
*Robotics 2*

# Robots with kinematic redundancy

## Part 1: Fundamentals

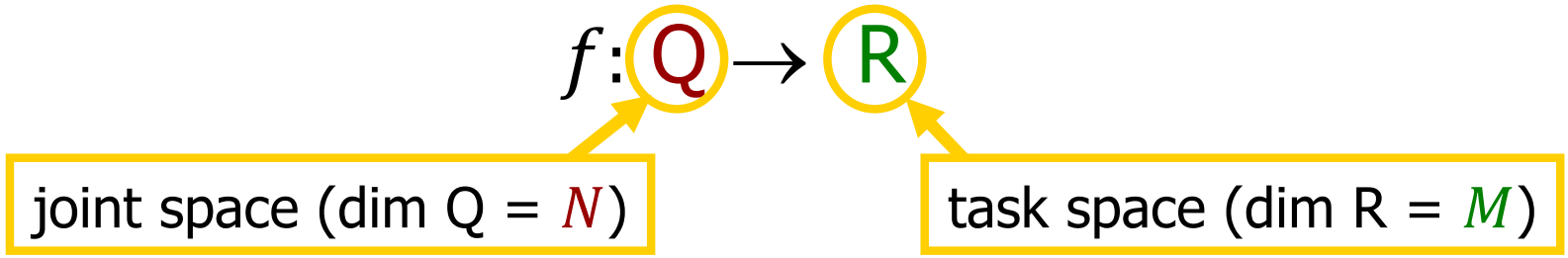
Prof. Alessandro De Luca

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI


SAPIENZA
UNIVERSITÀ DI ROMA

# Redundant robots

- direct kinematics of the task $r = f(q)$




- a robot is (kinematically) redundant for the task if $N > M$
  (more degrees of freedom than strictly needed for executing the task)
- $r$ may contain the position and/or the orientation of the end-effector or, more in general, be any parameterization of the task (even not in the Cartesian workspace)
- "redundancy" of a robot is thus a relative concept, i.e., it holds with respect to a given task

# Some E-E tasks and their dimensions


| TASKS [for the robot end-effector (E-E)] | dimension $M$ |
|---|---|
| position in the plane | 2 |
| position in 3D space | 3 |
| orientation in the plane | 1 |
| pointing in 3D space | 2 |
| position and orientation in 3D space | 6 |

a planar robot with $N = 3$ joints is redundant for the task of positioning its E-E in the plane ($M = 2$), but NOT for the task of positioning AND orienting the E-E in the plane ($M = 3$)

# Typical cases of redundant robots

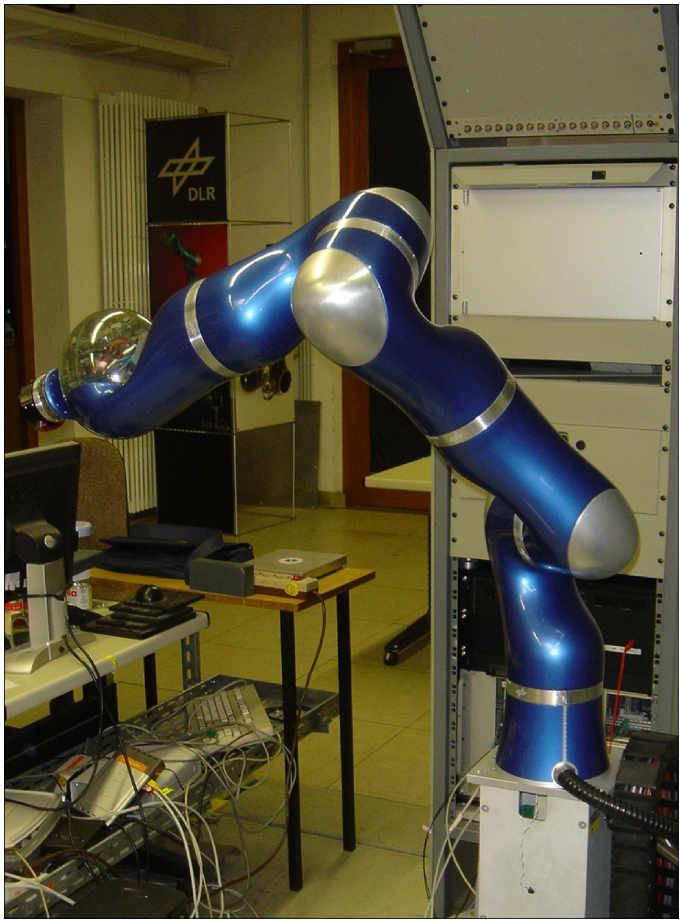

- 6R robot mounted on a linear track/rail
  - 7 dofs for positioning and orienting its end-effector in 3D space
- 6-dof robot used for arc welding tasks
  - task does not prescribe the final roll angle of the welding gun
- dexterous robotic hands
- multiple cooperating manipulators
- manipulator on a mobile base
- humanoid robots, team of mobile robots ...
- "kinematic" redundancy is not the only type…
  - redundancy of components (actuators, sensors)
  - redundancy in the control/supervision architecture

# Uses of robot redundancy

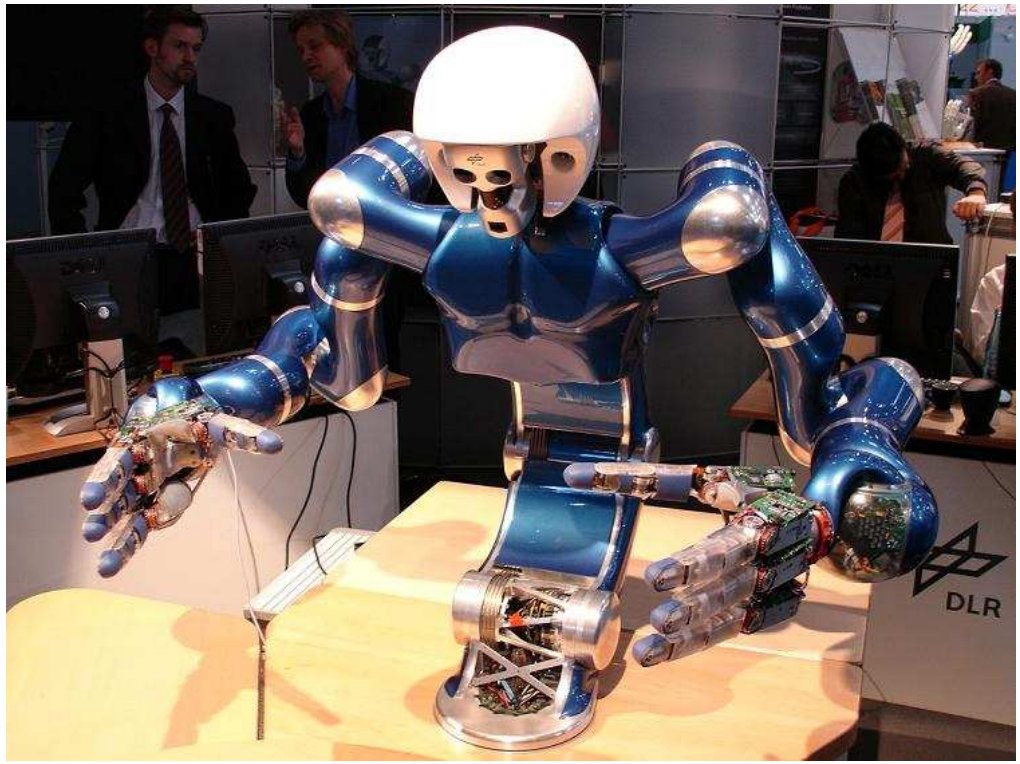
- avoid collision with obstacles (in Cartesian space) ...
- ... or kinematic singularities (in joint space)
- stay within the admissible joint ranges
- increase manipulability in specified directions
- uniformly distribute/limit joint velocities and/or accelerations
- minimize energy consumption or needed motion torques
- optimize execution time
- increase dependability with respect to faults
- ...

all objectives should be quantitatively "measurable"

# DLR robots: LWR-III and Justin




7R LWR-III lightweight manipulator:
elastic joints (HD), joint torque sensing,
13.5 kg weight = payload




Justin two-arm upper-body humanoid:
43R actuated =
two arms (2×7) + torso (3*)
+ head (2) + two hands (2×12),
45 kg weight

* = one joint is dependent on the motion of the other two

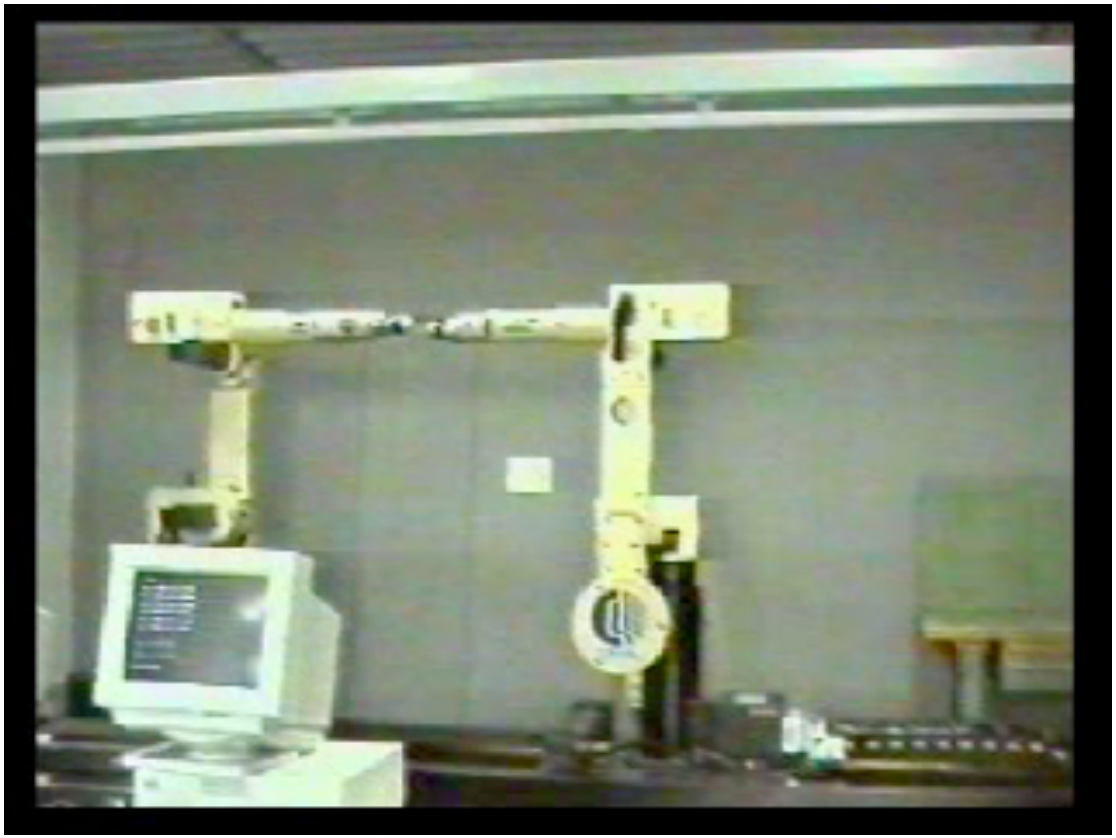
# Justin carrying a trailer

video



motion planning for DLR Justin robot in the configuration space, avoiding Cartesian obstacles and using robot redundancy

# Dual-arm redundancy



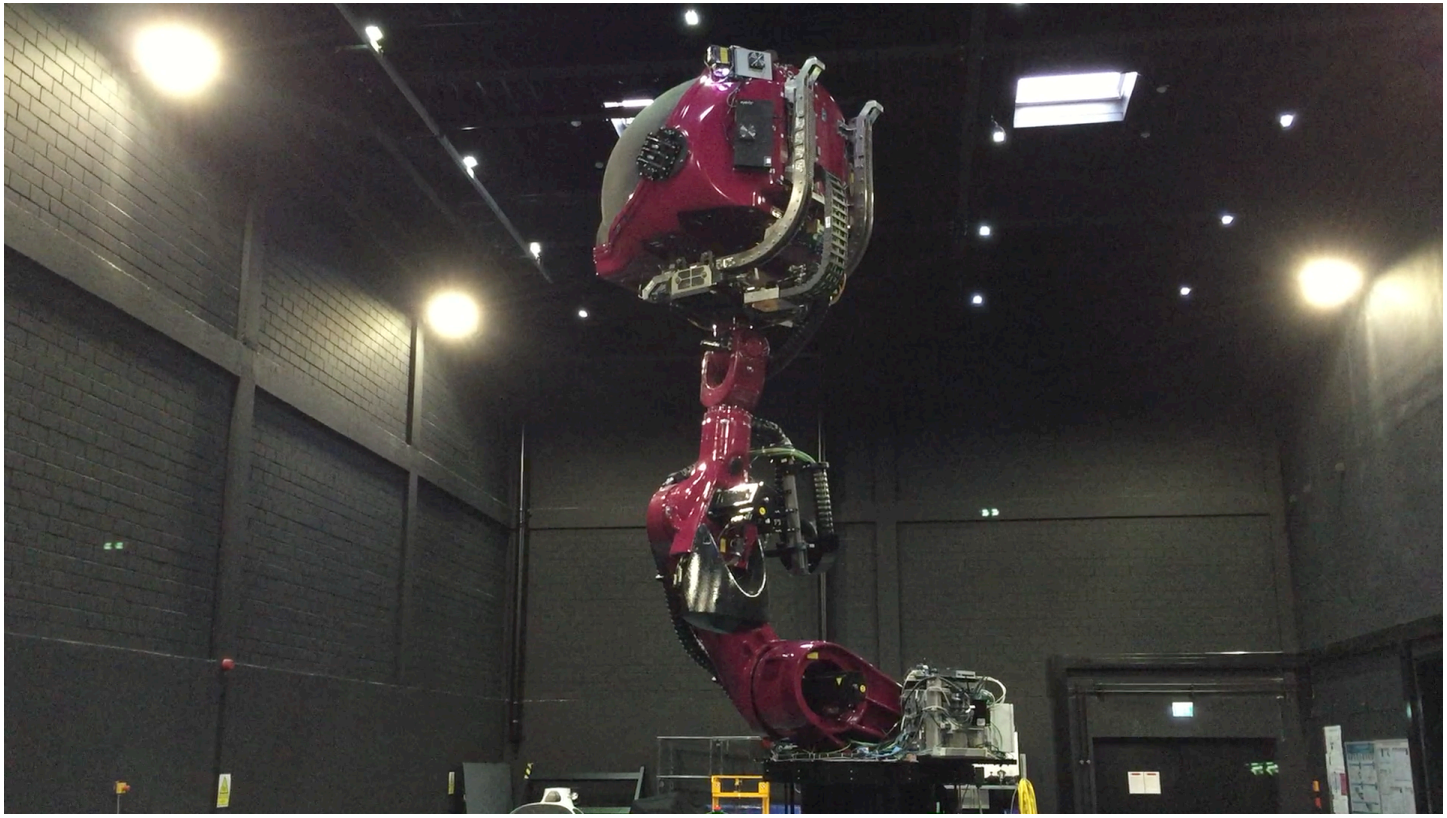
video

DIS, Uni Napoli

two 6R Comau robots, one mounted on a linear track (+1P)
coordinated 6D motion using the null-space of the right-side robot ($N - M = 1$)

# Motion cueing from redundancy

video



Max Planck Institute for Biological Cybernetics, Tübingen

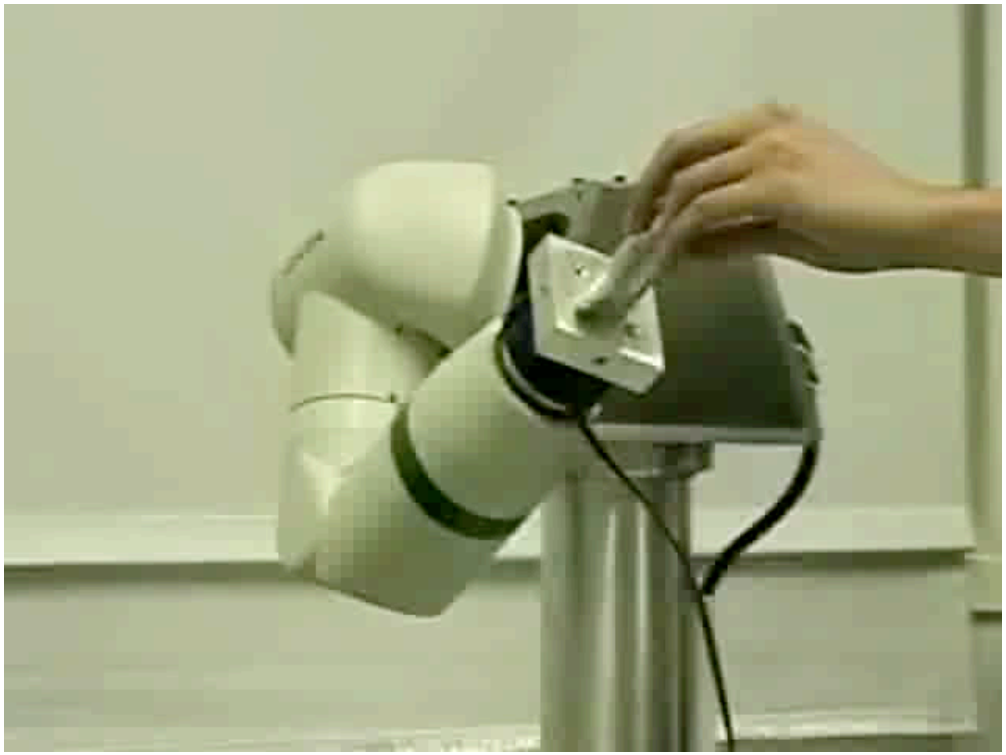
a 6R KUKA KR500 mounted on a linear track (+1P) with a sliding cabin (+1R), used as a dynamic emulation platform for human perception ($N - M = 2$)

# Self-motion

video

8R Dexter: self-motion with constant 6D pose of E-E ($N - M = 2$)

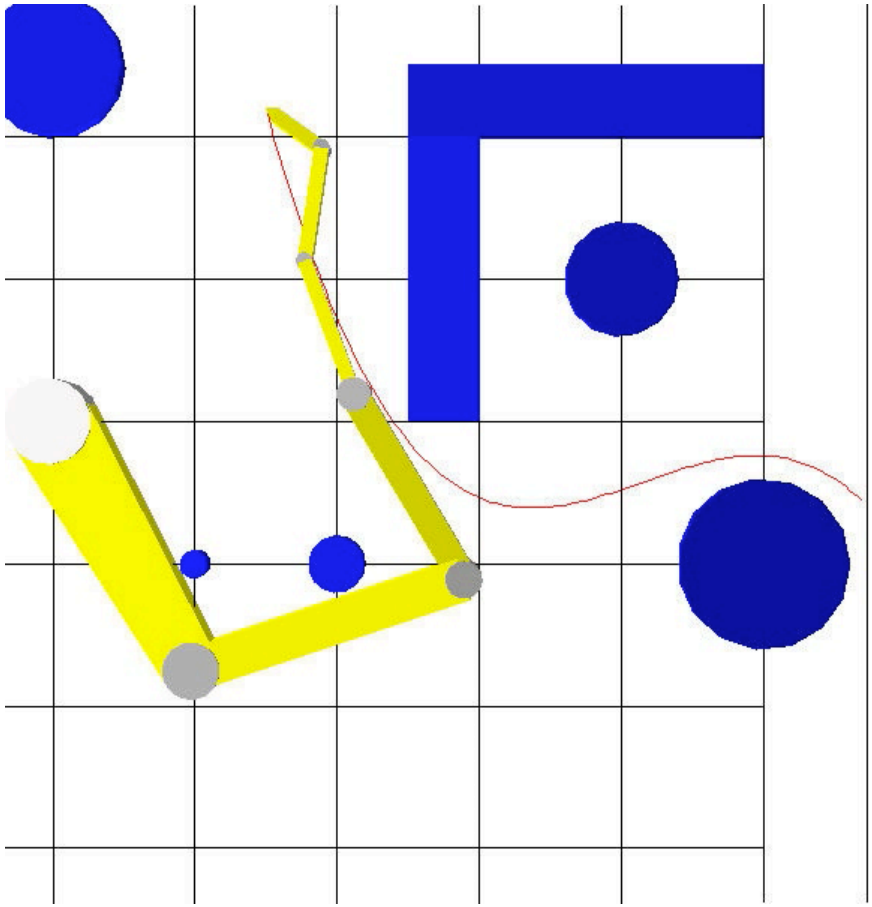
video



Nakamura's Lab, Uni Tokyo

6R robot with spherical shoulder in compliant tasks for the Cartesian E-E position ($N - M = 3$)

# Obstacle avoidance



video

6R planar arm moving on a given geometric path for the E-E ($N - M = 4$)

# Disadvantages of redundancy

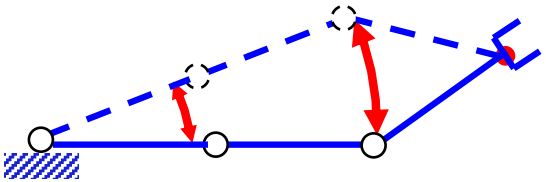

- potential benefits should be traded off against
  - a greater structural complexity of construction
    - mechanical (more links, transmissions, ...)
    - more actuators, sensors, ...
    - costs
  - more complicated algorithms for inverse kinematics and motion control

# Inverse kinematics problem

- find $q(t)$ that realizes the task: $f\big(q(t)\big) = r(t)$ (at all times $t$)
- infinite solutions exist when the robot is redundant

  (even for $r(t) = r$ = constant)

$N = 3 > 2 = M$



$r$ = constant
E-E position

- the robot arm may have "internal displacements" that are unobservable at the task level (e.g., not contributing to E-E motion)
  - these joint displacements can be chosen so as to improve/optimize in some way the behavior of the robotic system
- self-motion: an arm reconfiguration in the joint space that does not change/affect the value of the task variables $r$
- solutions are mainly sought at differential level (e.g., velocity)

# Redundancy resolution

via optimization of an objective function


## Local methods

given $\dot{r}(t)$ and $q(t)$, $t = kT_s$

optimization of $H(q, \dot{q})$

$\dot{q}(kT_s)$ ← ON-LINE

$$q\big((k+1)T_s\big) = q(kT_s) + T_s\ \dot{q}(kT_s)$$

discrete-time form

## Global methods

given $r(t)$, $t \in [t_0,\ t_0 + T]$, $q(t_0)$

optimization of $\int_{t_0}^{t_0+T} H(q, \dot{q})dt$

$$q(t),\ t \in [t_0,\ t_0 + T]$$

OFF-LINE

relatively EASY
(a LQ problem)

⟺

quite DIFFICULT
(nonlinear TPBV problems arise)

# Local resolution methods


## three classes of methods for solving $\dot{r} = J(q)\dot{q}$

1. **Jacobian-based methods** (here, analytic Jacobian in general!)

   among the infinite solutions, one is chosen, e.g., that minimizes a suitable (possibly weighted) norm

2. **null-space methods**

   a term is added to the previous solution so as not to affect execution of the task trajectory, i.e., belonging to the null-space $\mathcal{N}(J(q))$

3. **task augmentation methods**

   redundancy is reduced/eliminated by adding $S \leq N - M$ further auxiliary tasks (when $S = N - M$, the problem has been "squared")

$$r = f(q) \Rightarrow \dot{r} = J(q)\dot{q}$$

# 1 Jacobian-based methods

we look for a solution to $\dot{r} = J(q)\dot{q}$ in the form

$J = \underbrace{\square}_{N}\}M$ $\qquad \dot{q} = K(q)\dot{r} \qquad$ $K = \underbrace{\square}_{M}\}N$

minimum requirement for $K$: $J(q)K(q)J(q) = J(q)$

( ➡ $K$ = generalized inverse of $J$ )

$$\forall \dot{r} \in \mathcal{R}(J(q)) \Rightarrow \quad J(q)[K(q)\dot{r}] = J(q)K(q)J(q)\dot{q} = J(q)\dot{q} = \dot{r}$$

example:
if $J = [J_a \; J_b]$, $\det(J_a) \neq 0$, one such generalized inverse of $J$ is $K_r = \begin{pmatrix} J_a^{-1} \\ 0 \end{pmatrix}$
(actually, this is a stronger right-inverse)

# Pseudoinverse


$$\dot{q} = J^{\#}(q)\dot{r}$$ ... a very common choice: $K = J^{\#}$

- $J^{\#}$ always exists, and is the unique matrix satisfying

$$J\,J^{\#}\,J = J \qquad J^{\#}\,J\,J^{\#} = J^{\#}$$
$$\left(J\,J^{\#}\right)^{T} = J\,J^{\#} \qquad \left(J^{\#}\,J\right)^{T} = J^{\#}\,J$$

- if $J$ is full (row) rank, $J^{\#} = J^{T}(J\,J^{T})^{-1}$; else, it is computed numerically using the SVD (Singular Value Decomposition) of $J$ (`pinv` of Matlab)
- the pseudo-inverse joint velocity is the only that minimizes the norm $\|\dot{q}\|^2 = \dot{q}^T\dot{q}$ among all joint velocities that minimize the task error norm $\|\dot{r} - J(q)\dot{q}\|^2$
- if the task is feasible ($\dot{r} \in \mathcal{R}(J(q))$), there will be no task error

# Weighted pseudoinverse

$$\dot{q} = J_W^{\#}(q)\dot{r}$$ another choice: $K = J_W^{\#}$

- if $J$ is full (row) rank, $J_W^{\#} = W^{-1}J^T(JW^{-1}J^T)^{-1}$
- the solution $\dot{q}$ minimizes the weighted norm

$$\|\dot{q}\|_W^2 = \dot{q}^T W\, \dot{q}$$ $W > 0$, symmetric (often diagonal)

- large weight $W_i \Rightarrow$ small $\dot{q}_i$ (e.g., weights can be chosen proportionally to the inverse of the joint ranges)
- it is NOT a "pseudoinverse" (4th relation does not hold ...) but shares similar properties

# Singular Value Decomposition (SVD)


- the SVD routine of Matlab applied to $J$ provides two orthonormal matrices $U_{M\times M}$ and $V_{N\times N}$, and a matrix $\Sigma_{M\times N}$ of the form

$$\Sigma = \left(\begin{array}{ccc|c} \sigma_1 & & & \\ & \sigma_2 & & 0_{M\times(N-M)} \\ & & \ddots & \\ & & \sigma_M & \end{array}\right) \qquad \begin{array}{c} \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_\rho > 0 \\ \sigma_{\rho+1} = \cdots = \sigma_M = 0 \end{array}$$

singular values of $J$

where $\rho = \text{rank}(J) \leq M$, so that their product is

$$\boxed{J = U\Sigma V^T}$$

- the columns of $U$ are eigenvectors of $J\,J^T$ (associated to its non-negative eigenvalues $\sigma_i{}^2$), the columns of $V$ are eigenvectors of $J^TJ$
- the last $N-\rho$ columns of $V$ are a basis for the null space of $J$

$$Jv_i = \sigma_i u_i \quad (\mathrm{i} = 1, \cdots, \rho) \qquad \boxed{Jv_i = 0 \quad (\mathrm{i} = \rho + 1, \cdots, N)}$$

# Computation of pseudoinverses

- show that the pseudoinverse of $J$ is equal to

$$J = U\Sigma V^T \quad \Rightarrow \quad J^{\#} = V\Sigma^{\#}U^T \qquad \Sigma^{\#} = \begin{pmatrix} \frac{1}{\sigma_1} & & & \\ & \ddots & & \\ & & \frac{1}{\sigma_\rho} & \\ & & & 0_{(M-\rho)\times(M-\rho)} \\ \hline & & 0_{(N-M)\times M} & \end{pmatrix}$$

  for any rank $\rho$ of $J$

- show that matrix $J_W^{\#}$ appears when solving the constrained linear-quadratic (LQ) optimization problem (with $W > 0$, symmetric, and assuming $J$ of full rank)

$$\min \frac{1}{2}\|\dot{q}\|_W^2 \qquad \text{s.t.} \qquad J(q)\dot{q} - \dot{r} = 0$$

  and that the pseudoinverse is a particular case for $W = I$

- show that a weighted pseudoinverse of J can be computed by SVD/pinv as

$$J_{aux} = JW^{-1/2} \qquad J_W^{\#} = W^{-1/2}\,\text{pinv}(J_{aux})$$

applies equally to square and non-square matrices

# Singularity robustness
## Damped Least Squares (DLS)


unconstrained minimization of a **suitable** objective function

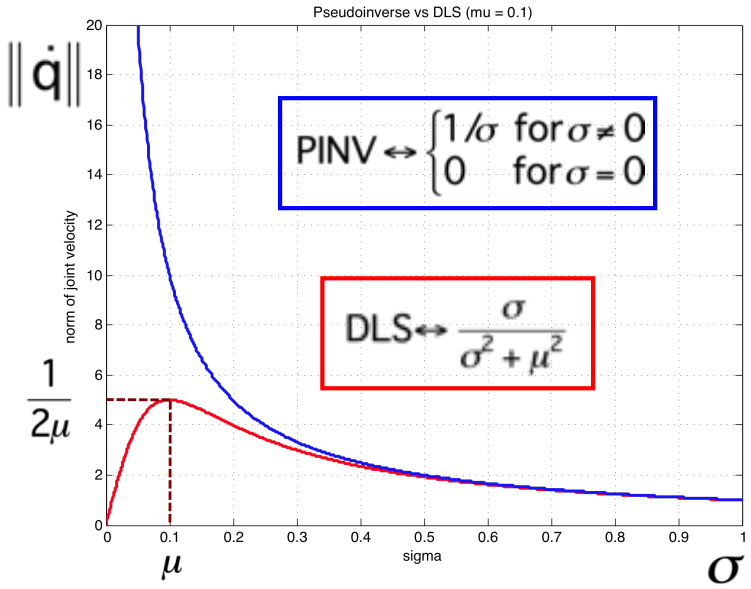
$$\min_{\dot{q}} H(\dot{q}) = \frac{\mu^2}{2}\|\dot{q}\|^2 + \frac{1}{2}\|\dot{r} - J\dot{q}\|^2$$

**compromise** between large joint velocity and task accuracy

SOLUTION $\dot{q} = J_{DLS}(q)\dot{r} = J^T(J\,J^T + \mu^2 I_M)^{-1}\dot{r}$

- induces a robust behavior when crossing singularities, but in its basic version gives always a task error $\dot{e} = \mu^2(J\,J^T + \mu^2 I_M)^{-1}\dot{r}$ (as in the N=M case)
  - $J_{DLS}$ is not a generalized inverse $K$
  - using SVD: $J = U\,\Sigma\,V^T \Rightarrow J_{DLS} = V\Sigma_{DLS}U^T$, $\Sigma_{DLS} = \begin{pmatrix} \underbrace{diag\left\{\frac{\sigma_i}{\sigma_i^2+\mu_i^2}\right\}}_{\rho\times\rho} & 0_{(M-\rho)\times(M-\rho)} \\ 0_{(N-M)\times\rho} & 0_{(N-M)\times(M-\rho)} \end{pmatrix}$
  - choice of a variable damping factor $\mu^2(q) \geq 0$, as a function of the minimum singular $\sigma_m(q)$ value of $J$ ≅ measure of distance to singularity
  - numerical filtering: introduces damping only/mostly in non-feasible directions for the task (see Maciejewski and Klein, *J of Rob Syst*, 1988)

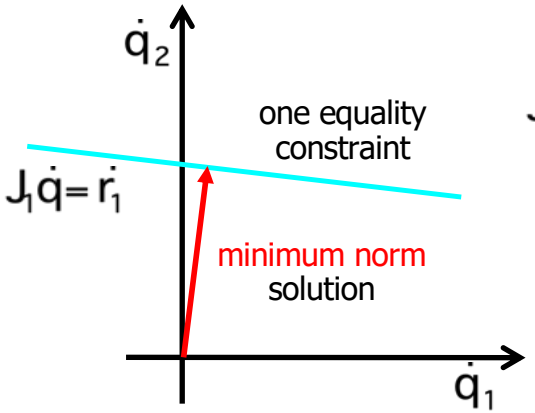
# Behavior of DLS solution

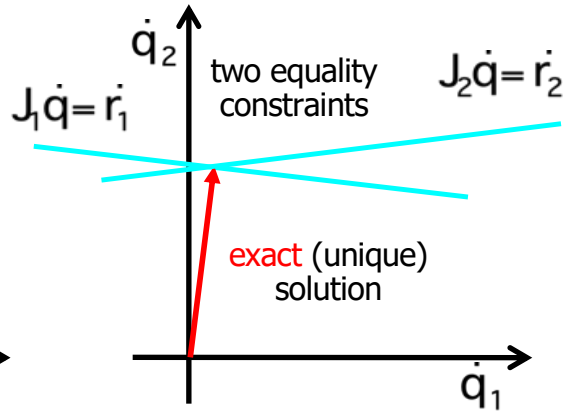


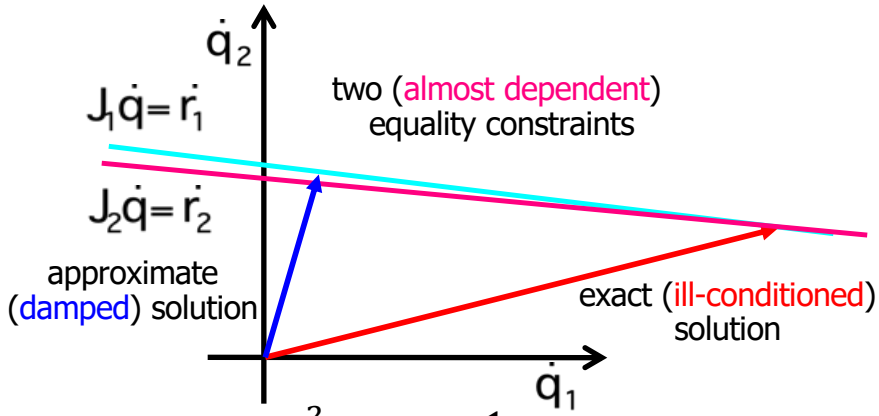

a. comparison of joint velocity norm with PINV (pseudoinverse) or DLS solutions

- in a direction of a singular vector $u$, when the associated singular value $\sigma \longrightarrow 0$
- PINV goes to infinity (and then is 0 at $\sigma = 0$)
- DLS peaks a value of $1/2\mu$ at $\sigma = \mu$ (and then smoothly goes to 0...)

b. graphical interpretation of "damping" effect (here $M = N = 2$, for simplicity)



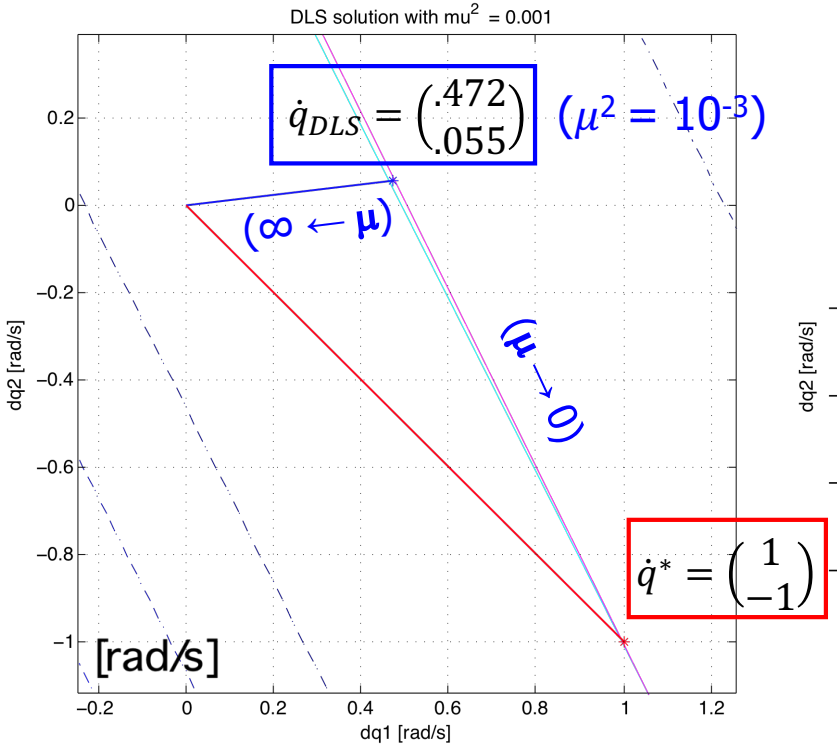



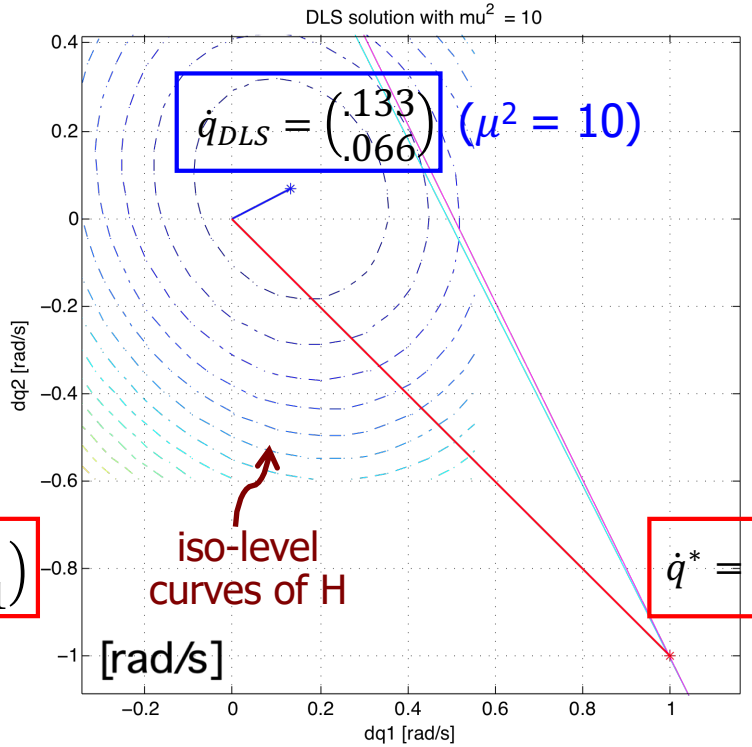




$$H(\dot{q}) = \frac{\mu^2}{2}\|\dot{q}\|^2 + \frac{1}{2}\|\dot{r} - J\dot{q}\|^2$$

# Numerical example of DLS solution

planar 2R arm, unit links, close to (stretched) singular configuration $q_1 = 45°$, $q_2 = 1.5°$)




$\dot{r} = \begin{pmatrix} -1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}$

$\in \mathcal{R}(J)$ even @singularity!

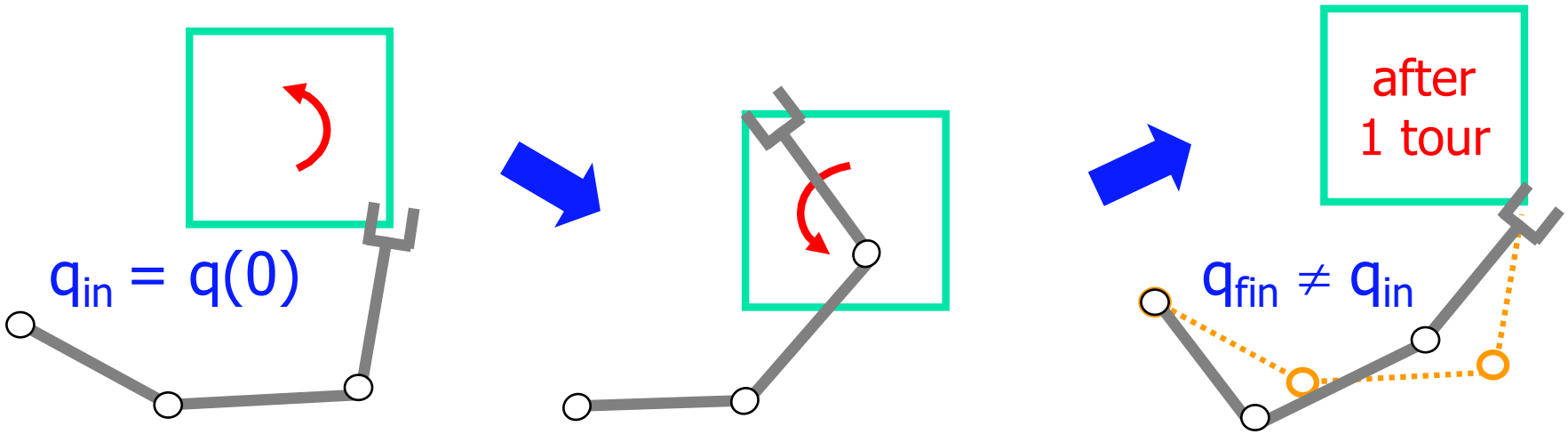
exact solution ($\mu$=0)

$$H = \frac{\mu^2}{2}\|\dot{q}\|^2 + \frac{1}{2}\|\dot{r} - J\dot{q}\|^2$$

| $\mu^2$ | 0 | $10^{-4}$ | $10^{-3}$ | $10^{-2}$ | 10 |
|---|---|---|---|---|---|
| $\|\|\dot{q}\|\|$ | $\sqrt{2}$ | .8954 | .4755 | .4467 | .1490 |
| $\|\|\dot{e}\|\|$ | 0 | $6.6 \cdot 10^{-3}$ | $1.4 \cdot 10^{-2}$ | $1.6 \cdot 10^{-2}$ | .6668 |
| $H_{\min}$ | 0 | $7.7 \cdot 10^{-5}$ | $2.2 \cdot 10^{-4}$ | $1.2 \cdot 10^{-3}$ | $3.4 \cdot 10^{-1}$ |

# Limits of Jacobian-based methods

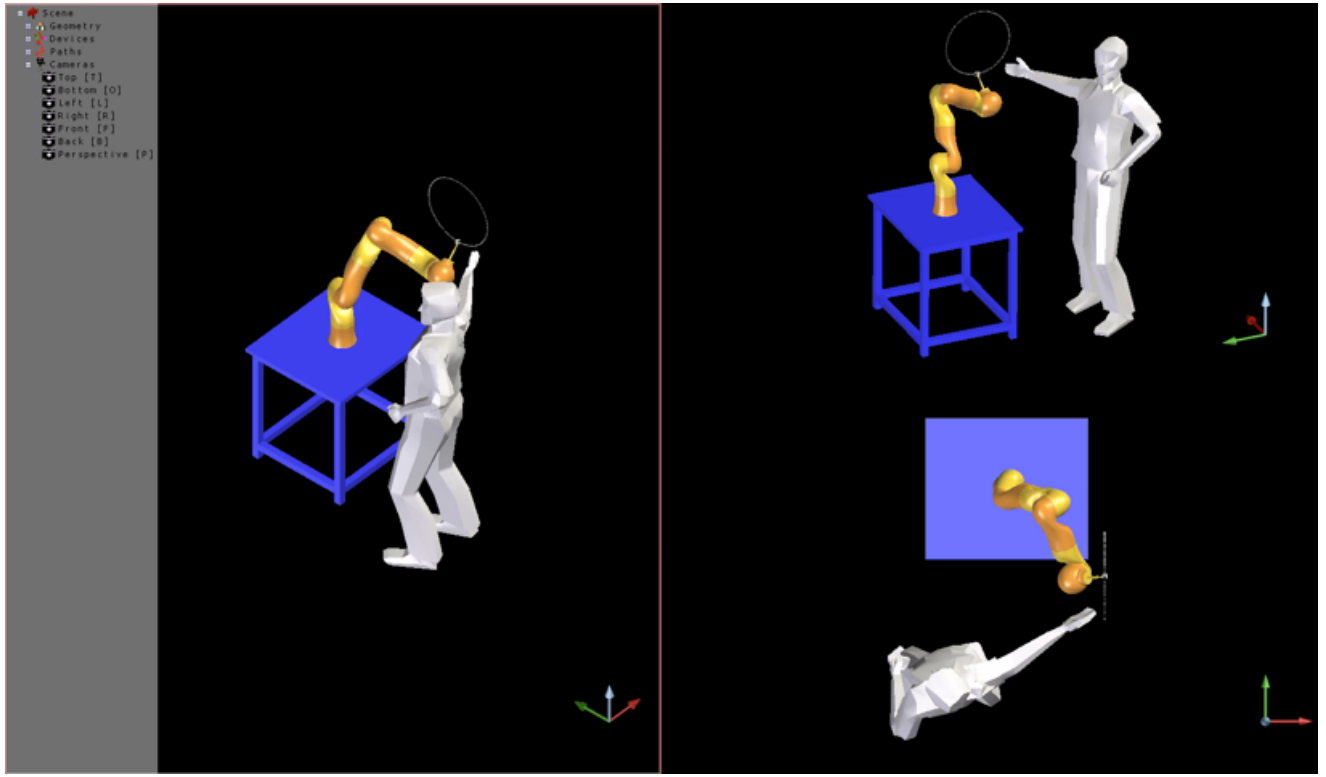
- no guarantee that singularities are globally avoided during task execution
  - despite joint velocities are kept to a minimum, this is only a local property and "avalanche" phenomena may occur
- typically lead to non-repeatable motion in the joint space
  - cyclic motions in task space do not map to cyclic motions in joint space




$$q(t) = q(0) + \int_0^t K\big(q(\tau)\big)\dot{r}(\tau)\, d\tau$$

# Drift with Jacobian pseudoinverse

- a 7R KUKA LWR4 robot moves in the vicinity of a human operator
- we command a cyclic Cartesian path (only in position, $M = 3$), to be repeated several times using the pseudoinverse solution
- (unexpected) collision of a link occurs during the third cycle ...




video

# 2 Null-space methods

general solution of $J\dot{q} = \dot{r}$

$$\dot{q} = J^{\#}\dot{r} + (I - J^{\#}J)\dot{q}_0$$

all solutions of the associated homogeneous equation $J\dot{q} = 0$ (self-motions)

a particular solution (here, the pseudoinverse) in $\mathcal{R}(J^T)$

"orthogonal" projection of $\dot{q}_0$ in $\mathcal{N}(J)$

properties of projector $[I - J^{\#}J]$

- symmetric
- idempotent: $[I - J^{\#}J]^2 = [I - J^{\#}J]$
- $[I - J^{\#}J]^{\#} = [I - J^{\#}J]$
- $J^{\#}\dot{r}$ is orthogonal to $[I - J^{\#}J]\dot{q}_0$

even more in general…

$$\dot{q} = K_1\dot{r} + (I - K_2J)\dot{q}_0$$

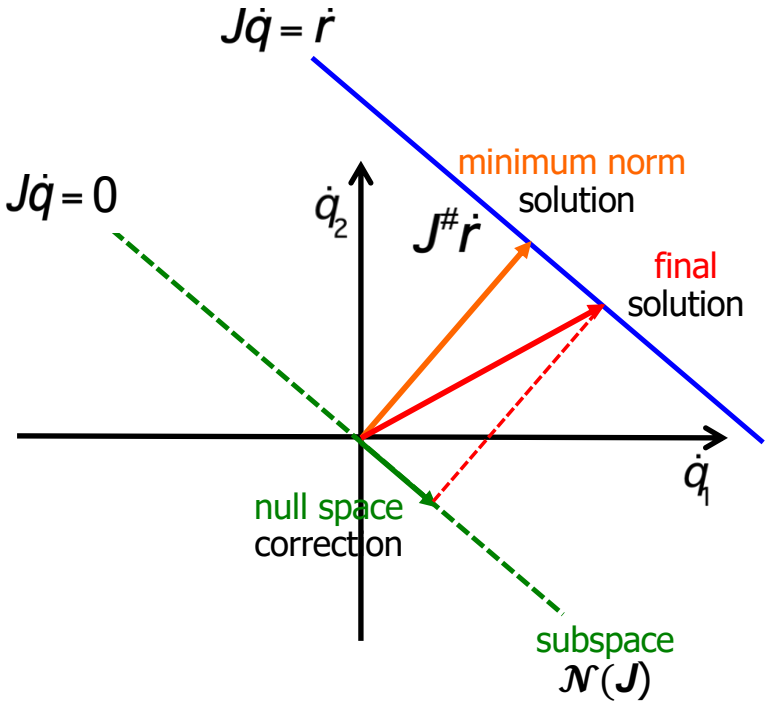
… but with less nice properties!

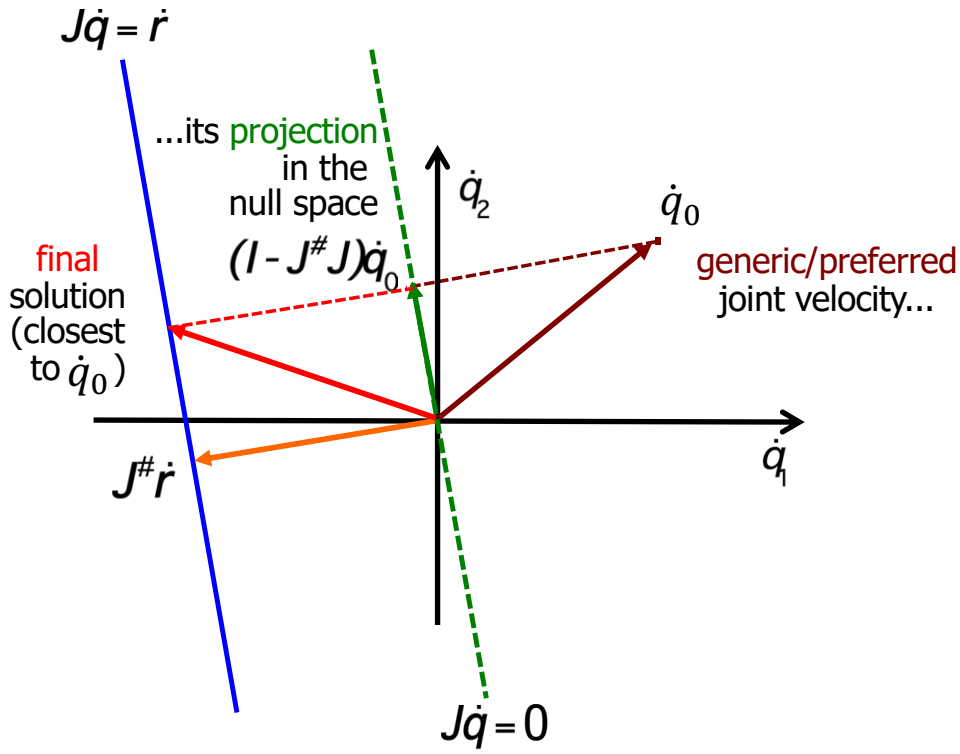
$K_1, K_2$ generalized inverses of $J$ ($JK_iJ = J$)

how do we choose $\dot{q}_0$?

# Geometric view on Jacobian null space

in the space of velocity commands




a correction is added to the original pseudoinverse (minimum norm) solution
i) which is in the null space of the Jacobian
ii) and possibly satisfies additional criteria or objectives

# Linear-Quadratic Optimization

## generalities

$$\min_x H(x) = \frac{1}{2}(x - x_0)^T W (x - x_0)$$

$$\text{s.t. } Jx = y \quad (M \times N)$$

$$x \in \mathbb{R}^N$$

$$W > 0 \text{ (symmetric)}$$

$$y \in \mathbb{R}^M$$

$$\text{rank}(J) = \rho(J) = M$$

$$L(x, \lambda) = H(x) + \lambda^T (Jx - y)$$ ← Lagrangian (with multipliers $\lambda$)

necessary conditions

$$\nabla_x L = \left(\frac{\partial L}{\partial x}\right)^T = W(x - x_0) + J^T \lambda = 0$$

$$\nabla_\lambda L = \left(\frac{\partial L}{\partial \lambda}\right)^T = Jx - y = 0$$

$$x = x_0 - W^{-1} J^T \lambda$$

$$Jx_0 - JW^{-1}J^T \lambda - y = 0$$

\+ sufficient condition for a minimum

$$\nabla_x^2 L = W > 0$$

$$\lambda = (JW^{-1}J^T)^{-1}(Jx_0 - y)$$

$M \times M$ invertible

$$x = x_0 + W^{-1}J^T(JW^{-1}J^T)^{-1}(y - Jx_0)$$

# Linear-Quadratic Optimization

## application to robot redundancy resolution


PROBLEM

$$\min_{\dot{q}} H(\dot{q}) = \frac{1}{2}(\dot{q}-\dot{q}_0)^T W (\dot{q}-\dot{q}_0)$$
$$\text{s.t.} \quad J\dot{q} = \dot{r}$$

$\dot{q}_0$ is a "privileged" joint velocity

SOLUTION

$$\dot{q} = \dot{q}_0 + \underbrace{W^{-1}J^T(JW^{-1}J^T)^{-1}}_{J_W^{\#}}(\dot{r} - J\dot{q}_0)$$

$$\dot{q} = J_W^{\#}\dot{r} + \left(I - J_W^{\#}J\right)\dot{q}_0$$

minimum weighted norm solution (for $\dot{q}_0 = 0$)

"projection" matrix in the null-space $\mathcal{N}(J)$

# Projected Gradient (PG)

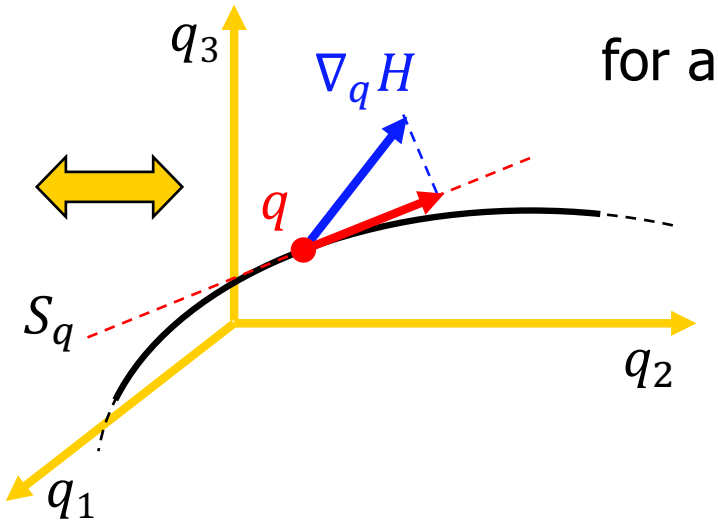
$$\dot{q} = J^{\#}\dot{r} + \left(I - J^{\#}J\right)\dot{q}_0$$

the choice $\dot{q}_0 = \nabla_q H(q)$ → differentiable objective function

realizes one step of a constrained optimization algorithm

while executing the time-varying task $r(t)$
the robot tries to increase the value of $H(q)$

projected gradient




for a fixed $\bar{r}$: $S_q = \{q \in \mathbb{R}^N : f(q) = \bar{r}\}$

$$\Rightarrow \dot{q} = \left(I - J^{\#}J\right)\nabla_q H$$

$N$-dimensional $\left(I - J^{\#}J\right)\nabla_q H = 0$

is a necessary condition
of constrained optimality

# Typical objective functions H(q)


- **manipulability** (maximize the "distance" from singularities)

$$H_{\text{man}}(q) = \sqrt{\det[J(q)J^T(q)]}$$

- **joint range** (minimize the "distance" from the mid points of the joint ranges)

$$q_i \in [q_{m,i}, q_{M,i}]$$

$$\bar{q}_i = \frac{q_{M,i} + q_{m,i}}{2}$$

$$H_{\text{range}}(q) = \frac{1}{2N}\sum_{i=1}^{N}\left(\frac{q_i - \bar{q}_i}{q_{M,i} - q_{m,i}}\right)^2$$

$$\dot{q}_0 = -\nabla_q H(q)$$

- **obstacle avoidance** (maximize the minimum distance to Cartesian obstacles)

also known as "clearance"

$$H_{\text{obs}}(q) = \min_{\substack{a \in \text{robot} \\ b \in \text{obstacles}}} \|a(q) - b\|^2$$

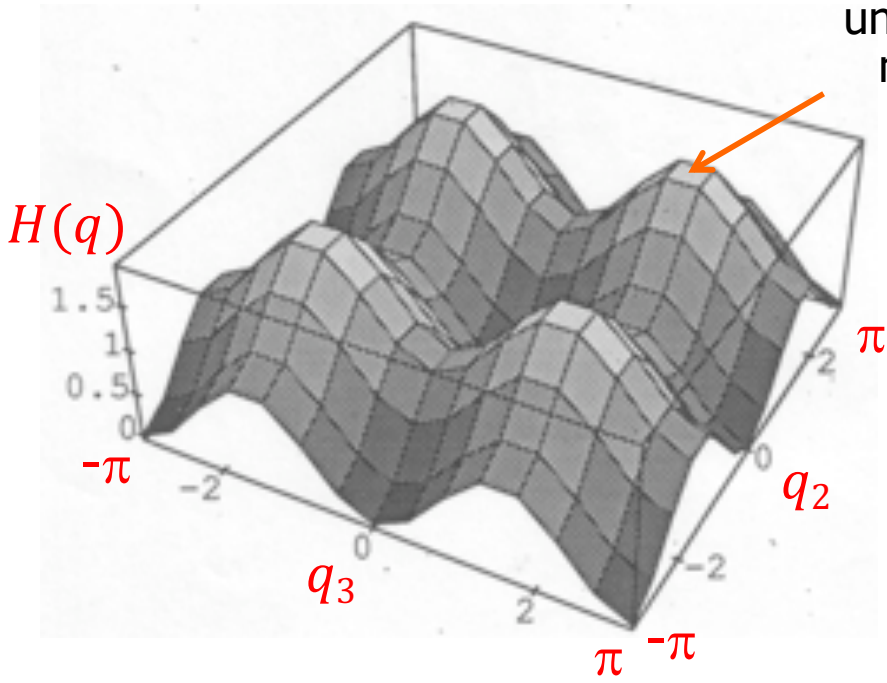
potential difficulties due to non-differentiability (this is a max-min problem)

# Singularities of planar 3R arm

the robot is redundant for a positioning task in the plane ($M = 2$)

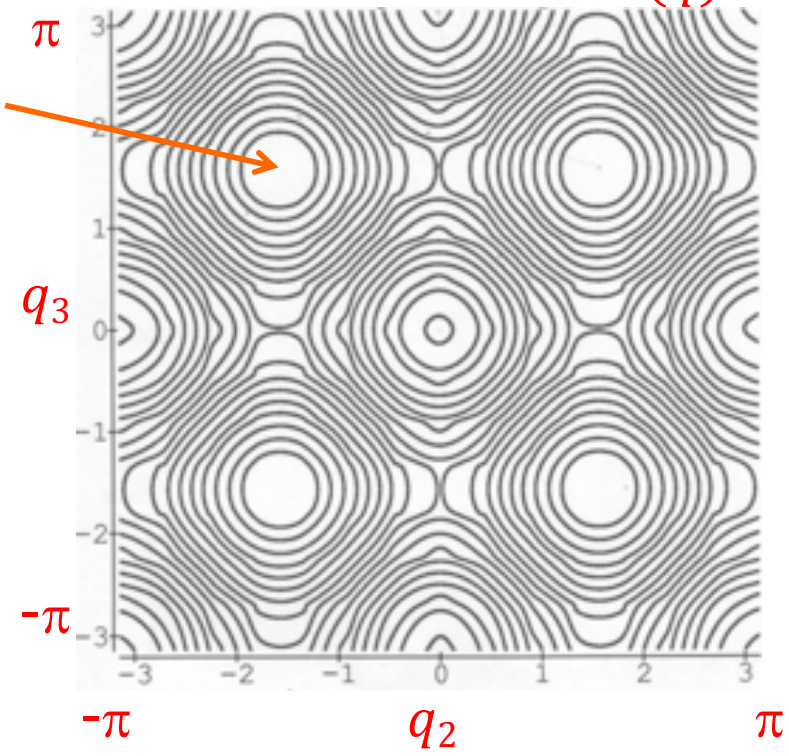
$$H(q) = \sin^2 q_2 + \sin^2 q_3$$

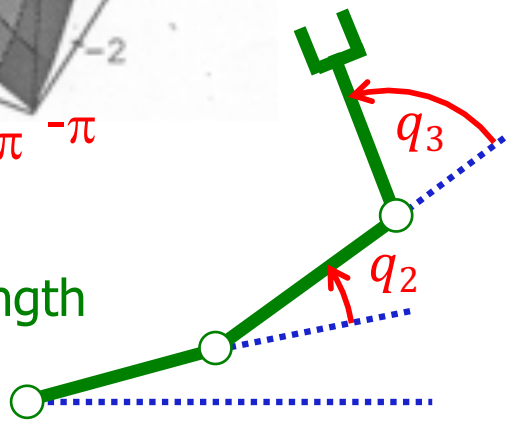
this $H$ is **not** $H_{\text{man}}$ but has the same minima




unconstrained maxima of $H(q)$

iso-level curves of $H(q)$



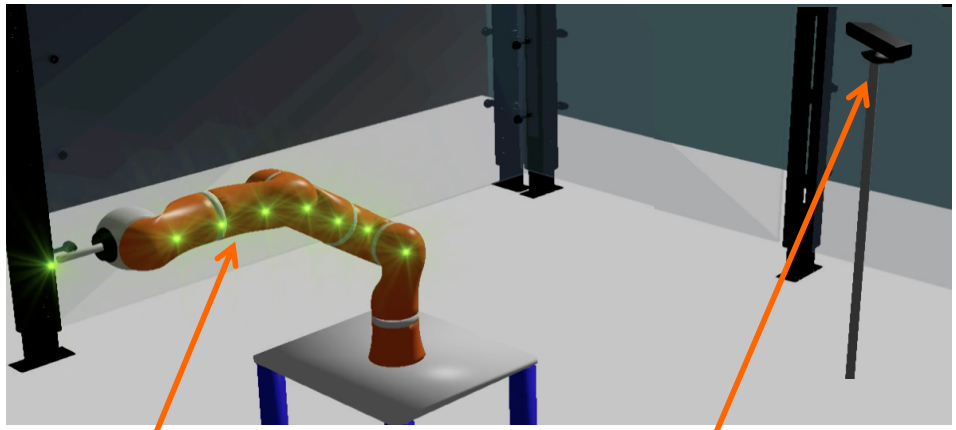




links of **equal** (unit) length

independent from $q_1$!

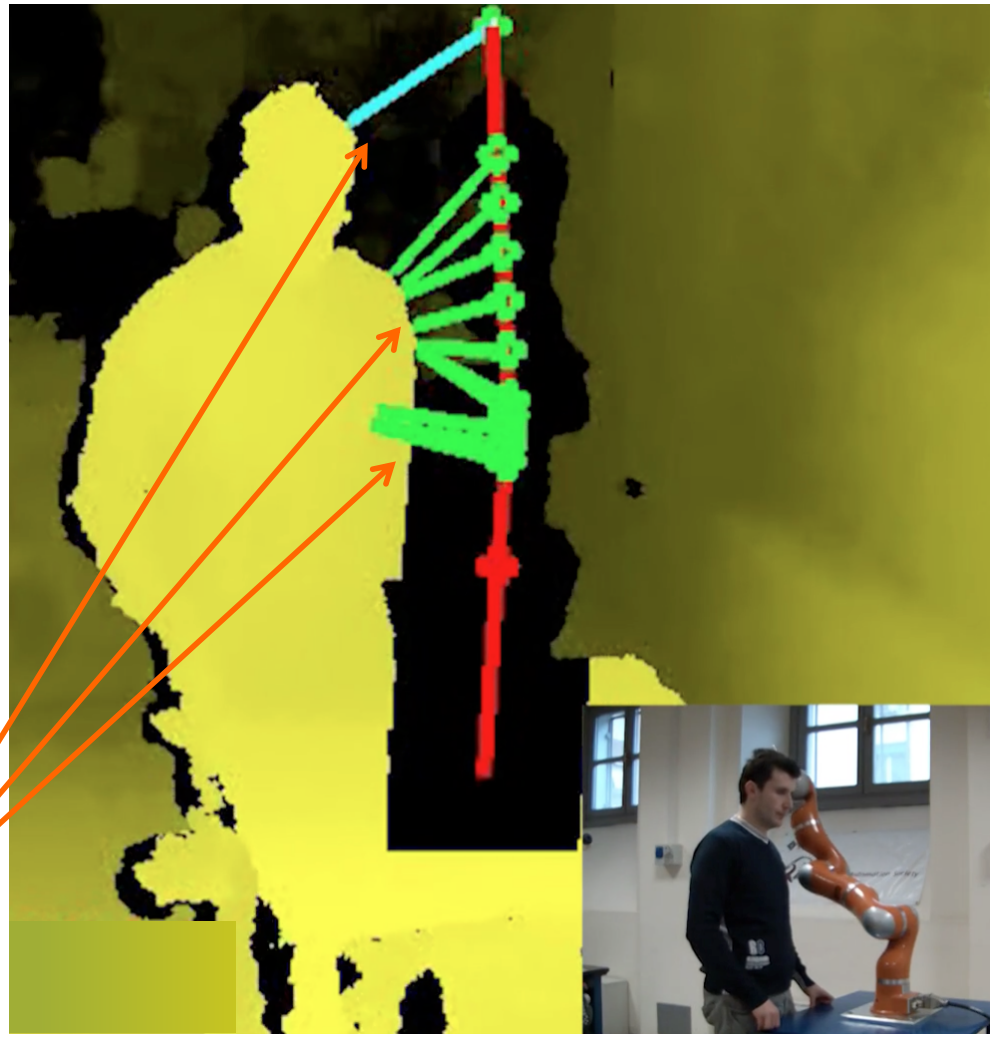
# Minimum distance computation

## in human-robot interaction



LWR4 robot with a finite number of control points $a(q)$ (8, including the E-E)

a Kinect sensor monitors the workspace giving the 3D position of points $b$ on obstacles that are fixed or moving (like humans)



distances in 3D (and then the clearance) are computed in this case as

$$\min_{\substack{a \in \{\text{control points}\} \\ b \in \text{human body}}} \|a(q) - b\|^2$$

# Comments on null-space methods


- the projection matrix $(I - J^{\#}J)$ has dimension $N \times N$, but only rank $N - M$ (if $J$ is full rank $M$), with some waste of information
- actual (efficient) evaluation of the solution

$$\dot{q} = J^{\#}\dot{r} + \left(I - J^{\#}J\right)\dot{q}_0 = \dot{q}_0 + J^{\#}(\dot{r} - J\dot{q}_0)$$

  but the pseudoinverse is needed anyway, and this is computationally intensive (SVD in the general case)
- in principle, the additional complexity of a redundancy resolution method should depend only on the redundancy degree $N - M$
- a constrained optimization method is available, which is known to be more efficient than the projected gradient (PG) —at least when the Jacobian has full rank …

# Decomposition of joint space


- if $\rho(J(q)) = M$, there exists a decomposition of the set of joints (possibly, after a reordering)

$$q = \begin{pmatrix} q_a \\ q_b \end{pmatrix} \begin{matrix} \} M \\ \} N-M \end{matrix} \quad \text{such that } \overbrace{J_a(q) = \frac{\partial f}{\partial q_a}}^{M \times M} \text{ is nonsingular}$$

- from the implicit function theorem, there exists an inverse function $g$

$$f(q_a, q_b) = r \quad \Rightarrow \quad q_a = g(r, q_b)$$

$$\text{with } \frac{\partial g}{\partial q_b} = -\left(\frac{\partial f}{\partial q_a}\right)^{-1} \frac{\partial f}{\partial q_b} = -J_a^{-1}(q) J_b(q)$$

- the $N - M$ variables $q_b$ can be selected independently (e.g., they are used for optimizing an objective function $H(q)$, "reduced" via the use of $g$ to a function of $q_b$ only)
- $q_a = g(r, q_b)$ is then chosen so as to correctly execute the task

# Reduced Gradient (RG)


- $H(q) = H(q_a, q_b) = H(g(r, q_b), q_b) = H'(q_b)$, with $r$ at current value
- the Reduced Gradient (w.r.t. $q_b$ only, but still keeping the effects of this choice into account) is

$$\nabla_{q_b} H' = [-(J_a^{-1} J_b)^T \quad I_{N-M}] \nabla_q H$$

($\neq \nabla_{q_b} H$ only!!)

$$\boxed{\nabla_{q_b} H' = 0}$$

is a "compact"
(i.e., $N - M$ dimensional)
necessary condition
of constrained optimality

- algorithm

$$\dot{q}_b = \nabla_{q_b} H'$$ ← step in the gradient direction of the reduced $(N - M)$-dim space

$$J_a \dot{q}_a + J_b \dot{q}_b = \dot{r}$$ ← satisfaction of the $M$-dim task constraints

$$\dot{q}_a = J_a^{-1}(\dot{r} - J_b \dot{q}_b)$$

# Comparison between PG and RG


- Projected Gradient (PG)

$$\dot{q} = J^{\#}\dot{r} + \left(I - J^{\#}J\right)\nabla_q H$$

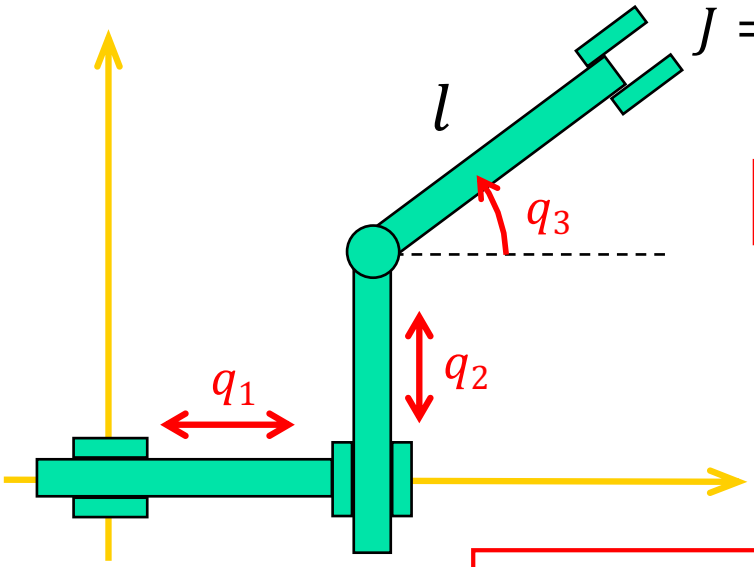
- Reduced Gradient (RG)

$$\dot{q} = \begin{pmatrix} \dot{q}_a \\ \dot{q}_b \end{pmatrix} = \begin{pmatrix} J_a^{-1} \\ 0 \end{pmatrix}\dot{r} + \begin{pmatrix} -J_a^{-1}J_b \\ I \end{pmatrix}\left(-(J_a^{-1}J_b)^T \quad I\right)\nabla_q H$$

- RG is analytically simpler and numerically faster than PG, but requires the search for a non-singular minor ($J_a$) of the robot Jacobian
- if $r$ = cost & $N - M = 1$ ⇒ same (unique) direction for $\dot{q}$, but RG has automatically a larger optimization step size
- else ⇒ RG and PG methods provide always different evolutions

# Analytic comparison

## PPR robot




$$J = \begin{pmatrix} 1 & 0 & -ls_3 \\ 0 & 1 & lc_3 \end{pmatrix} = (J_a \mid J_b) \quad q_a = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} \quad q_b = q_3$$

RG:

$$\dot{q} = \begin{pmatrix} J_a^{-1} \\ 0 \end{pmatrix} \dot{r} + \begin{pmatrix} -J_a^{-1} J_b \\ I \end{pmatrix} (-(J_a^{-1} J_b)^T \quad I) \nabla_q H$$

$$\dot{q} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix} \dot{r} + \begin{pmatrix} ls_3 \\ -lc_3 \\ 1 \end{pmatrix} (ls_3 \quad -lc_3 \quad 1) \nabla_q H$$

PG: $\dot{q} = J^{\#} \dot{r} + (I - J^{\#} J) \nabla_q H$

$$J^{\#} = \frac{1}{1 + l^2} \begin{pmatrix} 1 + l^2 c_3^2 & l^2 s_3 c_3 \\ l^2 s_3 c_3 & 1 + l^2 s_3^2 \\ -ls_3 & lc_3 \end{pmatrix} \quad I - J^{\#} J = \frac{1}{1 + l^2} \begin{pmatrix} l^2 s_3^2 & l^2 s_3 c_3 & ls_3 \\ l^2 s_3 c_3 & l^2 c_3^2 & -lc_3 \\ ls_3 & -lc_3 & 1 \end{pmatrix}$$

always < 1!!

# Joint range limits

$$q = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix} \theta = T\theta$$

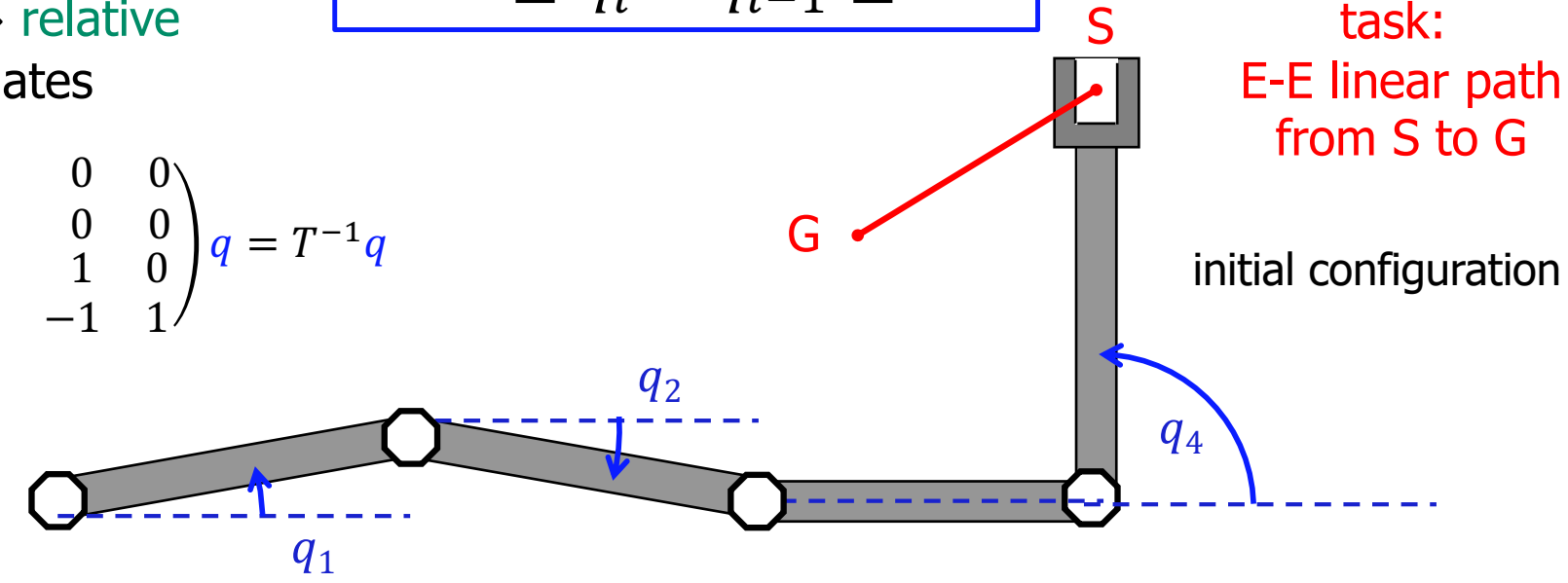
$$-90° \leq \theta_i \leq 90°$$

$$\Updownarrow$$

$$-90° \leq q_i - q_{i-1} \leq 90°$$

absolute ⇔ relative coordinates

$$\theta = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{pmatrix} q = T^{-1} q$$



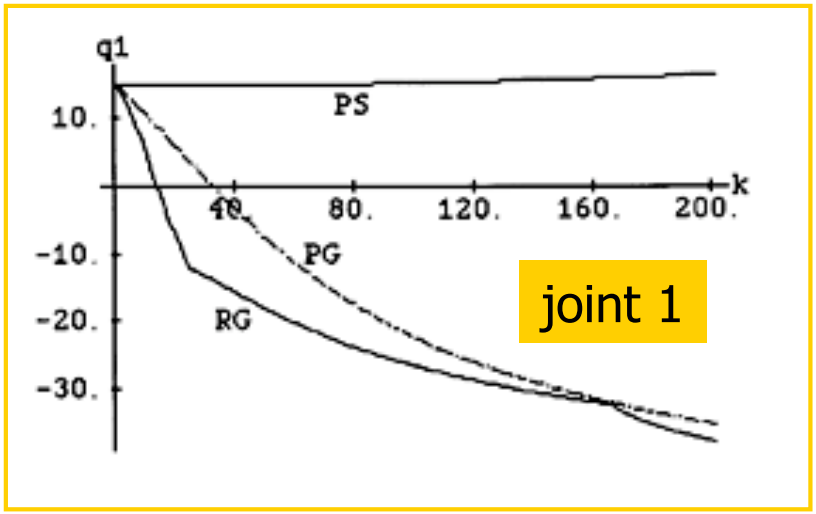

task:
E-E linear path
from S to G

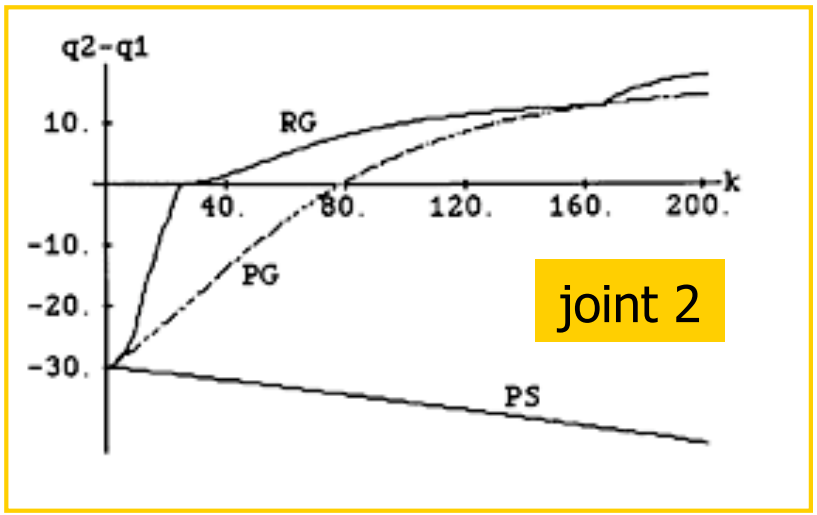
initial configuration

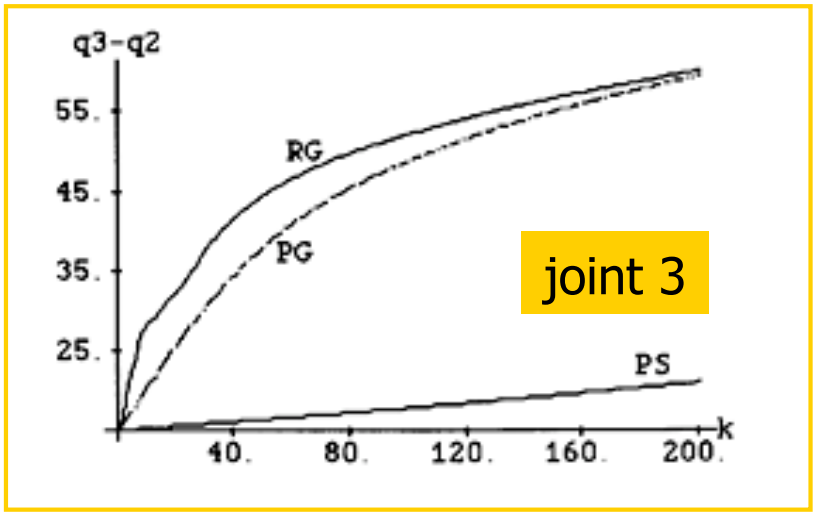
numerical comparison among pseudoinverse (PS), projected gradient (PG), and reduced gradient (RG) methods

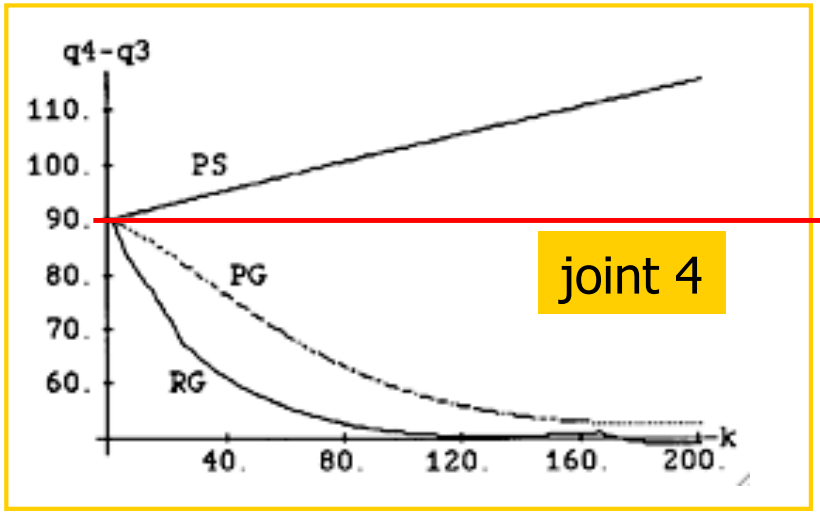
# Numerical results

## minimizing distance from mid joint range



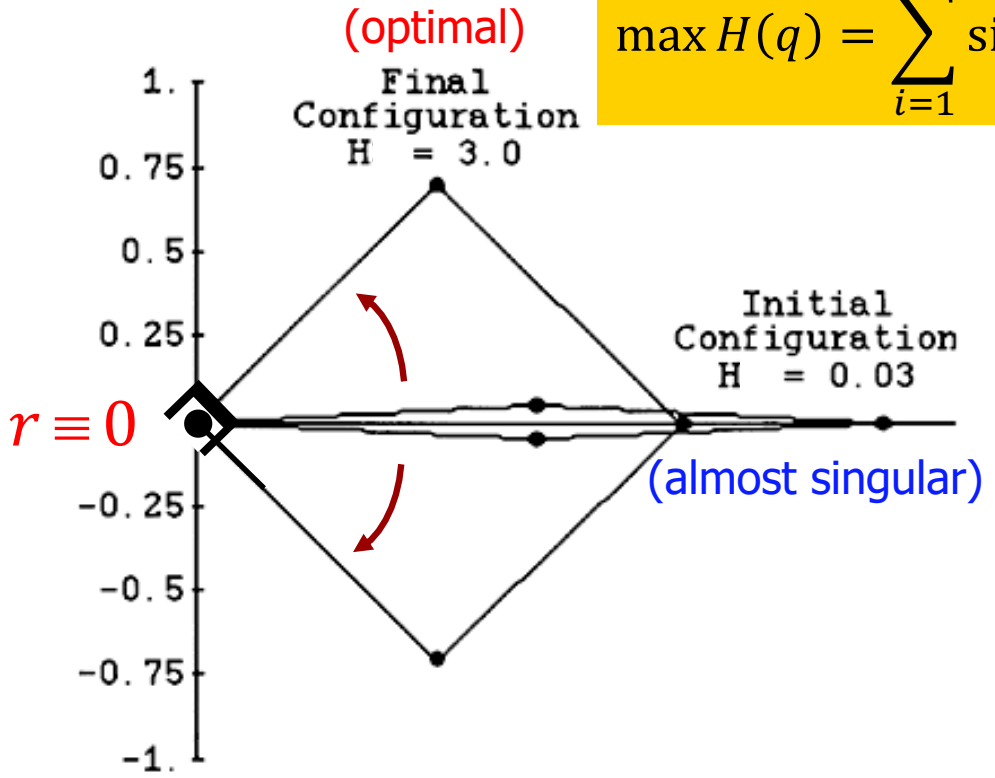




steps of numerical simulation

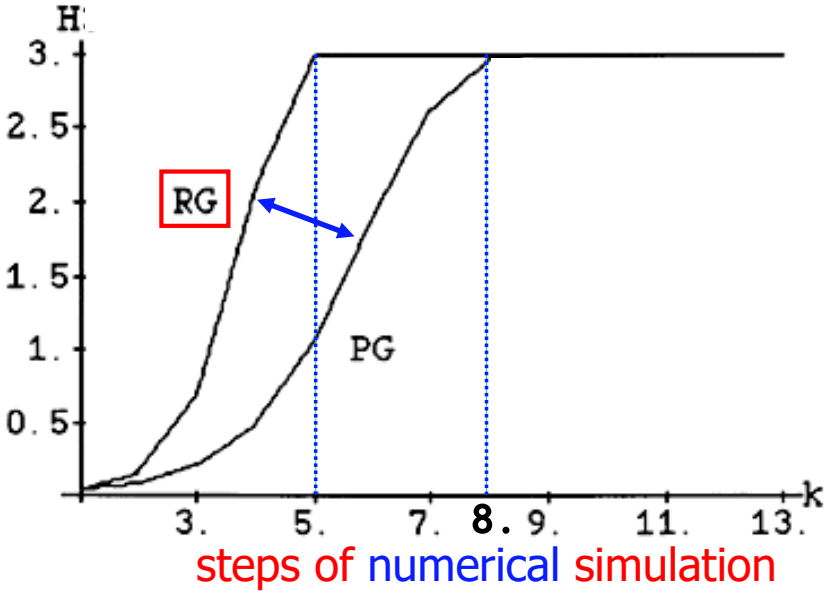
# Numerical results

## self-motion for escaping singularities




$$\max H(q) = \sum_{i=1}^{3} \sin^2(q_{i+1} - q_i)$$

this function is NOT the manipulability index, but has the same minima (= 0)




RG is faster than PG
(keeping the same accuracy on $r$)

# 3 Task augmentation methods


- an auxiliary task is added (task augmentation)

$$s \updownarrow \boxed{f_y(q) = y} \qquad S \leq N - M$$

corresponding to some desirable feature for the solution

$$r_A = \begin{pmatrix} r \\ y \end{pmatrix} = \begin{pmatrix} f(q) \\ f_y(q) \end{pmatrix} \Rightarrow \dot{r}_A = \begin{pmatrix} J(q) \\ J_y(q) \end{pmatrix} \dot{q} = J_A(q)\dot{q}$$

$J_A$: $N$ columns, $M + S$ rows

- a solution is chosen still in the form of a generalized inverse

$$\boxed{\dot{q} = K_A(q)\dot{r}_A}$$

or by adding a term in the null space of the augmented Jacobian matrix $J_A$

# Augmenting the task …

- advantage: better shaping of the inverse kinematic solution
- disadvantage: algorithmic singularities are introduced when

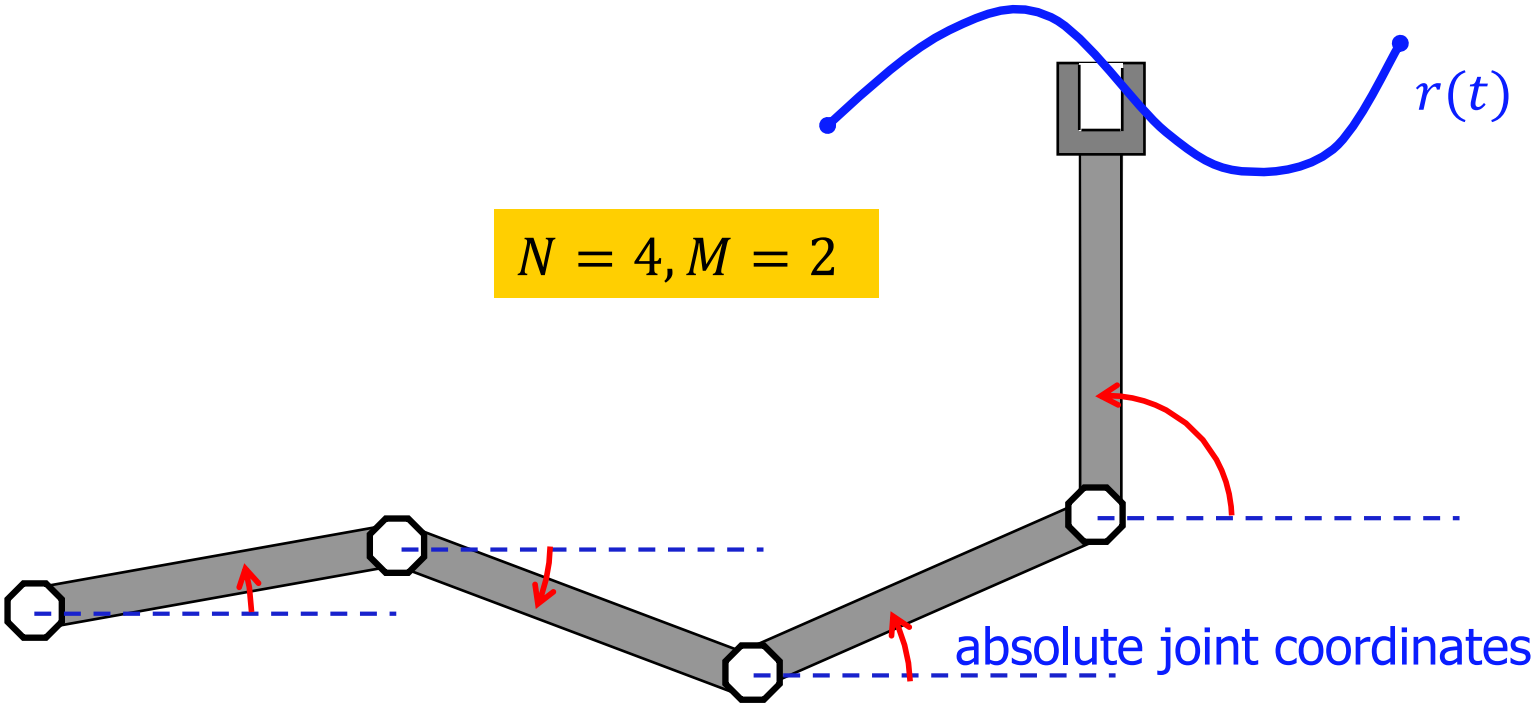
$$\rho(J) = M \quad \rho(J_y) = S \quad \text{but} \quad \rho(J_A) < M + S$$

to avoid this, it should be always $\mathcal{R}(J^T) \cap \mathcal{R}(J_y^T) = \emptyset$

difficult to be obtained globally!

rows of $J$ AND rows of $J_y$
are all together linearly independent

# Augmented task
## example




$$f_y(q) = q_4 = \pi/2 \quad (S = 1)$$

last link is to be held vertical…

# Extended Jacobian ($S = N–M$)


- square $J_A$: in the absence of algorithmic singularities, we can choose

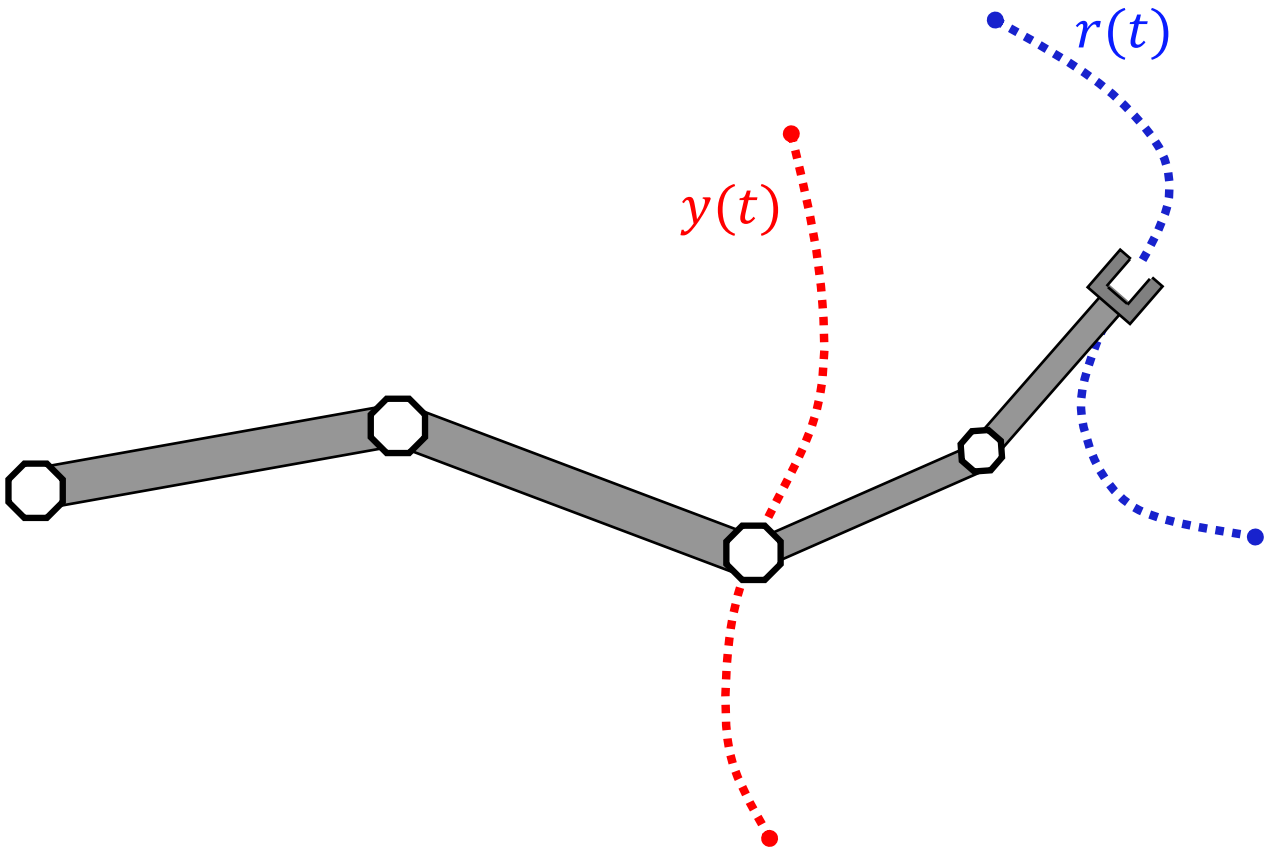
$$\dot{q} = J_A^{-1}(q)\dot{r}_A$$

- the scheme is then repeatable
  - provided no singularities are encountered during a complete task cycle*
- when the $N - M$ conditions $f_y(q) = 0$ correspond to necessary (and sufficient) conditions for constrained optimality of a given objective function $H(q)$ (see RG method, slide #36), this scheme guarantees that constrained optimality is locally preserved during task execution
- in the vicinity of algorithmic singularities, the execution of both the original task as well as the auxiliary task(s) are affected by errors (when using DLS inversion)

* there exists an unexpected phenomenon in some 3R manipulators having "generic" kinematics: the robot may sometimes perform a pose change after a full cycle, even if no singularity has been encountered during motion (see J. Burdick, *Mech. Mach. Theory*, 30(1), 1995)

# Extended Jacobian
## example

MACRO-MICRO manipulator




$N = 4, M = 2$

$$\dot{r} = J(q_1, \dots, q_4)\dot{q}$$
$$\dot{y} = J_y(q_1, q_2)\dot{q}$$

$$J_A = \left(\begin{array}{c|c} * & * \\ * & 0 \end{array}\right) \quad 4\times4$$

# Task Priority

if the original (primary) task $\dot{r}_1 = J_1(q)\dot{q}$ has higher priority than the auxiliary (secondary) task $\dot{r}_2 = J_2(q)\dot{q}$

- we first address the task with highest priority

$$\dot{q} = J_1^{\#}\dot{r}_1 + \left(I - J_1^{\#}J_1\right)v_1$$

- and then choose $v_1$ so as to satisfy, if possible, also the secondary (lower priority) task

$$\dot{r}_2 = J_2\dot{q} = J_2\,J_1^{\#}\dot{r}_1 + J_2\left(I - J_1^{\#}J_1\right)v_1 = J_2\,J_1^{\#}\dot{r}_1 + J_2P_1v_1$$

the general solution for $v_1$ takes the usual form

$$v_1 = (J_2P_1)^{\#}\left(\dot{r}_2 - J_2\,J_1^{\#}\dot{r}_1\right) + \left(I - (J_2P_1)^{\#}(J_2P_1)\right)v_2$$

$v_2$ is available for the execution of further tasks of lower (ordered) priorities

# Task Priority (continue)

- substituting the expression of $v_1$ in $\dot{q}$

$$\dot{q} = J_1^{\#}\dot{r}_1 + P_1(J_2P_1)^{\#}\left(\dot{r}_2 - J_2J_1^{\#}\dot{r}_1\right) + P_1\left(I - (J_2P_1)^{\#}(J_2P_1)\right)v_2$$

$P(BP)^{\#} = (BP)^{\#}$ when matrix $P$ is idempotent and symmetric

$= (J_2P_1)^{\#}$

possibly $= 0$

- main advantage: highest priority task is ideally no longer affected by algorithmic singularities (error is restricted only to secondary task)

task 1: follow —

task 2: vertical third link