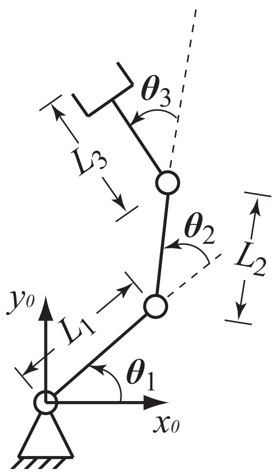
Technische Universität München **Chair of Automatic Control Engineering** Dr.-Ing. Marion Leibold **Chair of Information-Oriented Control** Dr.-Ing. Stefan Sosnowski	**Robot Control Laboratory** 2. Assignment

Given is a planar robot with $N = 3$ segments from the first Matlab assignment (shown below). The robot consists of three revolute joints θ_1, θ_2 and θ_3. The length of the segments is defined as $L_1 = 4\,\text{m}$, $L_2 = 3\,\text{m}$ and $L_3 = 2\,\text{m}$.



Exercise 1: Jacobian Matrix

a) Compute the Jacobian matrix of the forward kinematics

$$
\begin{aligned}
r_x &= L_1 \cdot \cos\theta_1 + L_2 \cdot \cos(\theta_1 + \theta_2) + L_3 \cdot \cos(\theta_1 + \theta_2 + \theta_3) \\
r_y &= L_1 \cdot \sin\theta_1 + L_2 \cdot \sin(\theta_1 + \theta_2) + L_3 \cdot \sin(\theta_1 + \theta_2 + \theta_3) \\
\phi &= \theta_1 + \theta_2 + \theta_3
\end{aligned}
$$

and discuss the existence of singularities. Additionally sketch the configurations of the singularities (as in figure V.9 in the lecture material).

b) Program the MATLAB function **cal_J** that computes the Jacobian matrix.

(8)	$\boldsymbol{J} = \text{cal\_J}(\theta_1, \theta_2, \theta_3)$	MAT
Input:	$\theta_1, \theta_2, \theta_3$ scalar angles in radians	
Output:	$\boldsymbol{J}$ 3×3 Jacobian matrix	
Allowed routines:	—	

Test: Test your results from a) and b) with the method $\mathbf{jacob0}()$ from the *Robotics Toolbox for Matlab*.

c) Simulate the movement of the robot using the **Resolved Rate Control** (compare to chapter V.5.2 of the lecture notes). Therefore the function **rrc** should be programmed that computes from the initial parameters

- initial joint angles $\boldsymbol{q}(t_0) = \boldsymbol{q}_0 = (\theta_{10}, \theta_{20}, \theta_{30})^T$ [rad, rad, rad]
- constant cartesian velocity $\dot{\boldsymbol{w}}(t_0) = \dot{\boldsymbol{w}}_0 = \dot{\boldsymbol{w}}_i = (\dot{r}_x, \dot{r}_y, \dot{\phi})^T$ [m/s, m/s, rad/s]
- constant external force $\boldsymbol{F}(t_0) = \boldsymbol{F}_0 = \boldsymbol{F}_i = (f_x, f_y, m_z)^T$ [N, N, Nm]
- simulation time T [s]

the output parameters

- joint angles $\boldsymbol{q}(t_i) = \boldsymbol{q}_i = (\theta_{1i}, \theta_{2i}, \theta_{3i})^T$ [rad, rad, rad]
- angular velocity of the joints $\dot{\boldsymbol{q}}(t_i) = \dot{\boldsymbol{q}}_{\boldsymbol{i}} = (\dot{\theta}_{1i}, \dot{\theta}_{2i}, \dot{\theta}_{3i})^T$ [rad/s, rad/s, rad/s]
- cartesian position of the end effector $\boldsymbol{w}(t_i) = \boldsymbol{w}_i = (r_{xi}, r_{yi}, \phi_i)^T$ [m, m, rad]
- determinant of the Jacobian matrix $|\boldsymbol{J}(\boldsymbol{q}_i)|$
- joint torques $\boldsymbol{\tau}(t_i) = \boldsymbol{\tau}_i = (\tau_{1i}, \tau_{2i}, \tau_{3i})^T$ [Nm, Nm, Nm] exerted by the external force $\boldsymbol{F}_0$

for all time steps t_i with initial time $t_0 = 0$ s. Use a time step discretization of $\triangle t = 0.1$ s and the explicit Euler method as a numerical integration scheme, i.e.

$$\dot{\boldsymbol{q}} = \boldsymbol{f}(t, \boldsymbol{q}) \rightarrow \boldsymbol{q}(t_{k+1}) = \boldsymbol{q}(t_k) + f(t_k, \boldsymbol{q}(t_k))\triangle t.$$

(18) $[\boldsymbol{Q}, \dot{\boldsymbol{Q}}, \boldsymbol{W}, \boldsymbol{K}_{det}, \boldsymbol{\Gamma}] = \text{rrc}(\boldsymbol{q}_0, \dot{\boldsymbol{w}}_0, \boldsymbol{F}_0, T)$ MAT

Input:	$\boldsymbol{q}_0$ 3×1 vector in radians
	$\dot{\boldsymbol{w}}_0$ 3×1 vector (third entry in radians)
	$\boldsymbol{F}_0$ 3×1 vector
	T scalar
Output:	$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{q}_0 & \boldsymbol{q}_1 & \dots & \boldsymbol{q}_n \end{bmatrix}$ 3×(n+1) matrix in radians
	$\dot{\boldsymbol{Q}} = \begin{bmatrix} \dot{\boldsymbol{q}}_0 & \dot{\boldsymbol{q}}_1 & \dots & \dot{\boldsymbol{q}}_n \end{bmatrix}$ 3×(n+1) matrix in radians/s
	$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{w}_0 & \boldsymbol{w}_1 & \dots & \boldsymbol{w}_n \end{bmatrix}$ 3×(n+1) matrix (third row in radians)
	$\boldsymbol{K} = \begin{bmatrix} \lvert\boldsymbol{J}(\boldsymbol{q}_0)\rvert & \lvert\boldsymbol{J}(\boldsymbol{q}_1)\rvert & \dots & \lvert\boldsymbol{J}(\boldsymbol{q}_n)\rvert \end{bmatrix}$ 1×(n+1) vector
	$\boldsymbol{\Gamma} = \begin{bmatrix} \boldsymbol{\tau}_0 & \boldsymbol{\tau}_1 & \dots & \boldsymbol{\tau}_n \end{bmatrix}$ 3×(n+1) matrix
Allowed routines:	Link(), SerialLink(), fkine(), jacob0()

Test your function with the following initial parameters, plot the results in five subplots and label the axes correctly.

- $\boldsymbol{q}_0 = (10°, 20°, 30°)^T$
- $\dot{\boldsymbol{w}}_0 = (0.2, -0.3, -0.2)^T$ (m/s, m/s, rad/s)
- $\boldsymbol{F}_0 = (1, 2, 3)^T$ (N, N, Nm)
- $T = 5$ sec

Use the forward kinematics to check whether the joint angles at time step $t = 5$ s correspond to the cartesian position and orientation of the end effector.

Exercise 2: Inverse Kinematics

a) Consider how the inverse kinematics to compute $\boldsymbol{q} = [\theta_1, \theta_2, \theta_3]^T$ from $\boldsymbol{w} = [r_x,\ r_y,\ \phi]^T$ could be derived.

b) Program a MATLAB function **rwl** that iteratively computes the inverse kinematics using **Newton-Raphson method** (equation V.6 in the lecture notes). Use the following conditions to terminate the iteration:

- Terminate if the difference between the newly computed angle and the old angle is less than $\triangle \boldsymbol{q} = 0.01$
- Terminate after 20 iteration steps

Inputs are the cartesian position and orientation of the end effector $\boldsymbol{w}$ and the initial guess $\boldsymbol{q}_0$ for the Newton-Raphson method. Outputs are the resulting angles $\boldsymbol{q}$.

(12) $\boldsymbol{q} = \text{rwl}(\boldsymbol{w}, \boldsymbol{q}_0)$	MAT
Input:	$\boldsymbol{w}$ 3×1 vector (third entry in radians)
	$\boldsymbol{q}_0$ 3×1 vector in radians
Output:	$\boldsymbol{q}$ 3×1 vector in radians
Allowed routines:	Link(), SerialLink(), fkine(), jacob0()

Test your function with the method $\mathbf{ikine()}$ from the *Robotics Toolbox for Matlab*, an initial guess $\boldsymbol{q}_0 = [0.1, 0.2, 0.3]^T$ and the following transformation matrices:

$$1)\quad {}^0T_E = \begin{bmatrix} 0.5 & -0.8660 & 0 & 6.3925 \\ 0.8660 & 0.5 & 0 & 6.0302 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$2)\quad {}^0T_E = \begin{bmatrix} 0.5 & -0.8660 & 0 & 7.5373 \\ 0.8660 & 0.5 & 0 & 3.9266 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Please note that you have to substitute the rotational part of the matrices with the rotational part of the forward kinematics for the angles $\Theta = (10°, 20°, 30°)^T$. The error should stay in the range of 10^{-3} degree. The orientation of the end effector can be computed with the atan2() function and the solutions for the forward kinematics of assignment 1.