Inverse Kinematics Resolved Rates Algorithm for Serial Robots

The inverse kinematics problem of serial robots is nonlinear in joint variables. Therefore, given the desired end effector pose[1] $\mathbf{x}_d \in \mathbb{R}^m$ it is difficult to solve for the desired joint values $\mathbf{q_d}$ in closed-form. There is a need for a fast online solution for this problem for control purposes. This handout explains how this problem is easily solved using the "resolved rates" algorithm.

The key idea behind the resolved rates algorithm is that, despite the nonlinearity of the inverse kinematics problem, its time derivative (known as the *instantaneous inverse kinematics problem*) is linear in joint speeds $\dot{\mathbf{q}} \in \mathbb{R}^n$ and end effector twist $\dot{\mathbf{x}} \in \mathbb{R}^m$. This problem is stated in Eq. (1) where $\mathbf{J}_{m\times n}$ is the instantaneous inverse kinematics Jacobian matrix [2].

$$\dot{\mathbf{x}} = \mathbf{J}\dot{\mathbf{q}}, \qquad \dot{\mathbf{x}} \in \mathbb{R}^m, \qquad \dot{\mathbf{q}} \in \mathbb{R}^n \tag{1}$$

For each time t the robot has a *current* pose (position and orientation $(\mathbf{p}_c, \mathbf{R}_c)$) obtained from direct kinematics as a function of current joint values $\mathbf{q}_c$. The difference between the *desired* pose $(\mathbf{p}_d, \mathbf{R}_d)$ and the current pose defines the position and the orientation error that, if added to the current pose, would result in the desired pose.

The position error is easy to calculate using Eq. (2). The orientation error is calculated using Eq. (3). It is essentially the rotation angle θ_e about an axis $\hat{\mathbf{m}}_e$. These two entities are the axis-angle parametrization of the orientation error matrix $\mathbf{R}_e$, which can be calculated using Eq. (4). This rotation matrix is the necessary rotation in a *fixed frame sequence* that would correct the current orientation and result in the desired orientation. The rotation angle θ_e can be obtained using Eq. (5) and the axis of rotation $\hat{\mathbf{m}}_e$ can be obtained using Eq. (6).

$$\delta_p = \sqrt{(\mathbf{p}_d - \mathbf{p}_c)^{\mathrm{T}}(\mathbf{p}_d - \mathbf{p}_c)} \tag{2}$$

$$\delta_\omega = \sqrt{(\theta_e\hat{\mathbf{m}}_e)^{\mathrm{T}}(\theta_e\hat{\mathbf{m}}_e)} = \theta_e \tag{3}$$

$$\mathbf{R}_d = \mathbf{R}_e\left(\theta_e, \hat{\mathbf{m}}_e\right)\mathbf{R}_c \quad \text{(note this is a fixed frame rotation sequence)} \rightarrow \mathbf{R}_e = \mathbf{R}_d\mathbf{R}_c^{\mathrm{T}} \tag{4}$$

$$\theta_e = cos^{-1}\left(\frac{\mathbf{e}_{1c}^{\mathrm{T}}\mathbf{e}_{1d} + \mathbf{e}_{2c}^{\mathrm{T}}\mathbf{e}_{2d} + \mathbf{e}_{3c}^{\mathrm{T}}\mathbf{e}_{3d} - 1}{2}\right) = cos^{-1}\left(\frac{\mathrm{Tr}(\mathbf{R}_e) - 1}{2}\right) \tag{5}$$

$$\hat{\mathbf{m}}_e = \frac{(\mathbf{e}_{1c}\times\mathbf{e}_{1d} + \mathbf{e}_{2c}\times\mathbf{e}_{2d} + \mathbf{e}_{3c}\times\mathbf{e}_{3d})}{2\sin(\theta_e)} = \frac{\left(\mathbf{R}_e - \mathbf{R}_e^{\mathrm{T}}\right)^\vee}{2\sin(\theta_e)} = \frac{1}{2\sin(\theta_e)}\begin{bmatrix}\mathbf{R}_e(3,2) - \mathbf{R}_e(2,3)\\ \mathbf{R}_e(1,3) - \mathbf{R}_e(3,1)\\ \mathbf{R}_e(2,1) - \mathbf{R}_e(1,2)\end{bmatrix} \tag{6}$$

In the above equations $\mathbf{e}_{1c}\ldots\mathbf{e}_{3c}$ and $\mathbf{e}_{1d}\ldots\mathbf{e}_{3d}$ are the unit vectors of the current and desired orientation frames, the"vee" operator $(\cdot)^\vee$ extracts the vector of a skew symmetric matrix, and $\mathrm{Tr}(\cdot)$ is the trace operator.

Using the position error δ_p and the orientation error δ_ω, it is possible to calculate the desired twist (linear and angular velocity) $\dot{\mathbf{x}}_d = [\dot{\mathbf{p}}_d^{\mathrm{T}}, \boldsymbol{\omega}_d^{\mathrm{T}}]^{\mathrm{T}}$ that, if used in Eq. (1) to solve for $\dot{\mathbf{q}}$, would result in reducing the position and the orientation error to values smaller than acceptable thresholds $\delta_p \leq \varepsilon_p$ and $\delta_\omega \leq \varepsilon_\omega$. Such a way of defining $\dot{\mathbf{x}}_d$ is given in Eq. (7) and Eq. (8) where the desired linear speed and the desired angular velocity are linearly proportional to the position and orientation error.

$$\dot{\mathbf{p}}_d = k_v(\mathbf{p}_d - \mathbf{p}_c), \quad k_v > 0 \tag{7}$$

$$\boldsymbol{\omega}_d = k_\xi\theta_e\hat{\mathbf{m}}_e, \quad k_\xi > 0 \tag{8}$$

The solution of Eq. (1) for the desired joint speeds $\dot{\mathbf{q}}_d$ given the desired generalized velocity of the end effector $\dot{\mathbf{x}}_d$ is obtained by the generalized pseudo-inverse $\mathbf{J}^+$ of the Jacobian matrix.

[1] The pose $\mathbf{x}$ parametrizes position and orientation of the end effector. Typically m=6, but for planar or spherical robots it is convenient to use m=3

[2] we assume that $\dot{\mathbf{x}}$ is the vector of linear and angular velocity, i.e. $\mathbf{J}$ is the geometric Jacobian. If $\dot{\mathbf{x}}$ uses rates of Euler angles then $\mathbf{J}$ must be the analytical Jacobian

The generalized pseudo-inverse is exactly the inverse of $\mathbf{J}$ if $m = n$ and the robot is not singular. Otherwise, close to singularity, $\mathbf{J}^+$ is calculated based on the singularity-robust inverse in Eq. (9).

$$\dot{\mathbf{q}}_d = \mathbf{J}^+ \dot{\mathbf{x}}_d, \qquad \mathbf{J}^+ = \mathbf{J}^{\mathrm{T}} \left(\mathbf{J}\mathbf{J}^{\mathrm{T}} + \rho \mathbf{I} \right)^{-1} \tag{9}$$

where ρ is a small number and $\mathbf{I}$ is identity matrix.

Now that $\dot{\mathbf{q}}_d$ is known, it is possible to calculate the new control reference value of the joints for the next iteration by assuming that this speed was applied for an increment of time Δt, Eq. (10). Eq. (10) uses the desired joint speeds $\dot{\mathbf{q}}_d$ that would incrementally reduce the error between the current and desired pose. An example of the resolved rates algorithm is shown in figure 1.

$$\mathbf{q}_{i+1} = \mathbf{q}_i + \dot{\mathbf{q}}_d \, \Delta t \tag{10}$$

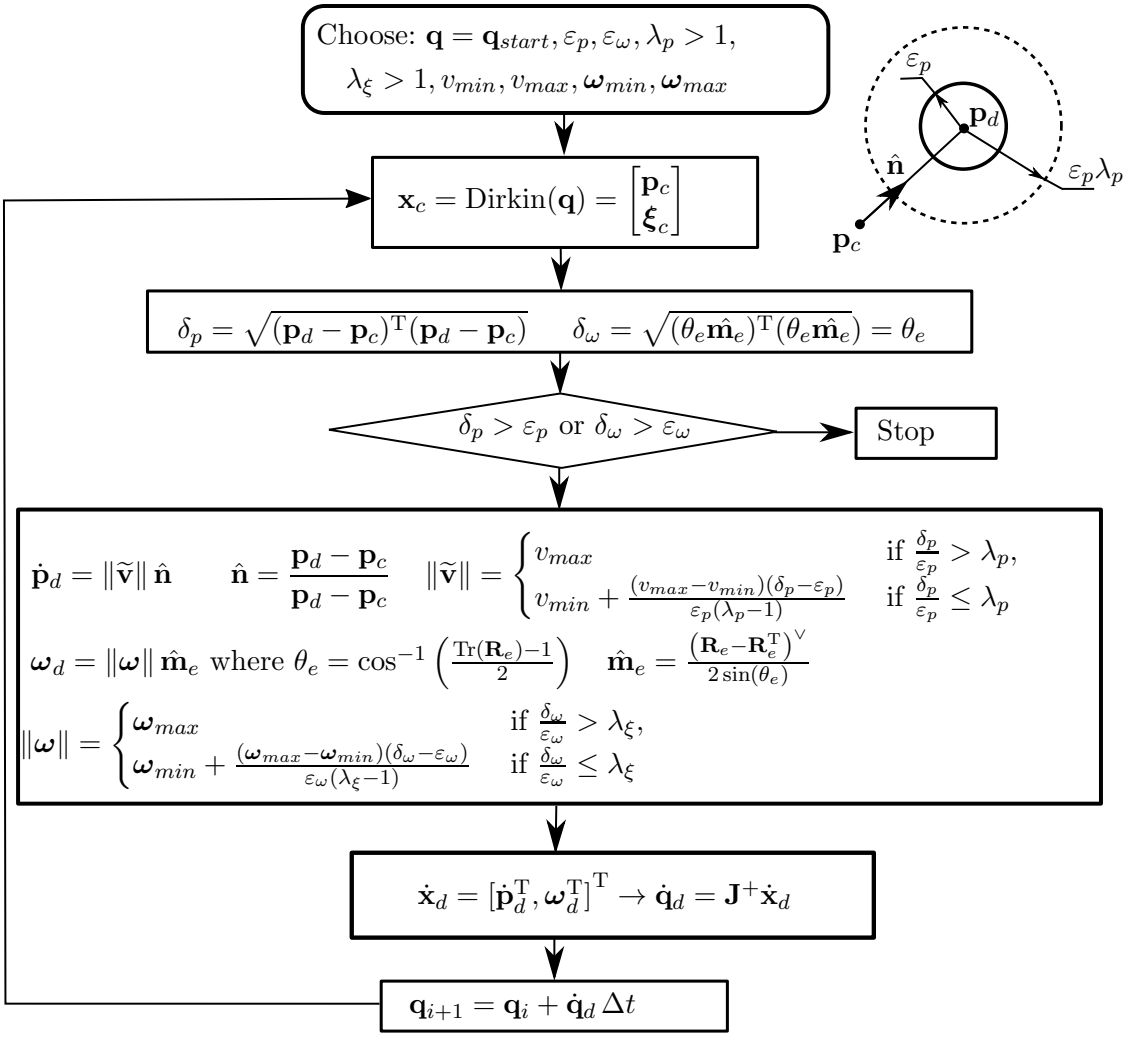

Figure 1: An implementation of resolved rates algorithm with radii of convergence for position and orientation error $\varepsilon_p, \varepsilon_\omega$ and with a linear speed decay when the errors $\delta_p < \lambda_p \varepsilon_p, \delta_\omega < \lambda_\xi \varepsilon_\omega$ where $\lambda_p, \lambda_\xi > 1$ and the operator $(\mathbf{S})^\vee$ extracts the vector of a skew symmatric matrix $\mathbf{S}$

To ensure stability of the algorithm, Δt, v_{max}, v_{min}, $\boldsymbol{\omega}_{max}$, $\boldsymbol{\omega}_{min}$ must be chosen to not violate the linear approximation due to the use of the Jacobian (i.e. the resulting $\Delta \mathbf{x}$ does not violate the small perturbation assumption). Also, to ensure convergence into the convergence radii, the following must hold:

$$\mathbf{v}_{min} \, \Delta t < \varepsilon_p, \quad \mathbf{v}_{max} \, \Delta t < \varepsilon_p \lambda_p \tag{11}$$

$$\boldsymbol{\omega}_{min} \, \Delta t < \varepsilon_\omega, \quad \boldsymbol{\omega}_{max} \, \Delta t < \varepsilon_\omega \lambda_\omega \tag{12}$$