e_o$

$e_o$: orientation error

↳ PD for orientation control

## ORIENTATION ERROR
(1) Rotation Matrix: ${}^eR_d = {}^wR_e^T\, {}^wR_d$ difficult handle 9 elements of rotation matrix

${}^w e_o = {}^wR_e\, {}^e e_o$ ↳ ${}^e e_o = \frac{1}{2}\begin{bmatrix} R_{32} - R_{23} \\ R_{13} - R_{31} \\ R_{21} - R_{12} \end{bmatrix}$ computed from rotation matrix

(2) Angle-Axis: $e_o = \hat r \sin(\Delta\theta)$.

(3) Euler Angles: $e_o = \Phi_d - \Phi_e(q)$ we use analytical Jacobian to map to torques. There are representations of singularities in $J_A$.

(4) Quaternions: (1) compute quaternions $Q_e$ and $Q_d$ associated with ${}^wR_e$ and ${}^wR_d$

(2) $\Delta Q = \bar Q_e \otimes Q_d = \begin{bmatrix} \eta_e \eta_d + \epsilon_e^T \epsilon_d \\ \eta_e \epsilon_d - \eta_d \epsilon_e - \epsilon_e \times \epsilon_d \end{bmatrix}$

↳ $e_o$

# INTERACTION
PASSIVE
Physical springs are introduced to reduce interaction force

ACTIVE
feedback control technique

DIRECT
explicit force feedback
measure contact force $f$
if $f < f^d \Rightarrow$ apply more force
if $f > f^d \Rightarrow$ apply less force

INDIRECT
Force and position at same time

Impedance
Indirectly regulate contact forces by generating a motion that satisfies a dynamic relation between force and position
$M_x \ddot p + D_x \dot p + K_x p = F$

Real Dynamics (in contact):
$M\ddot q + h = u + J^T F_{ext}$

Admittance
For robots that are only position controlled. Maps contact forces into displacements wrt the reference position. Need force sensor to measure contact force.
$\Delta q = J^{-1}_{(q)} K_x^{-1} F_{ext}$ CARTESIAN SPACE
$\Delta q = K_q^{-1} J^T F_{ext}$ JOINT SPACE

Torsional impedance: $M_\theta \Delta\ddot\phi + D_\theta \Delta\dot\phi + K_\theta \Delta\phi = T(\phi)^T u_{ext}$
represents also singularities

## SELECTION OF IMPEDANCE PARAMETERS
- $K_x$ (STIFFNESS) → ↑↑ compliant environment
  → ↓↓ stiff environment
- $M_x$ (INERTIA) → ↑↑ where we expect contact (↓↓ acc.)
  → ↓↓ where we expect free motion (↑↑ acc.)
- $D_x$ (DAMPING) → Regulate transient