Robotics 1

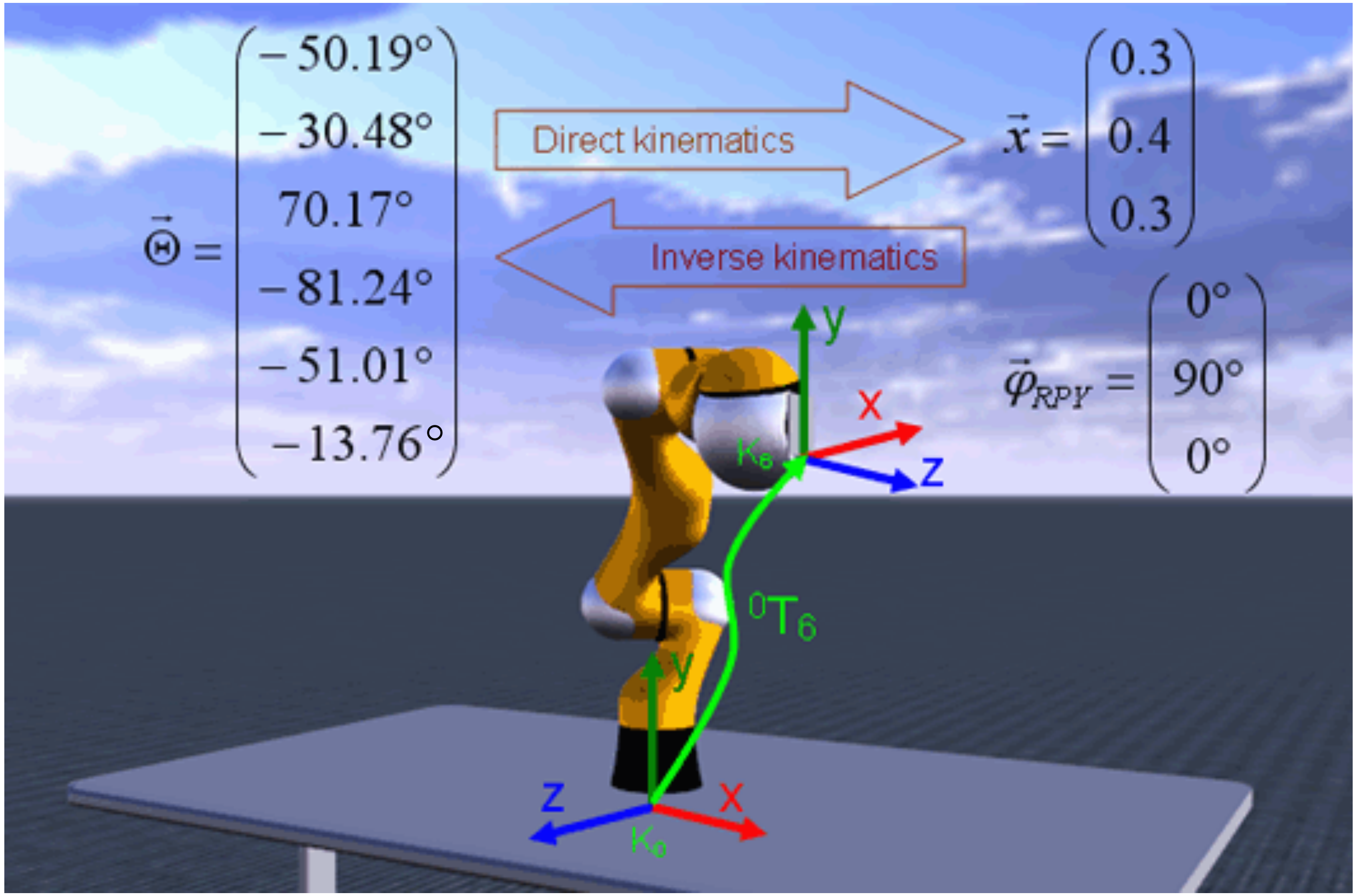
Inverse kinematics

Prof. Alessandro De Luca

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Inverse kinematics
what are we looking for?



direct kinematics is always unique;
how about inverse kinematics for this 6R robot?

Inverse kinematics problem

- "given a desired end-effector pose (position + orientation), find the values of the joint variables that will realize it"
- a synthesis problem, with input data in the form
  - $T = \left[\begin{array}{c:c} R & p \\ \hdashline 000 & 1 \end{array}\right]$
  - $r = \begin{bmatrix} p \\ \phi \end{bmatrix}$, or any other task vector

classical formulation:
inverse kinematics for a given end-effector pose

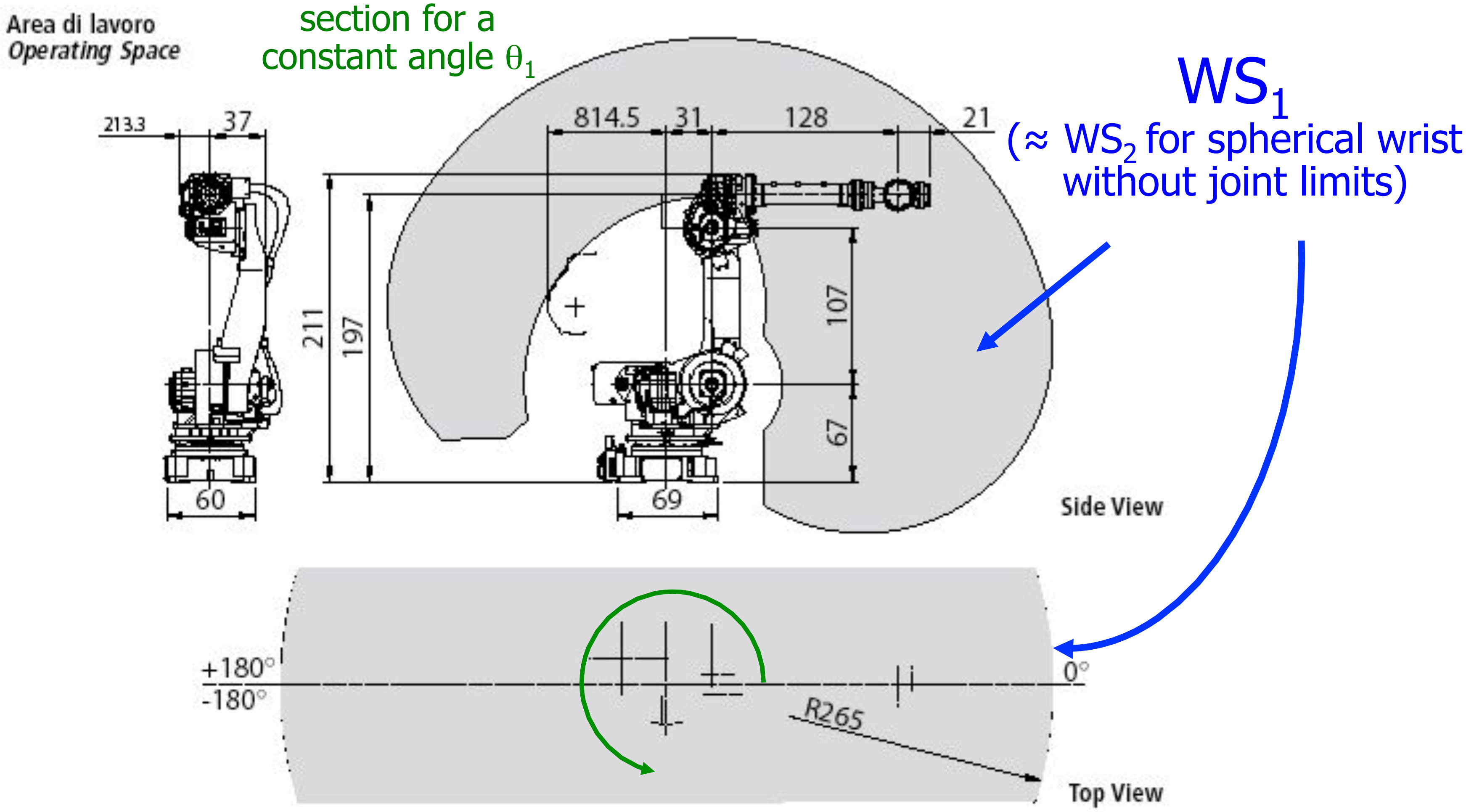
generalized formulation:
inverse kinematics for a given value of task variables

- a typical nonlinear problem
  - existence of a solution (workspace definition)
  - uniqueness/multiplicity of solutions
  - solution methods

Solvability and robot workspace

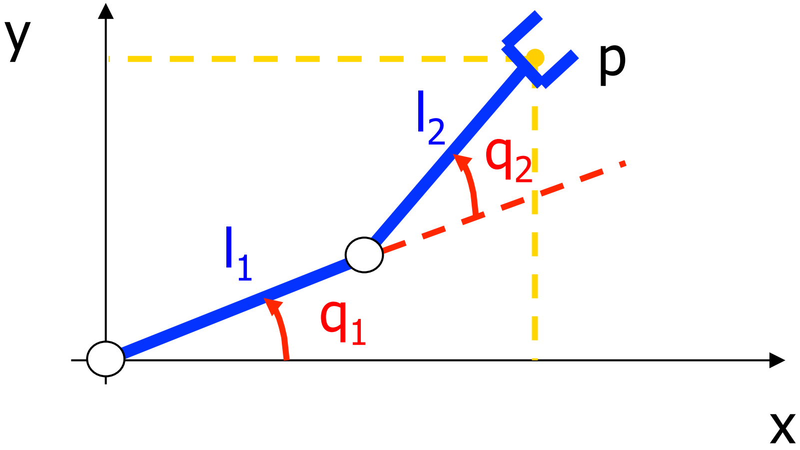
- primary workspace WS_1: set of all positions p that can be reached with at least one orientation (ϕ or R)
  - out of WS_1 there is no solution to the problem
  - for $p \in WS_1$ and a suitable ϕ (or R) there is at least one solution
- secondary (or *dexterous*) workspace WS_2: set of positions p that can be reached with any orientation (among those feasible for the robot direct kinematics)
  - for $p \in WS_2$ there is at least one solution for any feasible ϕ (or R)
- $WS_2 \subseteq WS_1$

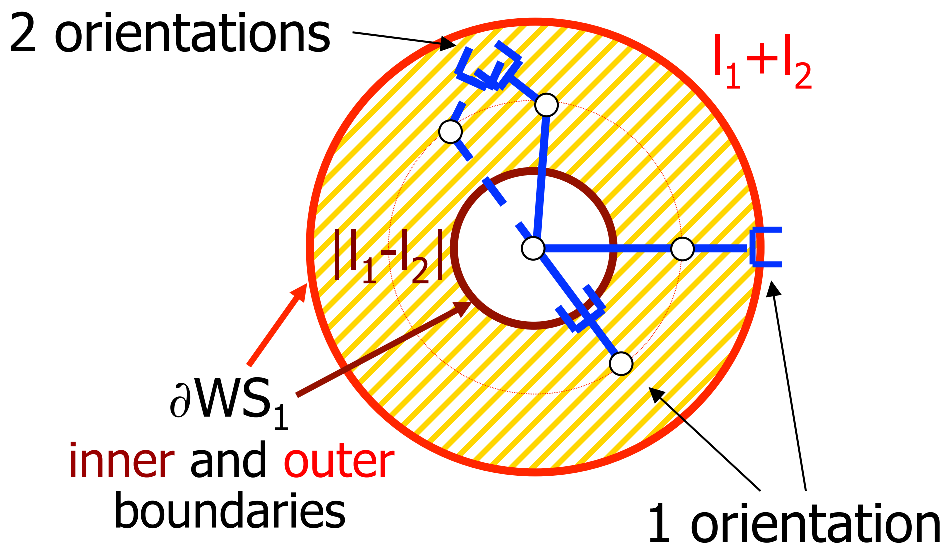
Workspace of Fanuc R-2000i/165F



rotating the base joint angle θ_1

Workspace of planar 2R arm





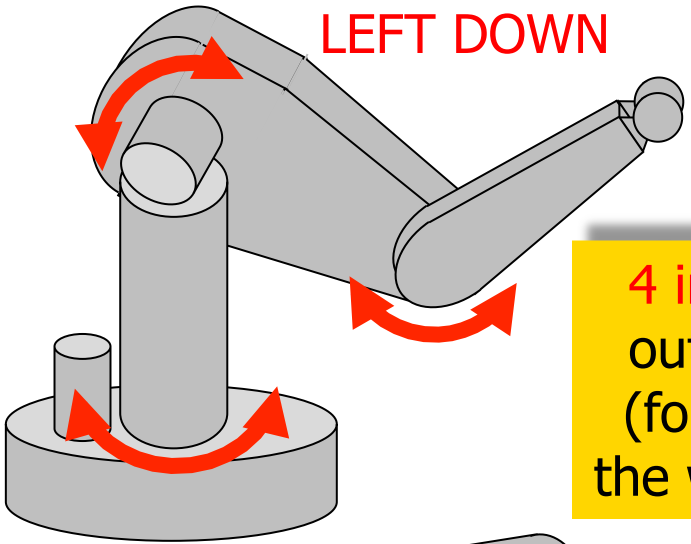
- if $l_1 \neq l_2$
  - $WS_1 = \{p \in R^2: |l_1-l_2| \leq \|p\| \leq l_1+l_2\}$
  - $WS_2 = \emptyset$
- if $l_1 = l_2 = \ell$
  - $WS_1 = \{p \in R^2: \|p\| \leq 2\ell\}$
  - $WS_2 = \{p = 0\}$ (infinite number of feasible orientations at the origin)

Wrist position and E-E pose

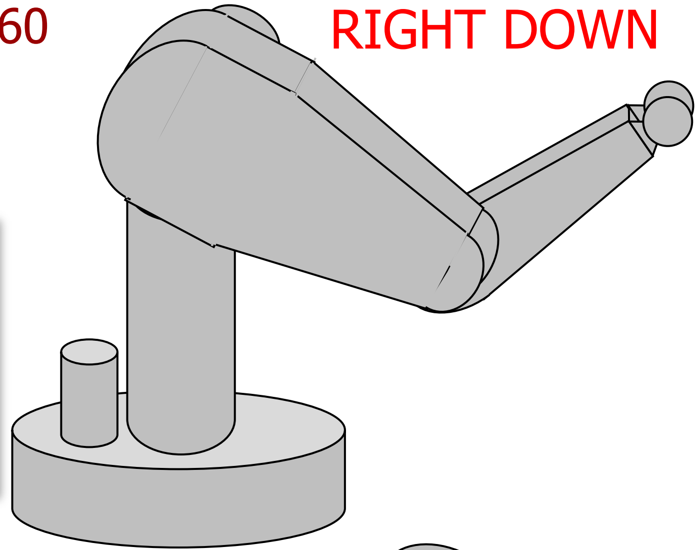
inverse solutions for an articulated 6R robot

Unimation PUMA 560

LEFT DOWN

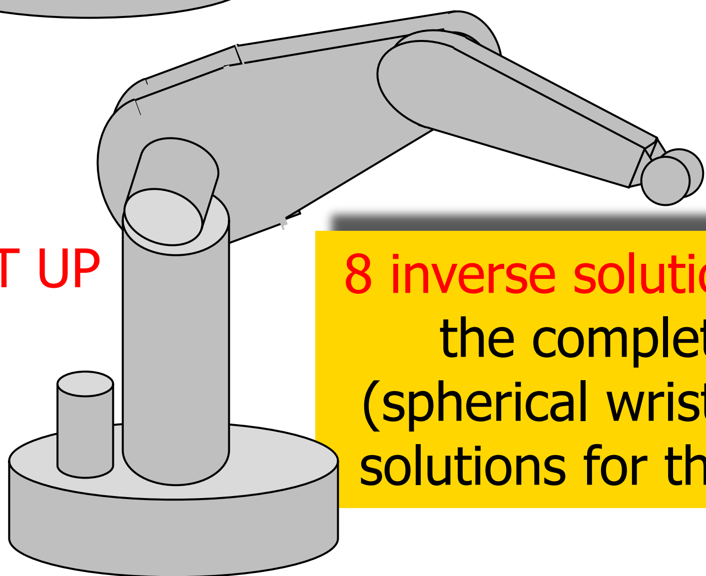


RIGHT DOWN

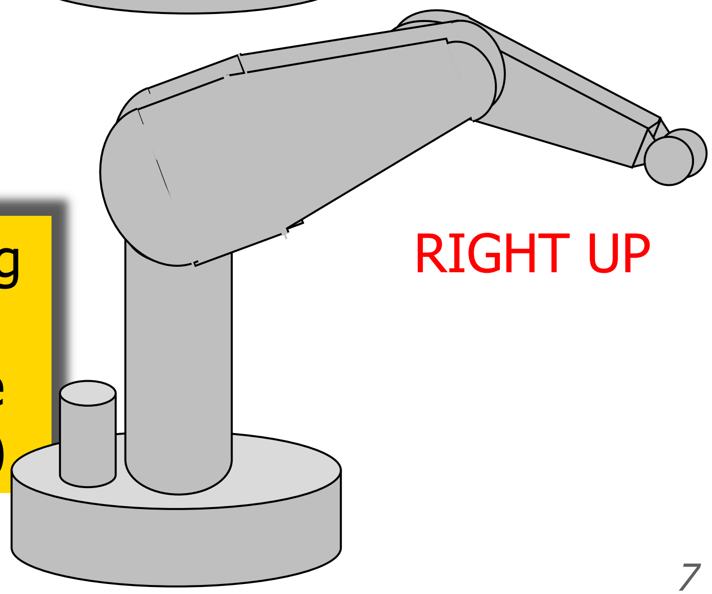


4 inverse solutions
out of singularities
(for the position of
the wrist center only)

LEFT UP

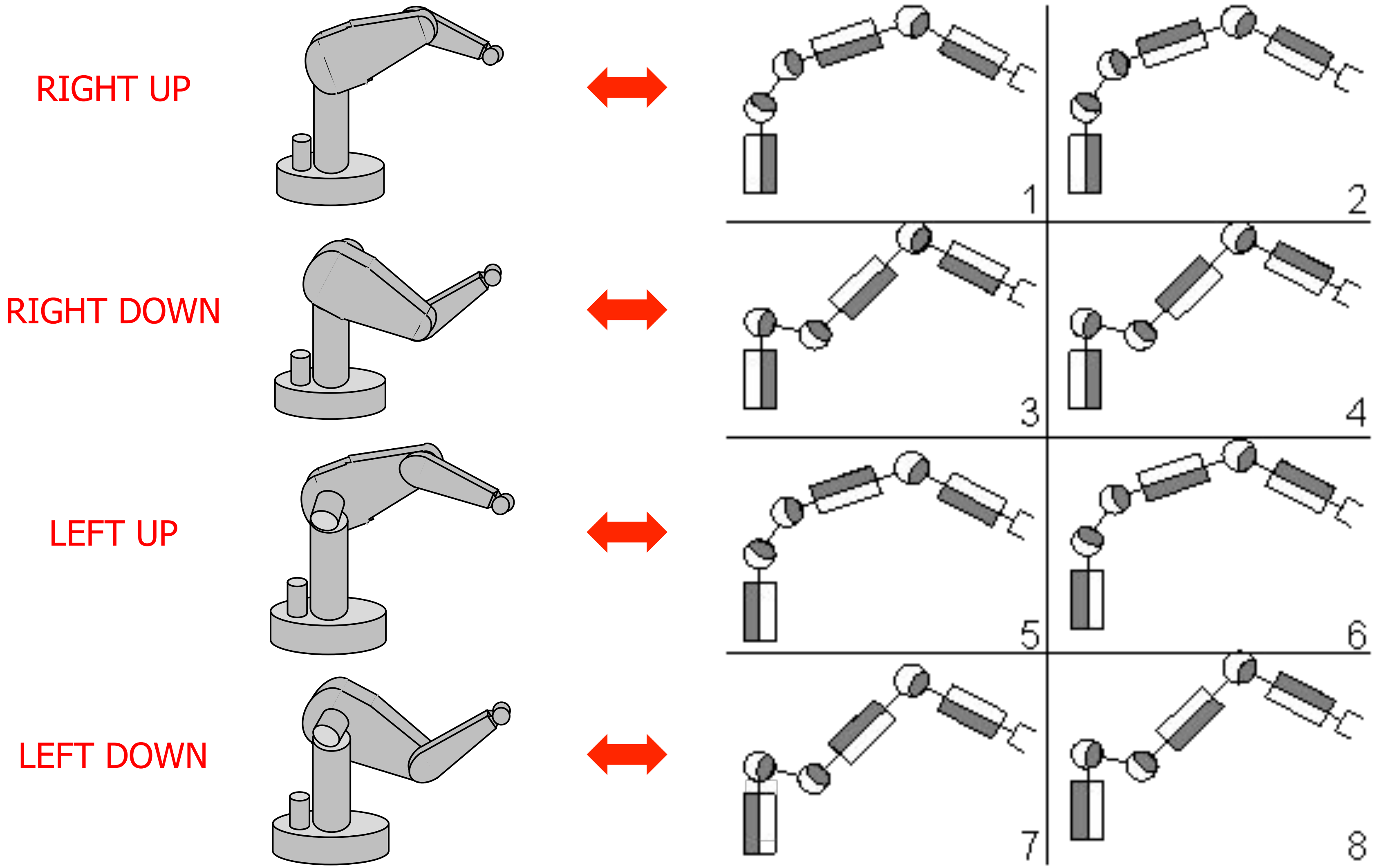


RIGHT UP



8 inverse solutions considering
the complete E-E pose
(spherical wrist: 2 alternative
solutions for the last 3 joints)

Counting and visualizing the 8 solutions
to the inverse kinematics of a Unimation Puma 560



Inverse kinematic solutions of UR10

6-dof Universal Robot UR10, with non-spherical wrist

video (slow motion)

desired pose

$$p = \begin{pmatrix} -0.2373 \\ -0.0832 \\ 1.3224 \end{pmatrix} \qquad R = \begin{pmatrix} \sqrt{3}/2 & 0.5 & 0 \\ -0.5 & \sqrt{3}/2 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

[m]

configuration at start

$$q = \begin{pmatrix} \pi/3 & -2\pi/3 & \pi/6 & 0 & \pi/2 & 0 \end{pmatrix}^T$$

[rad]

The 8 inverse kinematic solutions of UR10

shoulderRight
wristDown
elbowUp

$$q = \begin{pmatrix} 1.0472 \\ -1.2833 \\ -0.7376 \\ -2.6915 \\ -1.5708 \\ 3.1416 \end{pmatrix}$$

shoulderRight
wristDown
elbowDown

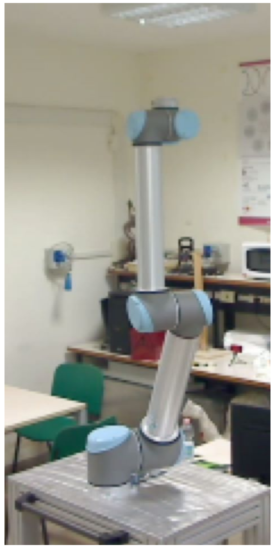
$$q = \begin{pmatrix} 1.0472 \\ -1.9941 \\ 0.7376 \\ -2.8273 \\ -1.5708 \\ 3.1416 \end{pmatrix}$$

shoulderRight
wristUp
elbowUp

$$q = \begin{pmatrix} 1.0472 \\ -1.5894 \\ -0.5236 \\ 0.5422 \\ 1.5708 \\ 0 \end{pmatrix}$$

shoulderRight
wristUp
elbowDown

$$q = \begin{pmatrix} 1.0472 \\ -2.0944 \\ 0.5236 \\ 0 \\ 1.5708 \\ 0 \end{pmatrix}$$



shoulderLeft
wristDown
elbowDown

$$q = \begin{pmatrix} 2.7686 \\ -1.0472 \\ -0.5236 \\ 3.1416 \\ -1.5708 \\ 1.4202 \end{pmatrix}$$

shoulderLeft
wristDown
elbowUp

$$q = \begin{pmatrix} 2.7686 \\ -1.5522 \\ 0.5236 \\ 2.5994 \\ -1.5708 \\ 1.4202 \end{pmatrix}$$

shoulderLeft
wristUp
elbowDown

$$q = \begin{pmatrix} 2.7686 \\ -1.1475 \\ -0.7376 \\ 0.3143 \\ 1.5708 \\ -1.7214 \end{pmatrix}$$

shoulderLeft
wristUp
elbowUp

$$q = \begin{pmatrix} 2.7686 \\ -1.8583 \\ 0.7376 \\ -0.4501 \\ 1.5708 \\ -1.7214 \end{pmatrix}$$

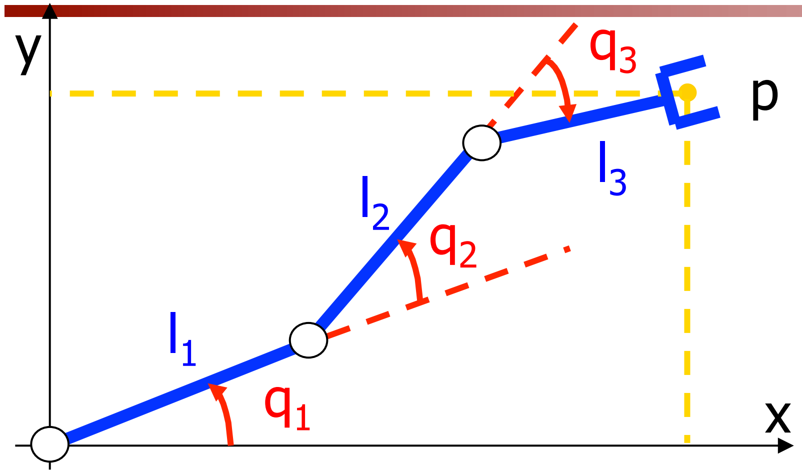
Multiplicity of solutions
some examples

- E-E positioning of a planar 2R robot arm
  - 2 regular solutions in WS_1
  - 1 solution on ∂WS_1
  - for $l_1 = l_2$: ∞ solutions in WS_2

  } *singular* solutions
- E-E positioning of an articulated elbow-type 3R robot arm
  - 4 regular solutions in WS_1 (with singular cases yet to be investigated..)
- spatial 6R robot arms
  - $\leq$ 16 distinct solutions, out of singularities: this "upper bound" of solutions was shown to be attained by a particular instance of "orthogonal" robot, i.e., with twist angles $\alpha_i = 0$ or $\pm\pi/2$ ($\forall i$)
  - analysis based on algebraic transformations of robot kinematics
    - transcendental equations are transformed into a single polynomial equation of one variable
    - seek for an equivalent polynomial equation of the least possible degree

A planar 3R arm

workspace and number/type of inverse solutions



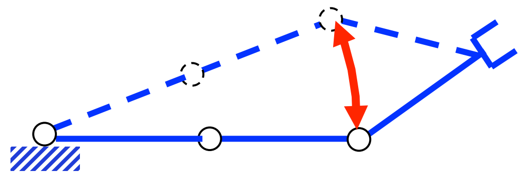
$l_1 = l_2 = l_3 = \ell$

$WS_1 = \{p \in R^2: \|p\| \leq 3\ell\}$

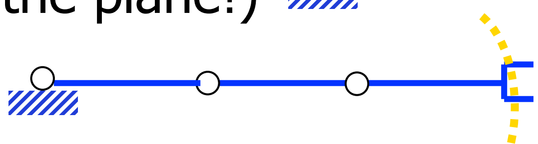
$WS_2 = \{p \in R^2: \|p\| \leq \ell\}$

any planar orientation is feasible in WS_2

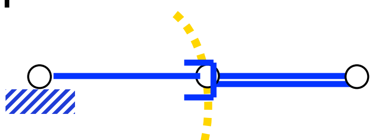
1. in WS_1 : ∞^1 regular solutions (except for 2. and 3.), at which the E-E can take a *continuum* of ∞ orientations (but *not all* orientations in the plane!)

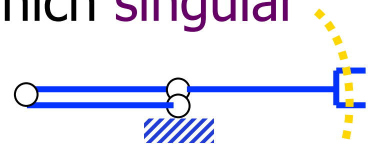


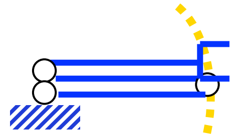
2. if $\|p\| = 3\ell$: only 1 solution, singular



3. if $\|p\| = \ell$: ∞^1 solutions, 3 of which singular







4. if $\|p\| < \ell$: ∞^1 regular solutions (never singular)

Multiplicity of solutions
summary of the general cases

- if $m = n$
  - $\nexists$ solutions
  - a finite number of solutions (regular/generic case)
  - "degenerate" solutions: infinite or finite set, but anyway different in number from the generic case (singularity)
- if $m < n$ (robot is redundant for the kinematic task)
  - $\nexists$ solutions
  - ∞^{n-m} solutions (regular/generic case)
  - a finite or infinite number of singular solutions
- use of the term singularity will become clearer when dealing with differential kinematics
  - instantaneous velocity mapping from joint to task velocity
  - lack of full rank of the associated $m \times n$ Jacobian matrix $J(q)$

Dexter robot (8R arm)

- m = 6 (position and orientation of E-E)
- n = 8 (all revolute joints)
- ∞^2 inverse kinematic solutions (redundancy degree = n-m = 2)

video

exploring inverse kinematic solutions by a self-motion

Solution methods

ANALYTICAL solution
(in closed form)

$\rightleftharpoons$

NUMERICAL solution
(in iterative form)

- preferred, if it can be found*
- use ad-hoc geometric inspection
- algebraic methods (solution of polynomial equations)
- systematic ways for generating a reduced set of equations to be solved

* sufficient conditions for 6-dof arms

- 3 consecutive rotational joint axes are incident (e.g., spherical wrist), or
- 3 consecutive rotational joint axes are parallel

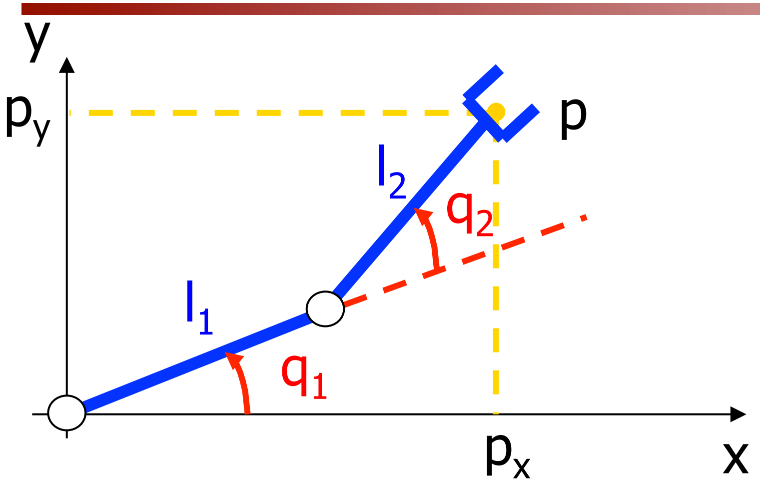
D. Pieper, PhD thesis, Stanford University, 1968

- certainly needed if n>m (redundant case), or at/close to singularities
- slower, but easier to be set up
- in its basic form, it uses the (analytical) Jacobian matrix of the direct kinematics map

$$J_r(q) = \frac{\partial f_r(q)}{\partial q}$$

- Newton method, Gradient method, and so on…

Inverse kinematics of planar 2R arm



direct kinematics

$$p_x = l_1\, c_1 + l_2\, c_{12}$$

$$\underbrace{p_y}_{\text{data}} = l_1\, s_1 + l_2\, s_{12}$$

q_1, q_2 unknowns

"squaring and summing" the equations of the direct kinematics

$$p_x^2 + p_y^2 - (l_1^2 + l_2^2) = 2\, l_1\, l_2\, (c_1\, c_{12} + s_1\, s_{12}) = 2\, l_1\, l_2\, c_2$$

and from this

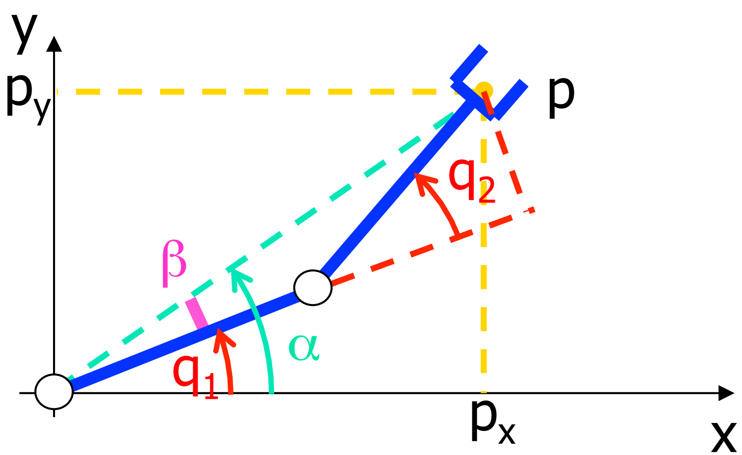
in analytical form

$$c_2 = (p_x^2 + p_y^2 - l_1^2 - l_2^2) / \, 2\, l_1\, l_2, \quad s_2 = \pm\sqrt{1 - c_2^2} \Rightarrow \boxed{q_2 = \text{ATAN2}\,\{s_2, c_2\}}$$

must be in [-1,1] (else, point p is outside robot workspace!)

2 solutions

Inverse kinematics of 2R arm (cont'd)



by geometric inspection

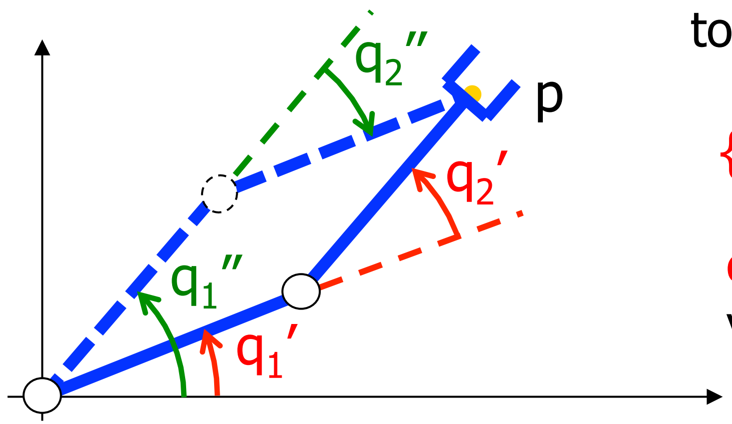
$$q_1 = \alpha - \beta$$

2 solutions
(one for each value of s_2)

$$\boxed{q_1 = \text{ATAN2}\,\{p_y, p_x\} - \text{ATAN2}\,\{l_2\, s_2\,,\, l_1 + l_2\, c_2\}}$$

note: difference of ATAN2 needs to be re-expressed in $(-\pi\,,\,\pi]$!

$\{q_1,q_2\}_{UP/LEFT}$



$\{q_1,q_2\}_{DOWN/RIGHT}$

q_2' e q_2'' have same absolute value, but opposite signs

Algebraic solution for q_1

another solution method…

$$\left.\begin{aligned} p_x &= l_1 c_1 + l_2 c_{12} = l_1 c_1 + l_2 (c_1 c_2 - s_1 s_2) \\ p_y &= l_1 s_1 + l_2 s_{12} = l_1 s_1 + l_2 (s_1 c_2 + c_1 s_2) \end{aligned}\right\}$$

linear in s_1 and c_1

$$\underbrace{\begin{bmatrix} l_1 + l_2 c_2 & -l_2 s_2 \\ l_2 s_2 & l_1 + l_2 c_2 \end{bmatrix}}_{\det = (l_1^2 + l_2^2 + 2\, l_1 l_2 c_2) > 0} \begin{bmatrix} c_1 \\ s_1 \end{bmatrix} = \begin{bmatrix} p_x \\ p_y \end{bmatrix}$$

except for $l_1 = l_2$ and $c_2 = -1$ being then q_1 undefined (singular case: ∞^1 solutions)

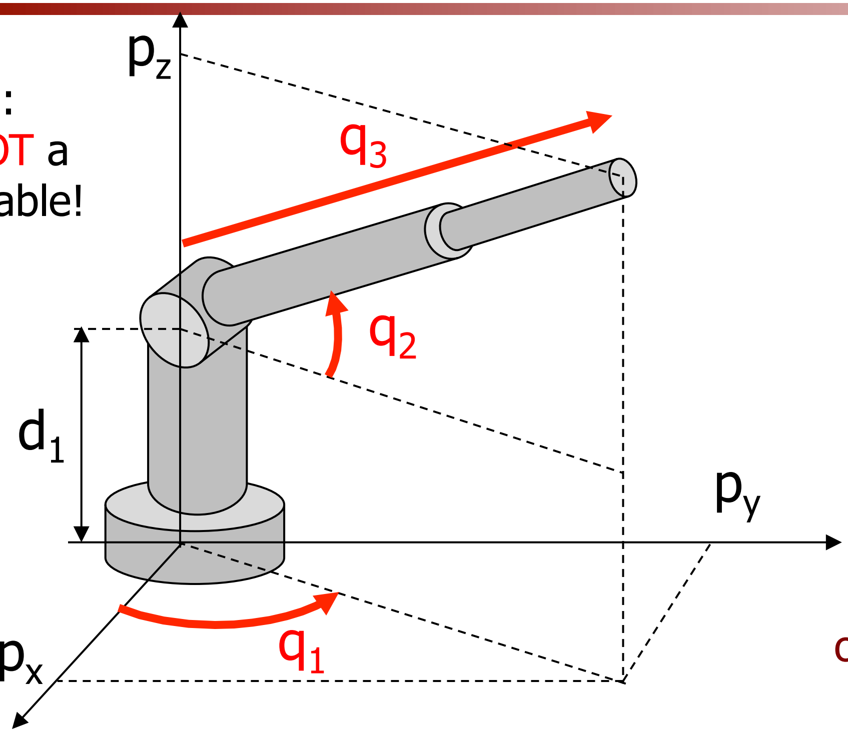
$$q_1 = \text{ATAN2}\,\{s_1, c_1\} = \text{ATAN2}\,\{(p_y(l_1+l_2c_2)-p_x l_2 s_2)/\det,\ (p_x(l_1+l_2c_2)+p_y l_2 s_2)/\det\}$$

notes: a) this method provides directly the result in $(-\pi, \pi]$

b) when evaluating ATAN2, $\det > 0$ can be eliminated from the expressions of s_1 and c_1

Inverse kinematics of polar (RRP) arm

Note:
q_2 is NOT a
DH variable!



$p_x = q_3\, c_2\, c_1$

$p_y = q_3\, c_2\, s_1$

$p_z = d_1 + q_3\, s_2$

$$p_x^2 + p_y^2 + (p_z - d_1)^2 = q_3^2$$

$$q_3 = +\sqrt{p_x^2 + p_y^2 + (p_z - d_1)^2}$$

our choice: take here only the positive value...

if $q_3 = 0$, then q_1 and q_2 remain both undefined (stop); else

$$q_2 = \text{ATAN2}\{(p_z - d_1)/q_3,\ \pm\sqrt{(p_x^2 + p_y^2)/q_3^2}\}$$

(if it stops, a singular case: ∞^2 or ∞^1 solutions)

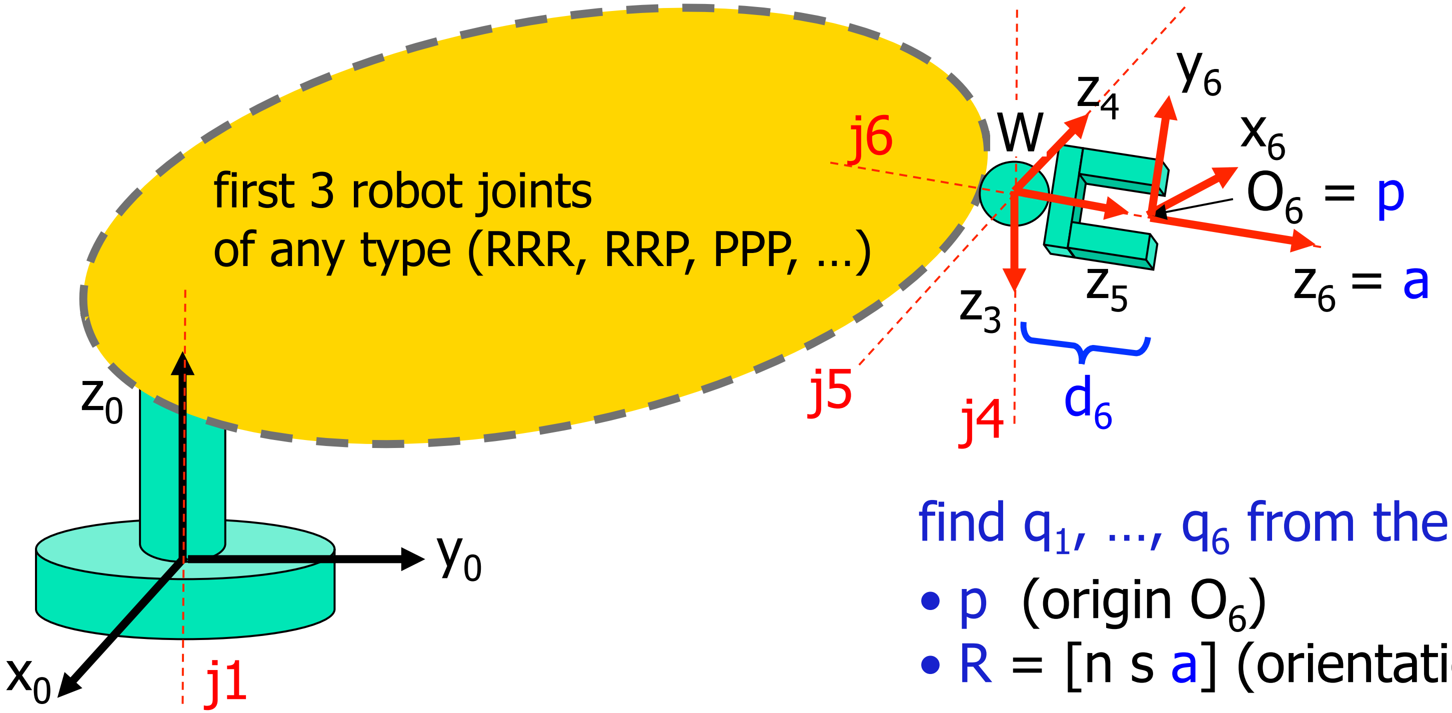
if $p_x^2+p_y^2 = 0$, then q_1 remains undefined (stop); else

$$q_1 = \text{ATAN2}\{p_y/c_2,\ p_x/c_2\}$$

(2 regular solutions $\{q_1,q_2,q_3\}$)

we have eliminated $q_3>0$ from both arguments!

Inverse kinematics for robots with spherical wrist



find $q_1, \ldots, q_6$ from the input data:

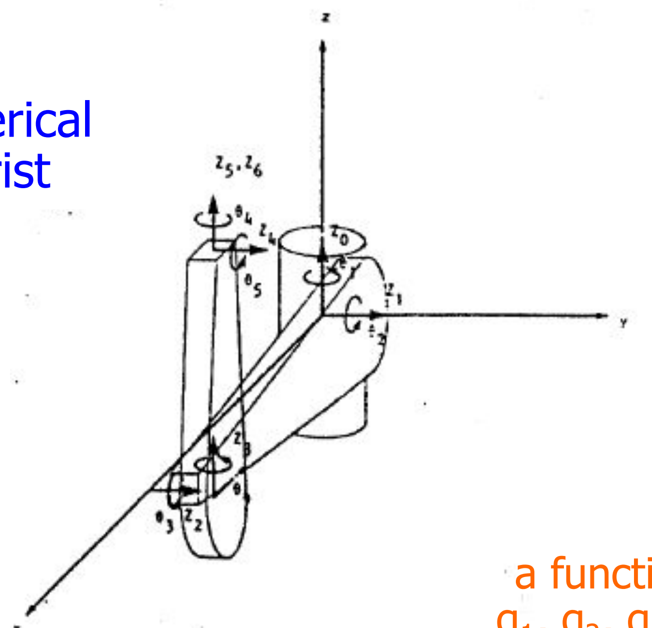
- p (origin O_6)
- R = [n s a] (orientation of RF_6)

1. $W = p - d_6\, a \;\rightarrow\; q_1, q_2, q_3$ (inverse "position" kinematics for main axes)
2. $\underset{\text{given}}{R} = \underset{\text{known, after 1.}}{{}^0R_3(q_1, q_2, q_3)}\ \underbrace{{}^3R_6(q_4, q_5, q_6)}_{\text{Euler ZYZ or ZXZ rotation matrix}} \;\rightarrow\; {}^3R_6(q_4, q_5, q_6) = {}^0R_3^T\, R \;\rightarrow\; q_4, q_5, q_6$

(inverse "orientation" kinematics for wrist)

6R example: Unimation PUMA 600

spherical
wrist



$$\left.\begin{aligned}
n_x &= C_1[C_{23}(C_4C_5C_6 - S_4S_6) - S_{23}S_5C_6] \\
&\quad - S_1[S_4C_5C_6 + C_4S_6] \\
n_y &= S_1[C_{23}(C_4C_5C_6 - S_4S_6) - S_{23}S_5C_6] \\
&\quad + C_1[S_4C_5C_6 + C_4S_6] \\
n_z &= -S_{23}(C_4C_5C_6 - S_4S_6) - C_{23}S_5C_6
\end{aligned}\right\} \; n = {}^0x_6(q)$$

$$\left.\begin{aligned}
o_x &= C_1[-C_{23}(C_4C_5S_6 + S_4C_6) + S_{23}S_5S_6] \\
&\quad - S_1[-S_4C_5S_6 + C_4C_6] \\
o_y &= S_1[-C_{23}(C_4C_5S_6 + S_4C_6) + S_{23}S_5S_6] \\
&\quad + C_1[-S_4C_5S_6 + C_4C_6] \\
o_z &= S_{23}(C_4C_5S_6 + S_4C_6) + C_{23}S_5S_6
\end{aligned}\right\} \; s = {}^0y_6(q)$$

$$\left.\begin{aligned}
a_x &= C_1(C_{23}C_4S_5 + S_{23}C_5) - S_1S_4S_5 \\
a_y &= S_1(C_{23}C_4S_5 + S_{23}C_5) + C_1S_4S_5 \\
a_z &= -S_{23}C_4S_5 + C_{23}C_5
\end{aligned}\right\} \; a = {}^0z_6(q)$$

a function of q_1, q_2, q_3 only!

$$\left.\boxed{\begin{aligned}
p_x &= C_1(d_4S_{23} + a_3C_{23} + a_2C_2) - S_1d_3 \\
p_y &= S_1(d_4S_{23} + a_3C_{23} + a_2C_2) + C_1d_3 \\
p_z &= -(-d_4C_{23} + a_3S_{23} + a_2S_2).
\end{aligned}}\right\} \; p = 0_6(q)$$

TABLE I
LINK PARAMETERS FOR PUMA ARM

Joint	α°	θ°	d	a	Range
1	-90°	θ_1	0	0	θ_1: $+/-160^\circ$
2	0	θ_2	0	a_2	θ_2: $+45 \to -225^\circ$
3	90°	θ_3	d_3	a_3	θ_3: $225^\circ \to -45^\circ$
4	-90°	θ_4	d_4	0	θ_4: $+/-170^\circ$
5	90°	θ_5	0	0	θ_5: $+/-135^\circ$
6	0	θ_6	0	0	θ_6: $+/-170^\circ$

$a_2 = 17.000$ $\quad a_3 = 0.75$
$d_3 = 4.937$ $\quad d_4 = 17.000$

here $d_6=0$,
so that $0_6=W$ directly

8 different inverse solutions
that can be found in closed form
(see Paul, Shimano, Mayer; 1981)

Numerical solution of inverse kinematics problems

- use when a closed-form solution q to $r_d = f_r(q)$ does not exist or is "too hard" to be found
- $J_r(q) = \dfrac{\partial f_r}{\partial q}$ (analytical Jacobian)
- Newton method (here for m=n)
  - $r_d = f_r(q) = f_r(q^k) + J_r(q^k)\,(q - q^k) + o(\|q - q^k\|^2)$ $\leftarrow$ neglected

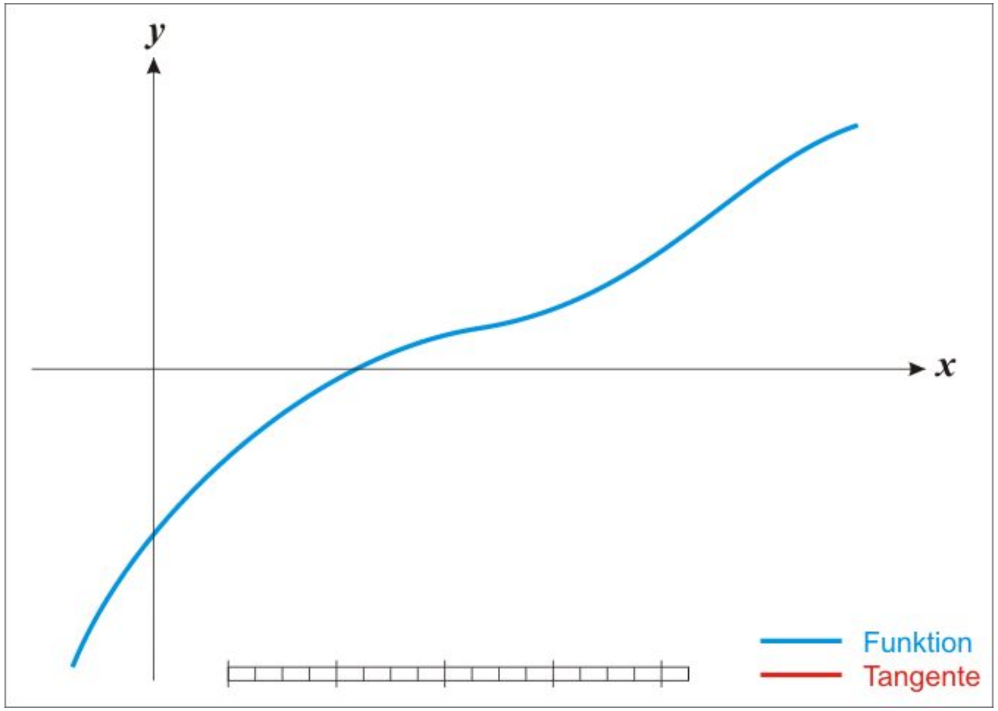
$$q^{k+1} = q^k + J_r^{-1}(q^k)\,[r_d - f_r(q^k)]$$

  - convergence if q^0 (initial guess) is close enough to some q^*: $f_r(q^*) = r_d$
  - problems near singularities of the Jacobian matrix $J_r(q)$
  - in case of robot redundancy (m<n), use the pseudo-inverse $J_r^{\#}(q)$
  - has quadratic convergence rate when near to solution (fast!)

Operation of Newton method

- in the scalar case, also known as "method of the tangent"
- for a differentiable function f(x), find a root of $f(x^*)=0$ by iterating as

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$



an approximating sequence

$$\{x_1, x_2, x_3, x_4, x_5, \dots\} \rightarrow x^*$$

animation from
http://en.wikipedia.org/wiki/File:NewtonIteration_Ani.gif

Numerical solution of inverse kinematics problems (cont'd)

- Gradient method (max descent)
  - minimize the error function

$$H(q) = \tfrac{1}{2} \, \| r_d - f_r(q) \|^2 = \tfrac{1}{2} \, [r_d - f_r(q)]^T \, [r_d - f_r(q)]$$

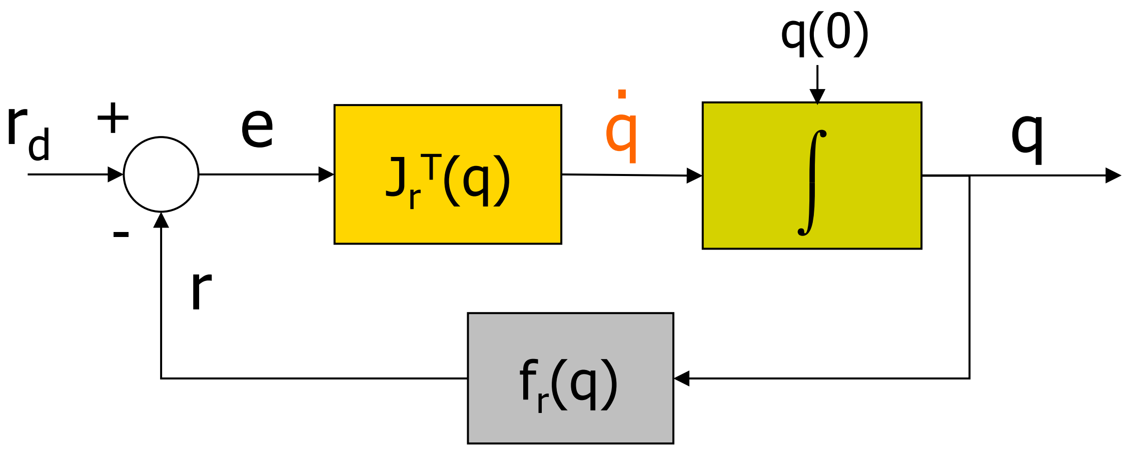
$$q^{k+1} = q^k - \alpha \, \nabla_q H(q^k)$$

from $\nabla_q H(q) = - J_r^T(q) \, [r_d - f_r(q)]$, we get

$$\boxed{q^{k+1} = q^k + \alpha \, J_r^T(q^k) \, [r_d - f_r(q^k)]}$$

  - the scalar step size $\alpha > 0$ should be chosen so as to guarantee a decrease of the error function at each iteration (too large values for α may lead the method to "miss" the minimum)
  - when the step size α is too small, convergence is extremely slow

Revisited as a "feedback" scheme



r_d = cost

(α = 1)

$e = r_d - f_r(q) \rightarrow 0 \;\Leftrightarrow$ closed-loop equilibrium e=0 is asymptotically stable

$V = \frac{1}{2}\, e^T e \geq 0$ Lyapunov candidate function

$$\dot{V} = e^T \dot{e} = e^T \frac{d}{dt}\left(r_d - f_r(q)\right) = -e^T J_r\, \dot{q} = -e^T J_r J_r^T e \leq 0$$

$\dot{V} = 0 \quad \Leftrightarrow \quad e \in Ker(J_r^T)$ in particular $e = 0$

asymptotic stability

Properties of Gradient method

- computationally simpler: Jacobian transpose, rather than its (pseudo)-inverse
- direct use also for robots that are redundant for the task
- may not converge to a solution, but it never diverges
- the discrete-time evolution of the continuous scheme

$$q^{k+1} = q^k + \Delta T \, J_r^T(q^k) \, [r_d - f(q^k)] \qquad (\alpha = \Delta T)$$

  is equivalent to an iteration of the Gradient method
- scheme can be accelerated by using a gain matrix K>0

$$\dot{q} = J_r^T(q) \, K \, e$$

note: K can be used also to "escape" from being stuck in a stationary point, by rotating the error e out of the kernel of J_r^T (if a singularity is encountered)

A case study

analytic expressions of Newton and gradient iterations

- 2R robot with $l_1 = l_2 = 1$, desired end-effector position $r_d = p_d = (1,1)$
- direct kinematic function and error

$$f_r(q) = \begin{pmatrix} c_1 + c_{12} \\ s_1 + s_{12} \end{pmatrix} \qquad e = p_d - f_r(q) = \begin{pmatrix} 1 \\ 1 \end{pmatrix} - f_r(q)$$

- Jacobian matrix

$$J_r(q) = \frac{\partial f_r(q)}{\partial q} = \begin{pmatrix} -(s_1 + s_{12}) & -s_{12} \\ c_1 + c_{12} & c_{12} \end{pmatrix}$$

- Newton versus Gradient iteration

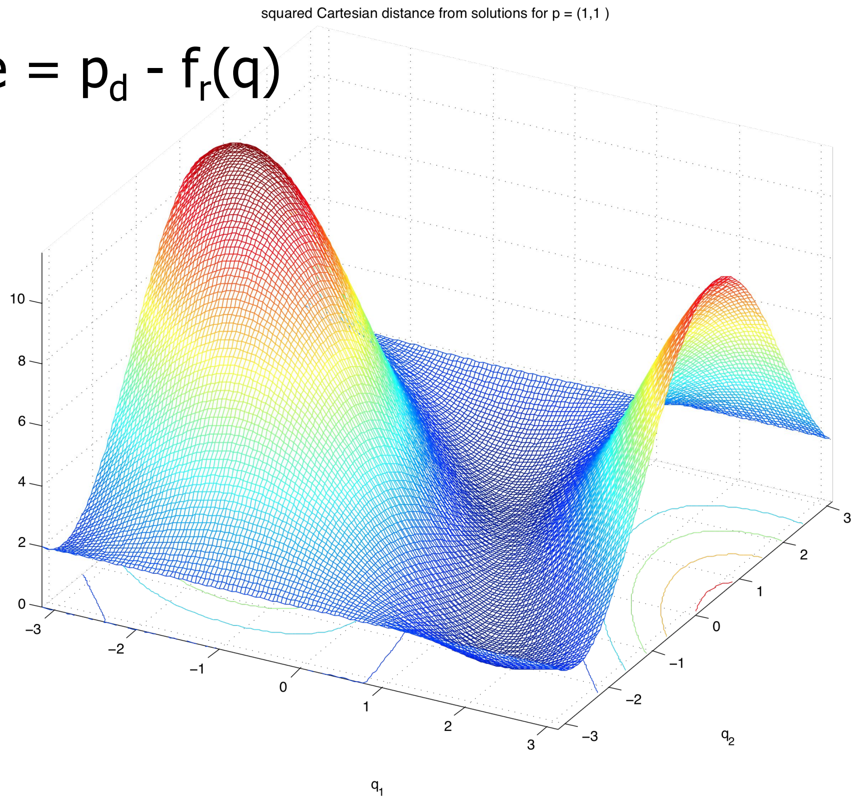
det $J_r(q)$

$$q^{k+1} = q^k + \left\{ \begin{array}{l} \overbrace{\left. \frac{1}{s_2} \begin{pmatrix} c_{12} & s_{12} \\ -(c_1 + c_{12}) & -(s_1 + s_{12}) \end{pmatrix} \right|_{q=q^k}}^{J_r^{-1}(q^k)} \\ \\ \underbrace{\left. \alpha \begin{pmatrix} -(s_1 + s_{12}) & c_1 + c_{12} \\ -s_{12} & c_{12} \end{pmatrix} \right|_{q=q^k}}_{J_r^{T}(q^k)} \end{array} \right\} \bullet \overbrace{\left. \begin{pmatrix} 1 - (c_1 + c_{12}) \\ 1 - (s_1 + s_{12}) \end{pmatrix} \right|_{q=q^k}}^{e_k}$$

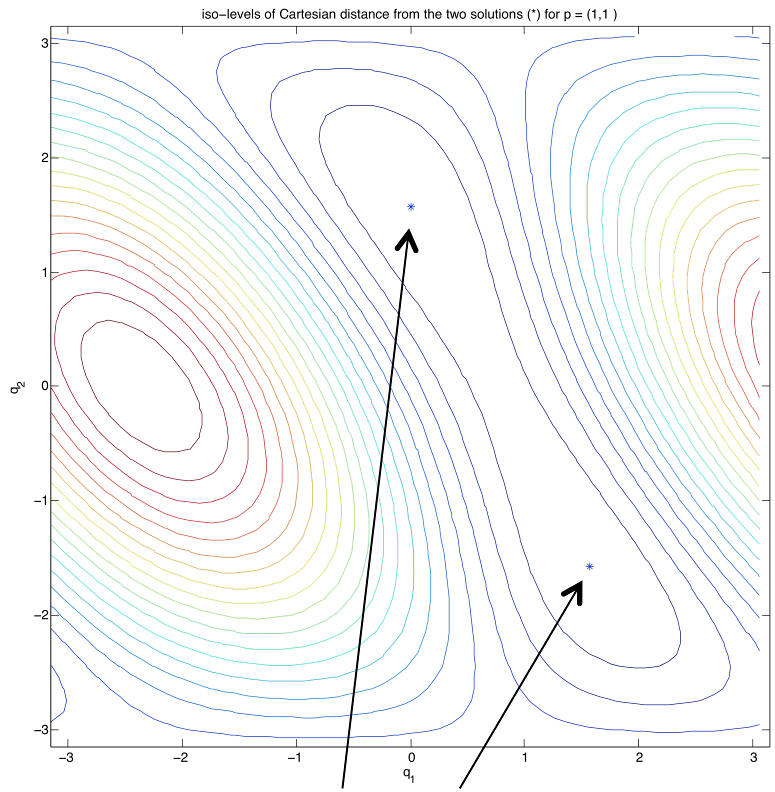
Error function

- 2R robot with $l_1=l_2=1$, desired end-effector position $p_d = (1,1)$

$e = p_d - f_r(q)$



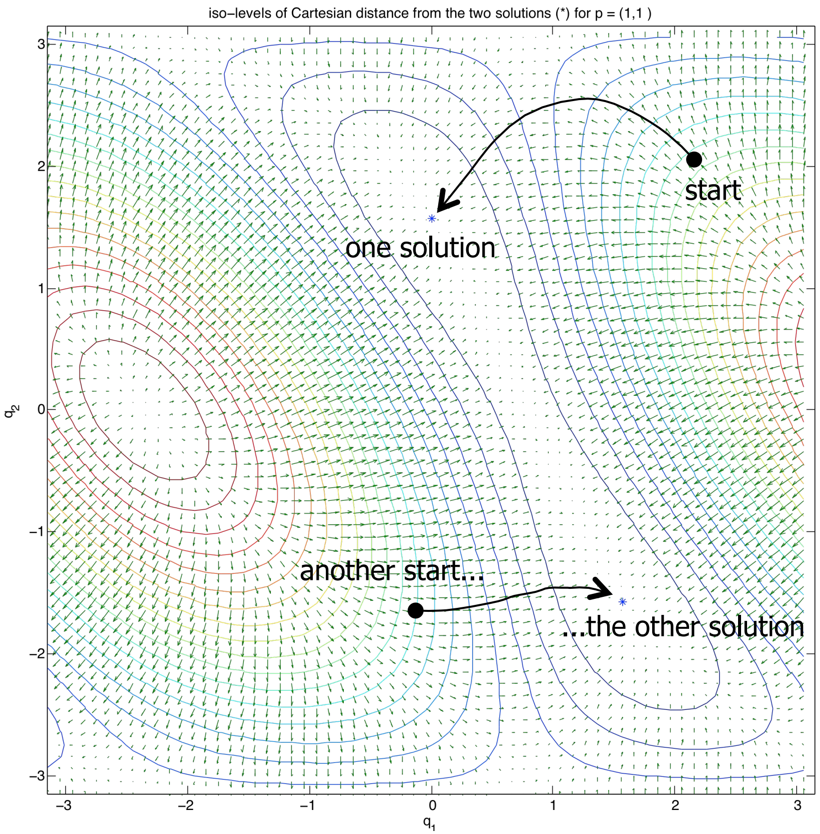
plot of $\|e\|^2$ as a function of $q = (q_1,q_2)$

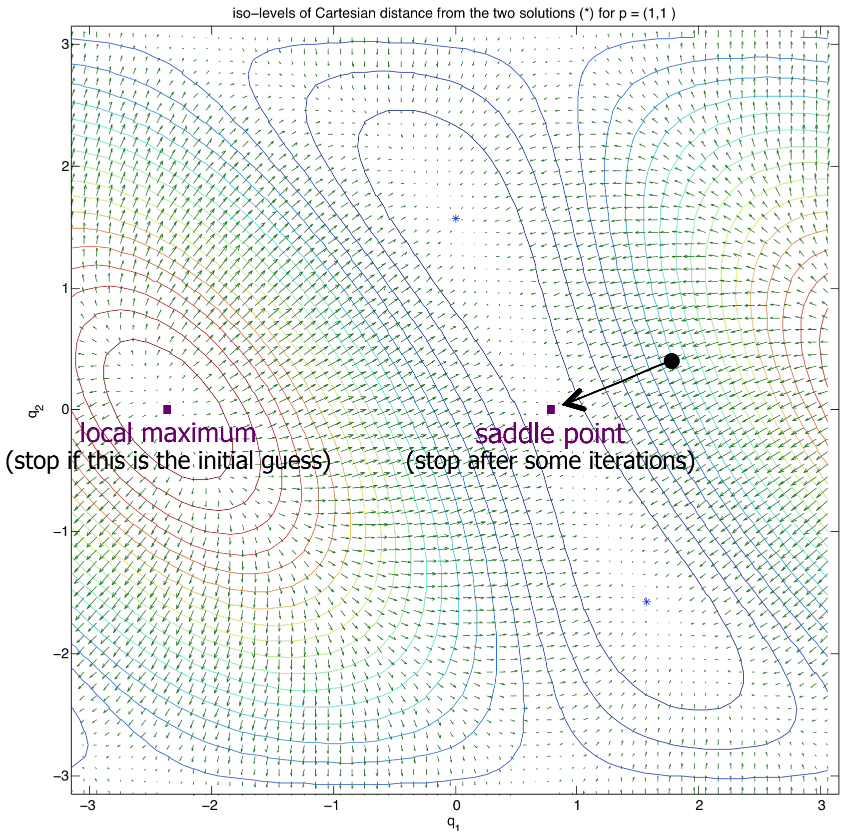


two local minima
(inverse kinematic solutions)

Error reduction by Gradient method

- flow of iterations along the negative (or anti-) gradient
- two possible cases: convergence or stuck (at zero gradient)





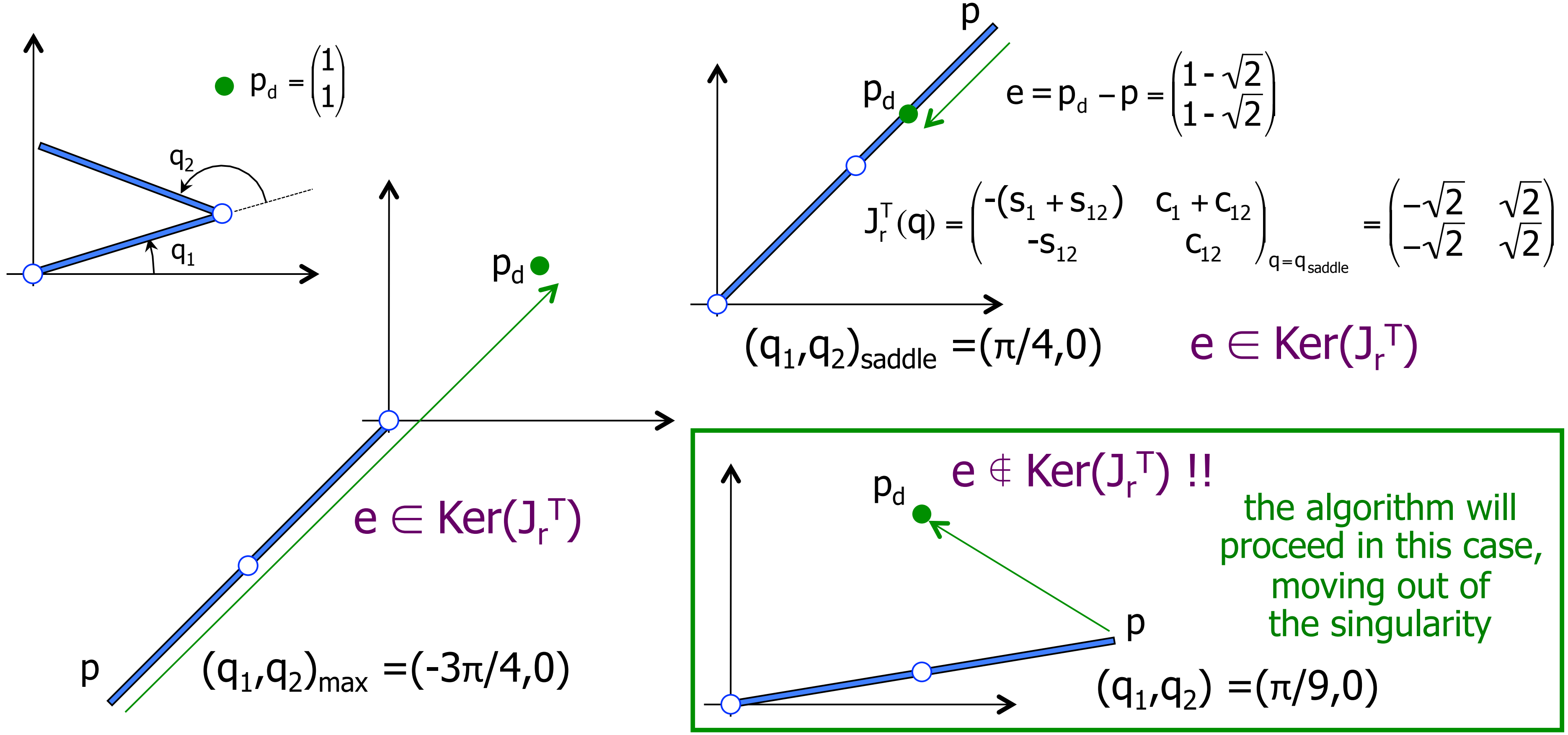
$(q_1,q_2)' = (0,\pi/2)$ $(q_1,q_2)'' = (\pi/2,-\pi/2)$

$\underbrace{(q_1,q_2)_{max} = (-3\pi/4,0) \quad (q_1,q_2)_{saddle} = (\pi/4,0)}_{e \in Ker(J_r^T)\ !}$

Convergence analysis

when does the gradient method get stuck?

- lack of convergence occurs when
  - the Jacobian matrix $J_r(q)$ is singular (the robot is in a "singular configuration")
  - AND the error is in the "null space" of $J_r^T(q)$



Issues in implementation

- initial guess q^0
  - only one inverse solution is generated for each guess
  - multiple initializations for obtaining other solutions
- optimal step size α in Gradient method
  - a constant step may work good initially, but not close to the solution (or vice versa)
  - an adaptive one-dimensional line search (e.g., Armijo's rule) could be used to choose the best α at each iteration
- stopping criteria

Cartesian error (possibly, separate for position and orientation) $\|r_d - f(q^k)\| \leq \varepsilon$

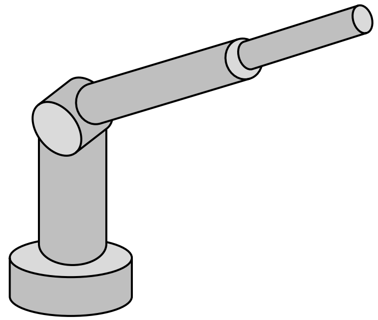
algorithm increment $\|q^{k+1}-q^k\| \leq \varepsilon_q$

- understanding closeness to singularities

$$\sigma_{min}\{J(q^k)\} \geq \sigma_0$$

numerical conditioning of Jacobian matrix (SVD) (or a simpler test on its determinant, for m=n)

Numerical tests on RRP robot



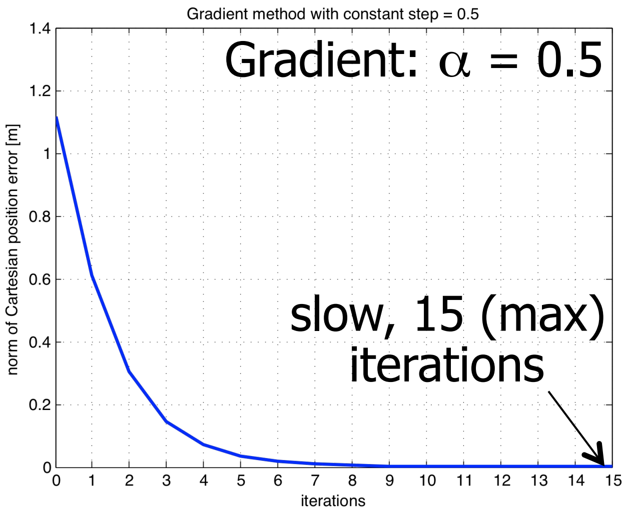
- RRP/polar robot: desired E-E position $r_d = p_d = (1, 1, 1)$ —see slide 19, $d_1=0.5$
- the two (known) analytic solutions, with $q_3 \geq 0$, are:

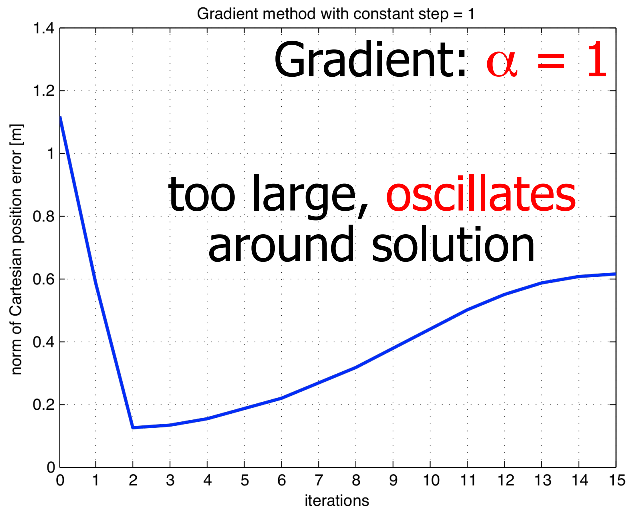
$$q^* = (0.7854,\ 0.3398,\ 1.5)$$
$$q^{**} = (q_1^* - \pi,\ \pi - q_2^*,\ q_3^*) = (-2.3562,\ 2.8018,\ 1.5)$$

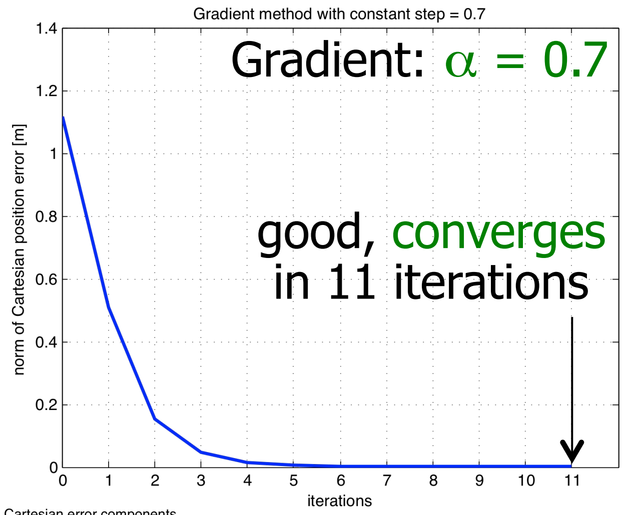
- norms $\varepsilon = 10^{-5}$ (max Cartesian error), $\varepsilon_q = 10^{-6}$ (min joint increment)
- $k_{max}=15$ (max iterations), $|\det(J_r)| \leq 10^{-4}$ (closeness to singularity)
- numerical performance of Gradient (with different α) vs. Newton
  - test 1: $q^0 = (0, 0, 1)$ as initial guess
  - test 2: $q^0 = (-\pi/4, \pi/2, 1)$ —"singular" start, since $c_2=0$ (see slide 19)
  - test 3: $q^0 = (0, \pi/2, 0)$ —"double singular" start, since also $q_3=0$
- solution and plots with Matlab code

Numerical test - 1

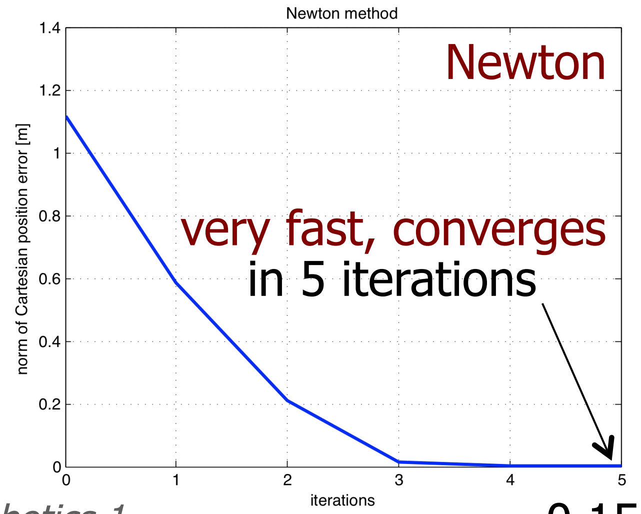
- test 1: $q^0 = (0, 0, 1)$ as initial guess; evolution of error norm





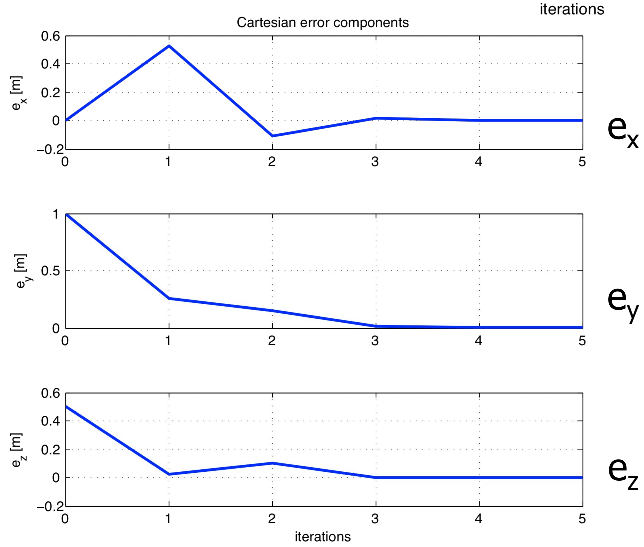


$0.57 \cdot 10^{-5}$



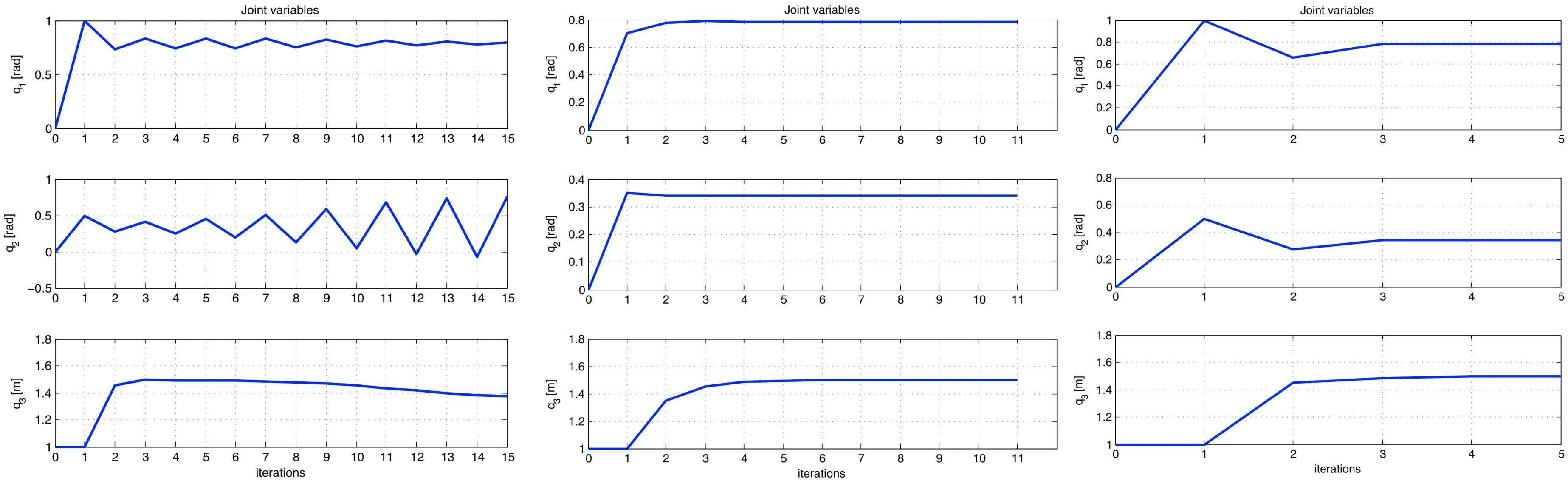
$0.15 \cdot 10^{-8}$

Cartesian errors component-wise



Numerical test - 1

- test 1: $q^0 = (0, 0, 1)$ as initial guess; evolution of joint variables



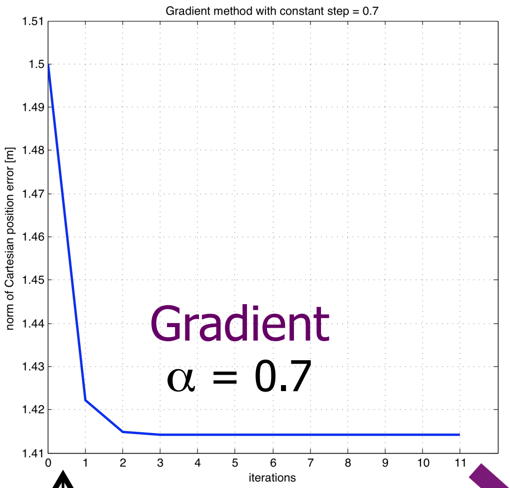
Gradient: $\alpha = 1$	Gradient: $\alpha = 0.7$	Newton
not converging to a solution	converges in 11 iterations	converges in 5 iterations

both to solution $q^* = (0.7854, 0.3398, 1.5)$

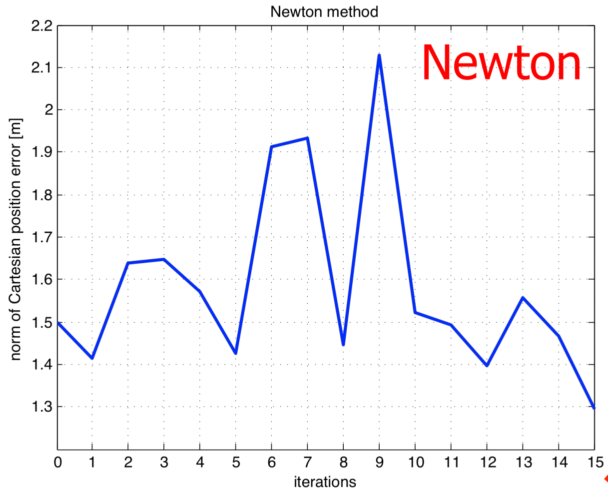
Numerical test - 2

- test 2: $q^0 = (-\pi/4, \pi/2, 1)$: singular start

error norms



Gradient
$\alpha = 0.7$

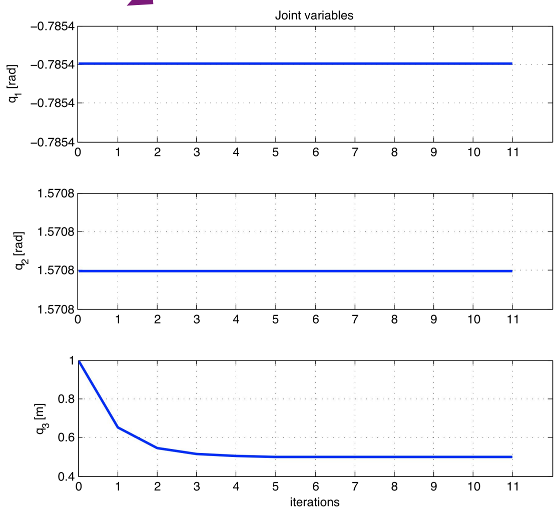


Newton ↔ with check of singularity: blocked at start

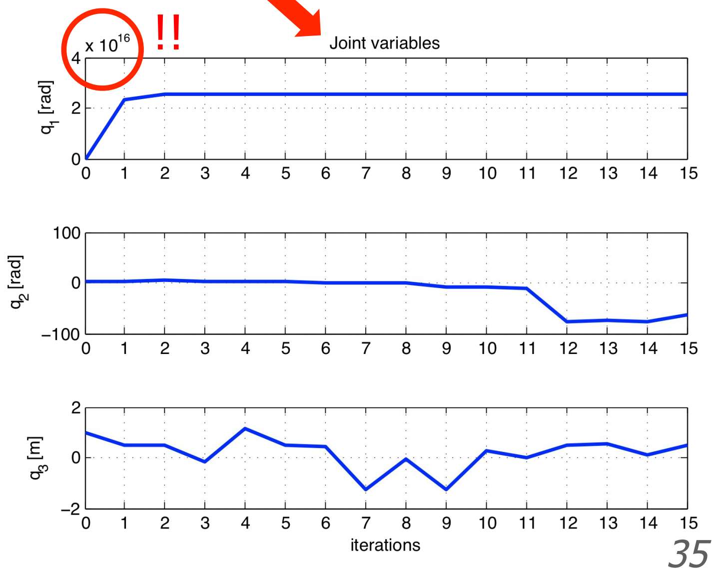
↔ without check: it diverges!

starts toward solution, but slowly stops (in singularity): when Cartesian error vector $e \in \mathrm{Ker}(J_r^T)$

joint variables



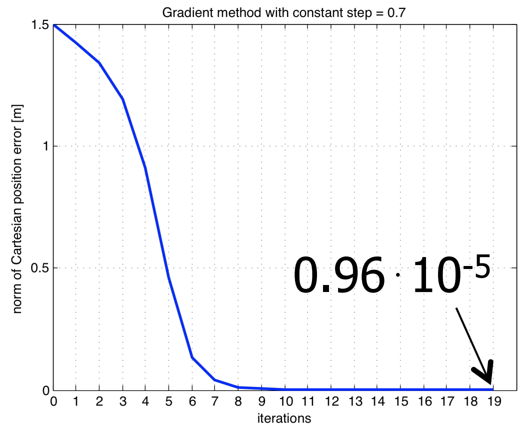
!!



Numerical test - 3

- test 3: $q^0 = (0, \pi/2, 0)$: "double" singular start

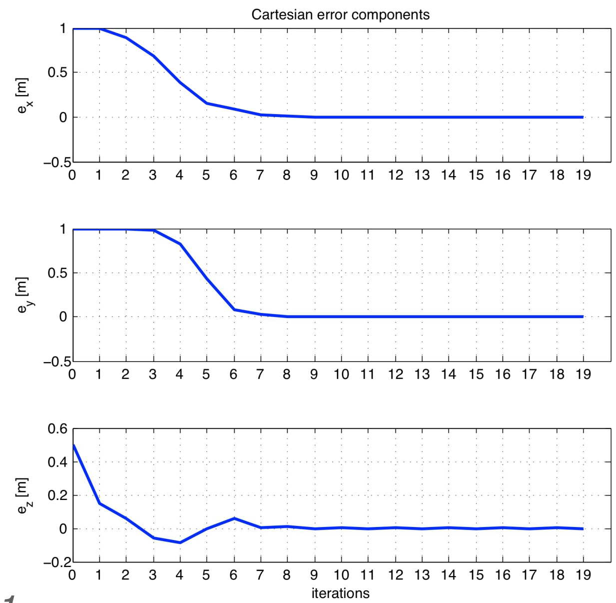
error norm



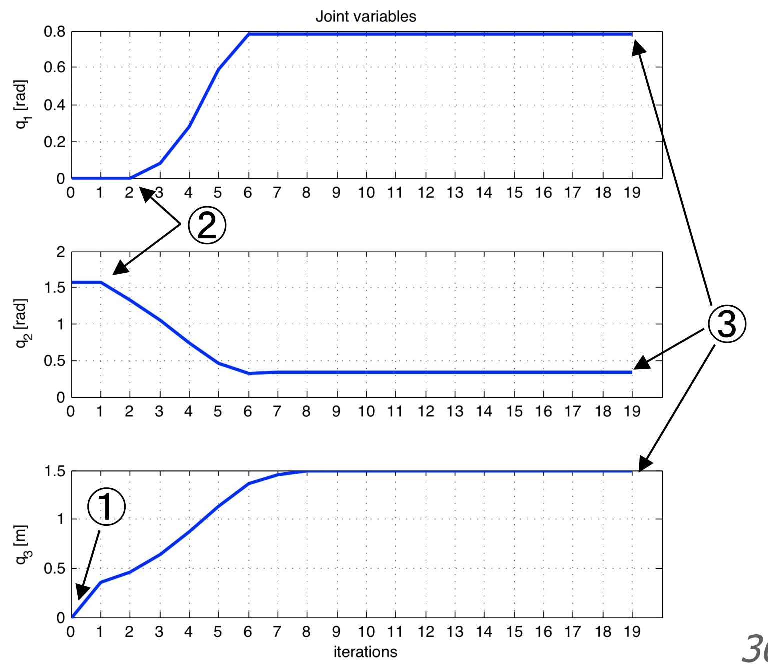
Gradient (with α = 0.7)
① starts toward solution
② exits the double singularity
③ slowly converges in 19 iterations to the solution $q^*=(0.7854, 0.3398, 1.5)$

Newton
is either
blocked at start
or (w/o check)
explodes!
→ "NaN" in Matlab

Cartesian errors



joint variables



Final remarks

- an efficient iterative scheme can be devised by combining
  - initial iterations with Gradient method ("sure but slow", having linear convergence rate)
  - switch then to Newton method (quadratic terminal convergence rate)
- joint range limits are considered only at the end
  - check if the solution found is feasible, as for analytical methods
- if the problem has to be solved on-line
  - execute iterations and associate an actual robot motion: repeat steps at times t_0, $t_1=t_0+T$, ..., $t_k=t_{k-1}+T$ (e.g., every T=40 ms)
  - the "good" choice for the initial q^0 at t_k is the solution of the previous problem at t_{k-1} (gives continuity, needs only 1-2 Newton iterations)
  - crossing of singularities and handling of joint range limits need special care in this case
- Jacobian-based inversion schemes are used also for kinematic control, along a continuous task trajectory $r_d(t)$