3 Task augmentation methods

- an auxiliary task is added (task augmentation)

number of additional components $\leftarrow S \updownarrow$

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

corresponding to some desirable feature for the solution

$$r_A = \begin{pmatrix} r \\ y \end{pmatrix} = \begin{pmatrix} f(q) \\ f_y(q) \end{pmatrix} \Rightarrow \dot{r}_A = \begin{pmatrix} J(q) \\ J_y(q) \end{pmatrix} \dot{q} = J_A(q)\dot{q} \qquad \underbrace{\boxed{J_A}}_{N} \} M + S$$

- 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

min 36:00

- disadvantage: algorithmic singularities are introduced when

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

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

difficult to be obtained globally!

$\Updownarrow$

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

Augmented task

example

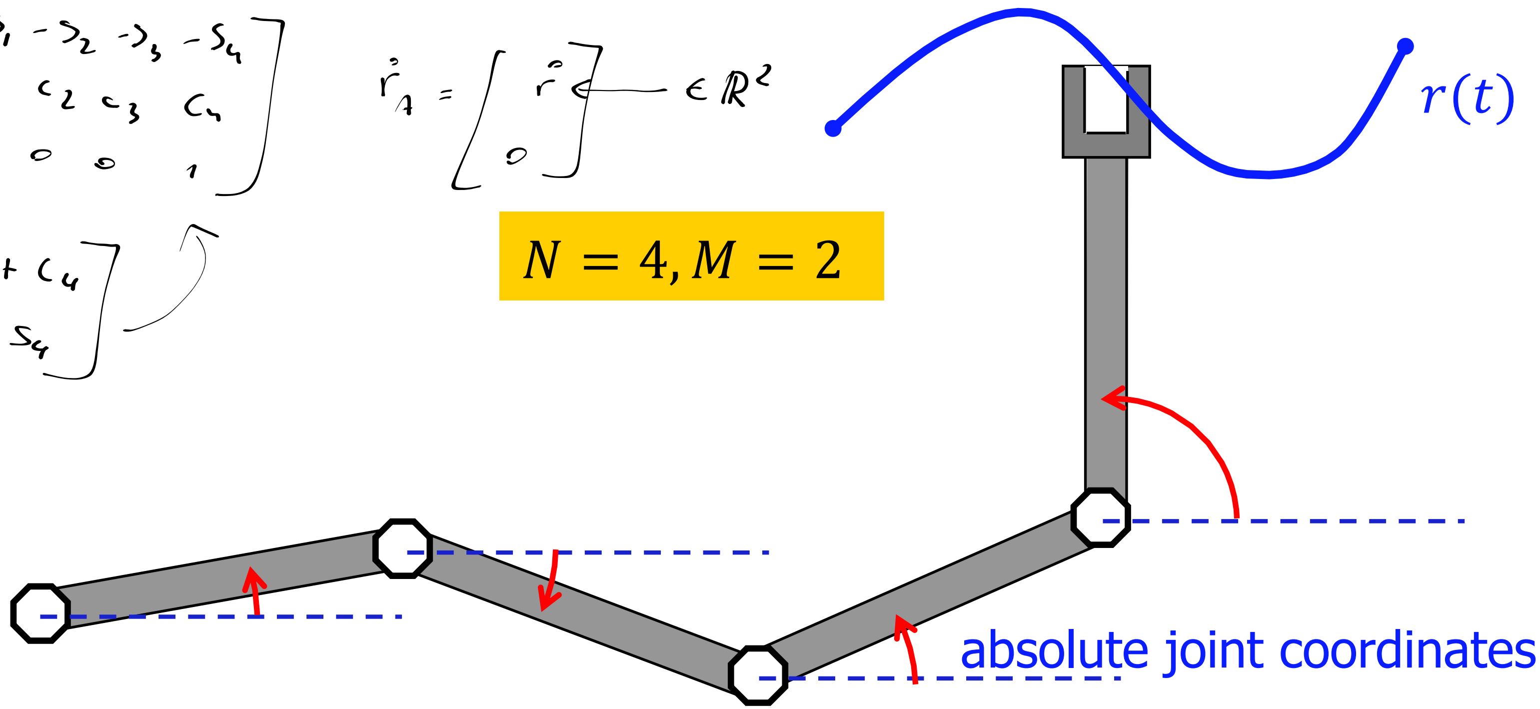
$l_i = 1 \quad i = 1, \dots, 4$

$$J_A = \begin{bmatrix} J \\ J_y \end{bmatrix} = \begin{bmatrix} -s_1 & -s_2 & -s_3 & -s_4 \\ c_1 & c_2 & c_3 & c_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\dot{r}_A = \begin{bmatrix} \dot{r} \\ 0 \end{bmatrix} \quad \in \mathbb{R}^2$$

$$p = \begin{bmatrix} c_1 + c_2 + c_3 + c_4 \\ s_1 + s_2 + s_3 + s_4 \end{bmatrix}$$

$N = 4, M = 2$



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

last link is to be held vertical…

Extended Jacobian ($S = N{-}M$)

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

$$\dot{q} = J_A^{-1}(q)\dot{r}_A \longrightarrow$$ DLS in case of algosing

- 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

min 45:00

$f_y(q) = [-(J_a^{-1}(q)\ J_b(q))\ \ I]\,\nabla_q H(q) = 0$

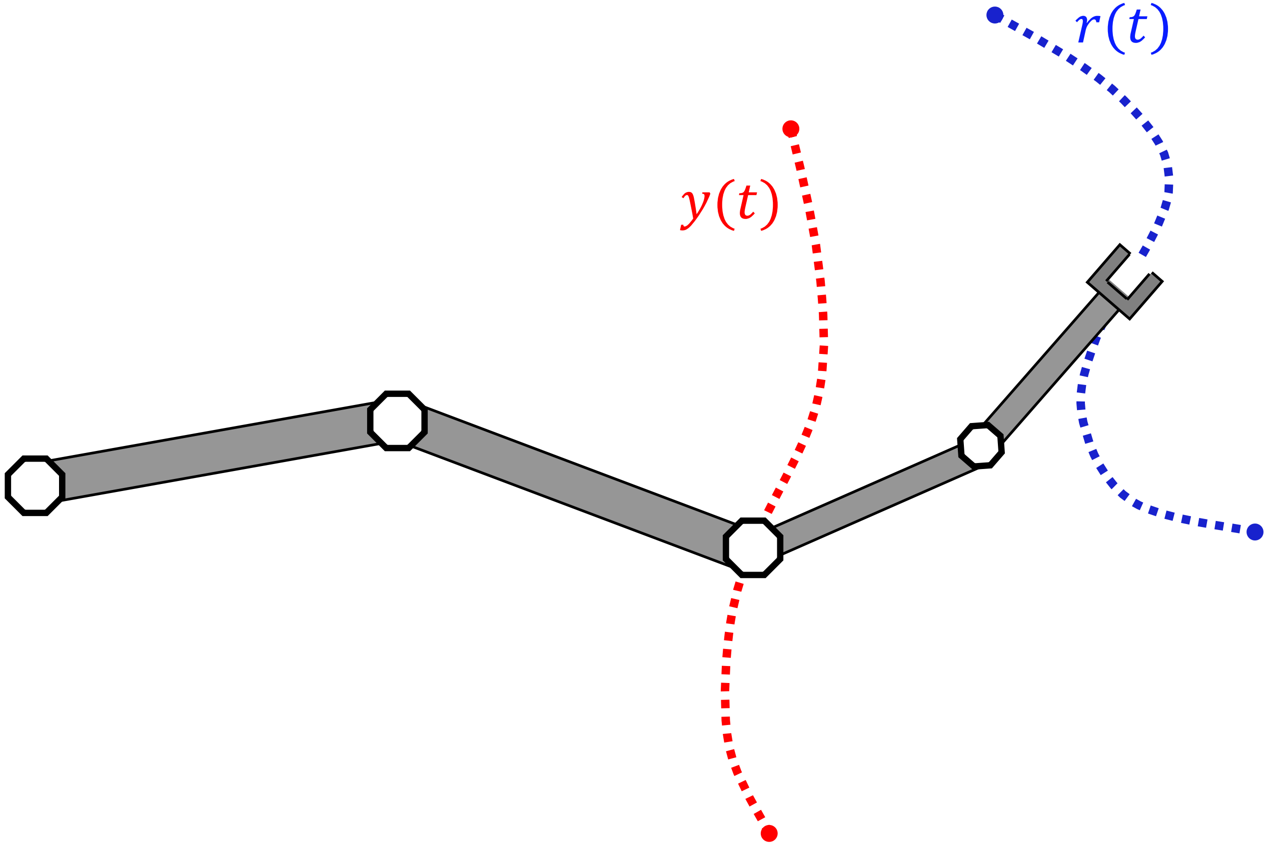
- in the vicinity of algorithmic singularities, the execution of both the original task as well as the auxiliary task(s) are affected by errors (when using DLS inversion)

* 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



$N = 4, M = 2$

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

$$\dot{y} = J_y(q_1, q_2)\dot{q}$$

⇓

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

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 $\dot{q}$

$$\boxed{\dot{q} = J_1^{\#}\dot{r}_1 + P_1(J_2P_1)^{\#}\left(\dot{r}_2 - J_2 J_1^{\#}\dot{r}_1\right) + P_1\left(I - (J_2P_1)^{\#}(J_2P_1)\right)v_2}$$

$P(BP)^{\#} = (BP)^{\#}$
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: keep third link vertical