Torque Minimization for Redundant Manipulators

Amr Aly (1848399)

January 10, 2020

Abstract

We study torque minimization as a criterion for resolving kinematic redundancy in manipulators. The minimization is studied, locally, at different time instants (current and/or future) in order to address the torque instability problem.

Introduction

Solving the inverse kinematics problem for non-redundant robots always has a unique solution (assuming no singularities). A robot is said to be redundant if the number of actuated degrees of freedom $n > m$, where the desired Cartesian task to be realized is m-dimensional.

For redundant robots, the inverse kinematics problem yields infinite solutions. To solve redundancy, joint motions are selected from infinite possibilities to achieve a secondary objective besides the desired end-effector motion. In this report, we define the secondary objective to be the minimum requirement of joint torques constrained by the defined trajectory.

Methods for resolving the kinematic redundancy are classified into two kinds – kinematic resolution or dynamic resolution – according to whether the manipulator dynamics are considered or not. Methods studied here fall under dynamic resolution since the manipulator dynamics affect the joint torques directly through the manipulator's dynamic model.

We study four methods to minimize the torque output. Some of the methods considered develop a global optimization scheme for the whole trajectory, however, this proved to be a very complex computation. Instead, local schemes were proposed that require local information of the end-effector motion.

The first method by Hollerbach et al [1] minimizes the instantaneous motor torques by resolving the redundancy at the acceleration level. The solution was unstable for longer movements. It produced motion oscillations, growing joint velocities, and sudden whipping torque effects. The second method is a variation of the first method, which takes into account the torque limits.

The third method by Li et al [2] formulates the problem as: using the torque at the current time instant $\boldsymbol{\tau}_k$ to drive the joint velocities to an optimal value $\dot{\boldsymbol{q}}_{k+1}^{opt}$ that minimizes the torque at the next time instant $\boldsymbol{\tau}_{k+1}$ subject to the task and torque bounds constraints.

The fourth method by Al Khudir et al [3] introduces an extension to the first method, that addresses the torque instability issue. Here, the joint torque norm is minimized over two successive discrete-time steps using a preview window. The dynamics are approximated in the future proximal state to keep the linear-quadratic property of the problem and to obtain a closed-form solution.

Dynamic Resolution of Redundancy

Kinematic resolution methods presume that, since the sum of squares of joint velocities is minimized by the pseudo-inverse of the task jacobian, then the kinetic energy is approximately minimized. However, to incorporate the generalized inverse into dynamics, the pseudo-inverse must be formulated in terms of joint accelerations.

We define some basic terminology for describing the manipulator. Consider a robot manipulator with generalized coordinates $\boldsymbol{q} \in \mathbb{R}^n$ with the Lagrangian dynamic model

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) \tag{1}$$

with inertia matrix $\boldsymbol{M}$, coriolis and centrifugal terms $\boldsymbol{c}$, gravity term $\boldsymbol{g}$ and commanded joint torque $\boldsymbol{\tau}$. We also use the shorthand notation

$$\boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q})$$

The desired task described by the variables $\boldsymbol{x} \in \mathbb{R}^m$, with $m < n$, is related to $\boldsymbol{q}$ by the task forward kinematics $\boldsymbol{x} = f(\boldsymbol{q})$.

The joint velocities $\dot{\boldsymbol{q}}$ is related to the end-effector velocity through

$$\dot{\boldsymbol{x}} = \boldsymbol{J}\dot{\boldsymbol{q}} \tag{2}$$

where $\boldsymbol{J} \in \mathbb{R}^{m \times n}$ is the task jacobian.

The joint acceleration is related to the end-effector acceleration by differentiating (2) to obtain

$$\ddot{\boldsymbol{x}} = \boldsymbol{J}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}\dot{\boldsymbol{q}} = \boldsymbol{J}\ddot{\boldsymbol{q}} + \boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \tag{3}$$

Instantaneous Torque Minimization

The method proposed by Hollerbach, denoted Minimum Torque Norm (MTN), minimizes the instant

squared torque norm

$$\begin{aligned} \min_{\ddot{q}_k} \quad & H_1 = \frac{1}{2}\|\boldsymbol{\tau}_k\|^2 \\ \text{s.t.} \quad & \boldsymbol{\tau}_k = \boldsymbol{M}_k\ddot{\boldsymbol{q}}_k + \boldsymbol{n}_k, \\ & \ddot{\boldsymbol{x}}_k = \boldsymbol{J}_k\ddot{\boldsymbol{q}}_k + \boldsymbol{h}_k \end{aligned} \tag{4}$$

The unique solution to (4) can be obtained, using the method of Lagrangian multipliers, as

$$\ddot{\boldsymbol{q}}^* = \boldsymbol{J}^{\#}_{M^{-2}}(\ddot{\boldsymbol{x}}_k - \boldsymbol{h}_k) - (\boldsymbol{I} - \boldsymbol{J}^{\#}_{M^{-2}}\boldsymbol{J}_k)\boldsymbol{M}_k^{-1}\boldsymbol{n}_k \tag{5}$$

where $\boldsymbol{J}^{\#}_{M^{-2}}$ is the weighted pseudo-inverse

$$\boldsymbol{J}^{\#}_{M^{-2}} = \boldsymbol{M}_k^{-2}\boldsymbol{J}_k^T(\boldsymbol{J}_k\boldsymbol{M}_k^{-2}\boldsymbol{J}_k^T)^{-1}$$

with the following short-hands: $\boldsymbol{M}_k = \boldsymbol{M}(\boldsymbol{q}_k), \boldsymbol{n}_k = \boldsymbol{n}(\boldsymbol{q}_k, \dot{\boldsymbol{q}}_k), \boldsymbol{J}_k = \boldsymbol{J}(\boldsymbol{q}_k), \boldsymbol{h}_k = \boldsymbol{h}(\boldsymbol{q}_k, \dot{\boldsymbol{q}}_k)$.

From (5) we notice that $\boldsymbol{M}_k^{-1}\boldsymbol{n}_k$ is the null space action that minimizes the torque norm. Also, the direct use of $\ddot{\boldsymbol{x}}_k = \ddot{\boldsymbol{x}}_{d,k}$ (the desired cartesian acceleration) results in an open loop solution. Therefore, a stabilizing PD feedback term is introduced to reduce the trajectory tracking error

$$\ddot{\boldsymbol{x}}_k = \ddot{\boldsymbol{x}}_{d,k} + \boldsymbol{K}_D(\dot{\boldsymbol{x}}_{d,k} - \dot{\boldsymbol{x}}_k) + \boldsymbol{K}_P(\boldsymbol{x}_{d,k} - \boldsymbol{x}_k) \tag{6}$$

with $m \times m$ gain matrices $\boldsymbol{K}_D > \boldsymbol{0}$ and $\boldsymbol{K}_P > \boldsymbol{0}$.

An extension to this method which considers the torque bounds, denoted Minimum Torque Norm with Bounds (MTNB), is also reviewed. Assume that the upper and lower joint torque limits are $\boldsymbol{\tau}^+$ and $\boldsymbol{\tau}^-$, respectively, and that the limits are motion-independent. The goal is to place the joint torques closest to the midpoints of the limits $\frac{1}{2}(\boldsymbol{\tau}^+ + \boldsymbol{\tau}^-)$. Thus, the minimization problem becomes

$$\begin{aligned} \min_{\ddot{q}_k} \quad & H_2 = \frac{1}{2}\left(\boldsymbol{\tau} - \frac{\boldsymbol{\tau}^+ + \boldsymbol{\tau}^-}{2}\right)^T \boldsymbol{W}\left(\boldsymbol{\tau} - \frac{\boldsymbol{\tau}^+ + \boldsymbol{\tau}^-}{2}\right) \\ \text{s.t.} \quad & \ddot{\boldsymbol{x}}_k = \boldsymbol{J}_k\ddot{\boldsymbol{q}}_k + \boldsymbol{h}_k \end{aligned} \tag{7}$$

where $\boldsymbol{W}$ is a symmetric positive-definite weighting matrix

$$\boldsymbol{W} = diag\left[\frac{1}{(\tau_i^+ + \tau_i^-)^2}\right]$$

and τ_i^+, τ_i^- are the upper and lower torque limits for joint i. H_2 is considered a function of $\boldsymbol{\tau}$, which in turn is a function of $\ddot{\boldsymbol{q}}$. Therefore, using the Lagrangian multipliers method, it is possible to obtain the optimal $\ddot{\boldsymbol{q}}^*$

$$\ddot{\boldsymbol{q}}^* = \boldsymbol{A}_k(\ddot{\boldsymbol{x}}_k - \boldsymbol{h}_k) + \boldsymbol{B}_k \tag{8}$$

where

$$\begin{aligned} \boldsymbol{A}_k &= (\boldsymbol{M}_k^T\boldsymbol{W}\boldsymbol{M}_k)^{-1}\boldsymbol{J}_K^T[\boldsymbol{J}_k(\boldsymbol{M}_k^T\boldsymbol{W}\boldsymbol{M}_k)^{-1}\boldsymbol{J}_k^T]^{-1}, \\ \boldsymbol{B}_k &= (\boldsymbol{A}_k\boldsymbol{J}_k - \boldsymbol{I})\boldsymbol{M}_k^{-1}\left(\boldsymbol{n}_k - \frac{\boldsymbol{\tau}^+ + \boldsymbol{\tau}^-}{2}\right) \end{aligned}$$

Similar to the previous method, PD feedback is applied to $\ddot{\boldsymbol{x}}_k$. It is worth to mention that both $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$ are obtained by the euler integration of $\ddot{\boldsymbol{q}}$ with $T_s = t_{k+1} - t_k$ as the timestep

$$\begin{aligned} \dot{\boldsymbol{q}}_{k+1} &= \dot{\boldsymbol{q}}(t_{k+1}) = \dot{\boldsymbol{q}}_k + \ddot{\boldsymbol{q}}_k T_s, \\ \boldsymbol{q}_{k+1} &= \boldsymbol{q}(t_{k+1}) = \boldsymbol{q}_k + \dot{\boldsymbol{q}}_k T_s + \frac{1}{2}\ddot{\boldsymbol{q}}_k T_s^2 \end{aligned} \tag{9}$$

Both MTN and MTNB have the following disadvantages:

- they minimize the joint torques locally without global consideration
- they minimize the joint torques regardless of whether they are within torque limits or not

Peak Torque Reduction

The method proposed by Li, denoted Peak Torque Reduction (PTR), follows a different approach. Instead of minimizing the joint torques at all times, PTR minimizes joint torques only when they exceed the limits. It also considers the contribution of both $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$, unlike previous methods.

The torque is not minimized upto a certain time instant t^P, at which the torque is expected to exceed the limits. In fact, the choice of $\ddot{\boldsymbol{q}}(t)$ is arbitrary as long as it satisfies (3) and

$$\dot{\boldsymbol{q}}^P_{opt} = \int_0^{t^P} \ddot{\boldsymbol{q}}(t)dt \tag{10}$$

subject to $\tau_i^- \leq \tau_i(t) \leq \tau_i^+$. The commanded torque can be rewritten by substituting (5) into (1) to obtain

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q})\left(\boldsymbol{A}(\boldsymbol{q})[\ddot{\boldsymbol{x}}(t) - \boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}})] + \boldsymbol{B}(\boldsymbol{q}, \dot{\boldsymbol{q}})\right) + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \tag{11}$$

Finally, the minimization problem is formulated as a nonlinear programming (NLP) problem by substituting (11) into (7), obtaining a fourth order objective function of $\dot{\boldsymbol{q}}$

$$\begin{aligned} \min_{\dot{q}} \quad & H_3 = \frac{1}{2}\left(\boldsymbol{\tau}^P(\boldsymbol{q}^P, \dot{\boldsymbol{q}}^P, \ddot{\boldsymbol{x}}^P) - \frac{\boldsymbol{\tau}^+ + \boldsymbol{\tau}^-}{2}\right)^T \\ & \times \boldsymbol{W}\left(\boldsymbol{\tau}^P(\boldsymbol{q}^P, \dot{\boldsymbol{q}}^P, \ddot{\boldsymbol{x}}^P) - \frac{\boldsymbol{\tau}^+ + \boldsymbol{\tau}^-}{2}\right) \\ \text{s.t.} \quad & \dot{\boldsymbol{x}}^P = \boldsymbol{J}(\boldsymbol{q}^P)\dot{\boldsymbol{q}}^P \end{aligned} \tag{12}$$

$\dot{\boldsymbol{q}}^P_{opt}$ is then obtained by minimizing H_3 and $\ddot{\boldsymbol{q}}^P_{opt}$ is available from (8).

However this approach offers a global optimization scheme and is very difficult to compute since it requires the information of the entire trajectory. It is also difficult to determine the time t^P at which the torque exceeds the limit, and the choice of $\ddot{\boldsymbol{q}}(t)$ for $t < t^P$ that satisfies (10). It is, therefore, required to develop a local optimization scheme that requires local information about the desired motion.

Consider the interval $t_k \leq t \leq t_{k+1}$, where t_k, t_{k+1} are two immediate time instants. The goal is to use the future state $k+1$ as an objective and use current torque

$\boldsymbol{\tau}_k$ to accelerate/decelerate the joint velocity from $\dot{\boldsymbol{q}}_k$ to $\dot{\boldsymbol{q}}_{k+1}^{opt}$ (thus, minimizing $\boldsymbol{\tau}_{k+1}$), while keeping $\boldsymbol{\tau}_k$ within limits.

The minimization problem in (12) can then be rewritten as

$$\begin{aligned}\min_{\dot{q}} \quad & H_3 = \frac{1}{2}\left(\boldsymbol{\tau}_{k+1}(\boldsymbol{q}_{k+1},\dot{\boldsymbol{q}}_{k+1}) - \frac{\boldsymbol{\tau}^+ + \boldsymbol{\tau}^-}{2}\right)^T \\ & \times \boldsymbol{W}\left(\boldsymbol{\tau}_{k+1}(\boldsymbol{q}_{k+1},\dot{\boldsymbol{q}}_{k+1}) - \frac{\boldsymbol{\tau}^+ + \boldsymbol{\tau}^-}{2}\right) \\ \text{s.t.} \quad & \dot{\boldsymbol{x}}_{k+1} = \boldsymbol{J}(\boldsymbol{q}_{k+1})\dot{\boldsymbol{q}}_{k+1}, \\ & \boldsymbol{\tau}_k^- \leq \boldsymbol{\tau}_k^i \leq \boldsymbol{\tau}_k^+ \end{aligned} \tag{13}$$

In other words, $\boldsymbol{\tau}_k$ is used to minimize $\boldsymbol{\tau}_{k+1}$. Hence, $\boldsymbol{\tau}_k$ is based on the preview information of the next state. When a large torque is needed at the next state due to the large $\dot{\boldsymbol{x}}_{d,k+1}$ and $\ddot{\boldsymbol{x}}_{d,k+1}$, a large current torque $\boldsymbol{\tau}_k^{opt}$ may be used (subject to the torque limits) to drive the manipulator to the optimum velocity $\dot{\boldsymbol{q}}_{k+1}^{opt}$ at the next state, which along with $\ddot{\boldsymbol{q}}_{k+1}^{opt}$, will reduce the joint torque $\boldsymbol{\tau}_{k+1}$.

In order to make the problem simpler and the dependencies more clear, we use the following substitutions. At current state k, $\boldsymbol{q}_k$ and $\dot{\boldsymbol{q}}_k$ are already known from the previous timestep. The joint position and velocity for the next state are computed using euler integration (similar to Eqn. 9)

$$\boldsymbol{q}_{k+1} \approx \boldsymbol{q}_k + \dot{\boldsymbol{q}}_k T_s \tag{14}$$

$$\dot{\boldsymbol{q}}_{k+1} \approx \dot{\boldsymbol{q}}_k + \ddot{\boldsymbol{q}}_k T_s \tag{15}$$

$\ddot{\boldsymbol{q}}$ can be obtained by rewriting (1) to be

$$\ddot{\boldsymbol{q}}_k = \boldsymbol{M}^{-1}(\boldsymbol{q}_k)\left[\boldsymbol{\tau}_k - \boldsymbol{n}(\boldsymbol{q}_k,\dot{\boldsymbol{q}}_k)\right] \tag{16}$$

equation (15) then becomes

$$\dot{\boldsymbol{q}}_{k+1} = \dot{\boldsymbol{q}}_k + \boldsymbol{M}^{-1}(\boldsymbol{q}_k)\left[\boldsymbol{\tau}_k - \boldsymbol{n}(\boldsymbol{q}_k,\dot{\boldsymbol{q}}_k)\right] T_s \tag{17}$$

It is clear that $\boldsymbol{q}_{k+1}$ can be directly computed and that the only unknown in $\dot{\boldsymbol{q}}_{k+1}$ is $\boldsymbol{\tau}_k$. Thus the only unknown variable in (13), in both the objective function and the constraint, is $\boldsymbol{\tau}_k$. So the optimization problem results in $\boldsymbol{\tau}_k^{opt}$, and from (16), (17) we obtain $\ddot{\boldsymbol{q}}_k^{opt}$, $\dot{\boldsymbol{q}}_{k+1}^{opt}$, respectively.

It is worthy to note that since PTR is based on the optimization of an NLP problem, it is, therefore, not suitable for real-time control; unlike other methods presented here which provide closed-form solutions to the task.

Stable Torque Minimization Using Short Preview

The method proposed by Al Khudir, denoted Model Based Preview (MBP), augments the idea of MTN with the idea of PTR. Here, we use two successive discrete-time samples to minimize the joint torque norm. The two samples are the current time instant and a future instant (possibly, the next sampling instant). To keep the linear-quadratic formulation of the problem, suitable approximations are applied to the dynamics of the future state.

MBP tries to address two shortcomings of the previous methods, real-time simplicity and stable torque behaviour. Computational efficiency is achieved thanks to the closed-form solution of a linear-quadratic (LQ) problem. Stability is realized by using a single preview state, which may be close to or further away from the current one.

To prevent the unstable behaviour, consider a preview window $T = pT_s$ from some integer $p \geq 1$. In the following, we consider the case when $p = 1$, but the same formulas can be applied for $p > 1$.

Similarly to previous methods, the goal is to minimize the following objective

$$\begin{aligned}\min_{\ddot{q}_k,\ddot{q}_{k+1}} \quad & H_4 = \frac{1}{2}\left(\omega_k\|\boldsymbol{\tau}_k\|^2 + \omega_{k+1}\|\boldsymbol{\tau}_{k+1}\|^2\right) \\ \text{s.t.} \quad & \boldsymbol{\tau}_k = \boldsymbol{M}_k\ddot{\boldsymbol{q}}_k + \boldsymbol{n}_k, \\ & \ddot{\boldsymbol{x}}_k = \boldsymbol{J}_k\ddot{\boldsymbol{q}}_k + \boldsymbol{h}_k, \\ & \boldsymbol{\tau}_{k+1} = \boldsymbol{M}_{k+1}\ddot{\boldsymbol{q}}_{k+1} + \boldsymbol{n}_{k+1}, \\ & \ddot{\boldsymbol{x}}_{k+1} = \boldsymbol{J}_{k+1}\ddot{\boldsymbol{q}}_{k+1} + \boldsymbol{h}_{k+1} \end{aligned} \tag{18}$$

where $\omega_k > 0$, $\omega_{k+1} > 0$ weigh the torque norms at current and future instants.

When using (9) in (18), the problem loses the LQ structure in the unknown joint accelerations. In fact, the objective function will no longer be quadratic in $\ddot{\boldsymbol{q}}_k$ and the constraints become nonlinear in $\ddot{\boldsymbol{q}}_k$. Therefore, suitable dynamic approximations are introduced to the future state in order to preserve the forward coupling between current acceleration and the command at the future state allowing a linear dependence of the constraints and a quadratic dependence of the objective on $\ddot{\boldsymbol{q}}_k$.

Consider the approximated task constraint at $t = t_{k+1}$

$$\ddot{\boldsymbol{x}}_{k+1} \approx \boldsymbol{J}_{k^+}\ddot{\boldsymbol{q}}_{k+1} + (\boldsymbol{J}_{k^+} - \boldsymbol{J}_k)\ddot{\boldsymbol{q}}_k + \boldsymbol{h}_{k^+} \tag{19}$$

with the following short-hands: $\boldsymbol{J}_{k^+} = \boldsymbol{J}(\boldsymbol{q}_k + \dot{\boldsymbol{q}}_k T), \boldsymbol{h}_{k^+} = \frac{\boldsymbol{J}_{k^+} - \boldsymbol{J}_k}{T}\dot{\boldsymbol{q}}_k$.

Next, consider the approximated squared norm of the torque at $t = t_{k+1}$

$$\begin{aligned}\|\boldsymbol{\tau}_{k+1}\|^2 \approx\ & \ddot{\boldsymbol{q}}_{k+1}^T\boldsymbol{M}_{k^+}^2\ddot{\boldsymbol{q}}_{k+1} \\ & + 2\left(\boldsymbol{S}_{k^+}(\dot{\boldsymbol{q}}_k + \ddot{\boldsymbol{q}}_k T) + \boldsymbol{g}_{k^+}\right)^T\boldsymbol{M}_{k^+}\ddot{\boldsymbol{q}}_{k+1} \\ & + (\dot{\boldsymbol{q}}_k + \ddot{\boldsymbol{q}}_k T)^T\boldsymbol{S}_{k^+}^T\boldsymbol{S}_{k^+}(\dot{\boldsymbol{q}}_k + \ddot{\boldsymbol{q}}_k T) \\ & + 2\boldsymbol{g}_{k^+}^T\boldsymbol{S}_{k^+}(\dot{\boldsymbol{q}}_k + \ddot{\boldsymbol{q}}_k T) + \boldsymbol{g}_{k^+}^T\boldsymbol{g}_{k^+} \end{aligned} \tag{20}$$

with the following short-hands: $\boldsymbol{M}_{k^+} = \boldsymbol{M}(\boldsymbol{q}_k + \dot{\boldsymbol{q}}_k T), \boldsymbol{g}_{k^+} = \boldsymbol{g}(\boldsymbol{q}_k + \dot{\boldsymbol{q}}_k T), \boldsymbol{S}_{k^+} = \boldsymbol{S}(\boldsymbol{q}_k + \dot{\boldsymbol{q}}_k T, \dot{\boldsymbol{q}})$. And $\boldsymbol{S}$ is obtained through a convenient factorization of $\boldsymbol{c}(\boldsymbol{q},\dot{\boldsymbol{q}})$ given by $\boldsymbol{c}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{S}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}}$ in which matrix $\boldsymbol{S}$ yields skew-symmetry for $\dot{\boldsymbol{M}} - 2\boldsymbol{S}$.

A full derivation of the approximations can be found in [3].

Now, using (19) and (20), the nonlinear optimization problem (18) has the following LQ approximation

$$\begin{aligned}\min_{\ddot{q}_k,\ddot{q}_{k+1}} \quad & H_4 = \frac{1}{2}\left(\omega_k\|\boldsymbol{\tau}_k\|^2 + \omega_{k+1}\|\boldsymbol{\tau}_{k+1}\|^2\right)\\ \text{s.t.} \quad & \boldsymbol{\tau}_k = \boldsymbol{M}_k\ddot{\boldsymbol{q}}_k + \boldsymbol{n}_k,\\ & \ddot{\boldsymbol{x}}_k = \boldsymbol{J}_k\ddot{\boldsymbol{q}}_k + \boldsymbol{h}_k,\\ & \boldsymbol{\tau}_{k+1} = \boldsymbol{M}_{k^+}\ddot{\boldsymbol{q}}_{k+1} + \boldsymbol{S}_{k^+}(\dot{\boldsymbol{q}}_k + \ddot{\boldsymbol{q}}_k T) + \boldsymbol{g}_{k^+},\\ & \ddot{\boldsymbol{x}}_{k+1} = \boldsymbol{J}_{k^+}\ddot{\boldsymbol{q}}_{k+1} + (\boldsymbol{J}_{k^+} - \boldsymbol{J}_k)\ddot{\boldsymbol{q}}_k + \boldsymbol{h}_{k^+}\end{aligned} \tag{21}$$

Again, using Lagrangian multipliers, the solution can be written in the following form

$$\ddot{\boldsymbol{q}}^* = \boldsymbol{A}_Q^{\#}\boldsymbol{b} - (\boldsymbol{I} - \boldsymbol{A}_Q^{\#}\boldsymbol{A})\boldsymbol{Q}^{-1}\boldsymbol{r} \tag{22}$$

with the weighted pseudo-inverse

$$\boldsymbol{A}_Q^{\#} = \boldsymbol{Q}^{-1}\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{A}^T)^{-1}$$

using the following substitutions

$$\begin{aligned}\boldsymbol{Q} &= \begin{pmatrix}\omega_k\boldsymbol{M}_k^2 + \omega_{k+1}T^2\boldsymbol{S}_{k^+}^T\boldsymbol{S}_{k^+} & \omega_{k+1}T\boldsymbol{S}_{k^+}^T\boldsymbol{M}_{k^+}\\ symm & \omega_{k+1}\boldsymbol{M}_{k^+}^2\end{pmatrix},\\ \boldsymbol{r} &= \begin{pmatrix}\omega_k\boldsymbol{M}_k(\boldsymbol{S}_k\dot{\boldsymbol{q}}_k + \boldsymbol{g}_k) + \omega_{k+1}T\boldsymbol{S}_{k^+}^T(\boldsymbol{S}_{k^+}\dot{\boldsymbol{q}}_k + \boldsymbol{g}_{k^+})\\ \omega_{k+1}\boldsymbol{M}_{k^+}(\boldsymbol{S}_{k^+}\dot{\boldsymbol{q}}_k + \boldsymbol{g}_{k^+})\end{pmatrix},\\ \boldsymbol{A} &= \begin{pmatrix}\boldsymbol{J}_k & \boldsymbol{0}\\ \boldsymbol{J}_{k^+} - \boldsymbol{J}_k & \boldsymbol{J}_{k^+}\end{pmatrix}, \quad \boldsymbol{b} = \begin{pmatrix}\ddot{\boldsymbol{x}}_k - \boldsymbol{h}_k\\ \ddot{\boldsymbol{x}}_{k+1} - \boldsymbol{h}_{k^+}\end{pmatrix}\end{aligned} \tag{23}$$

Similarly, the task based constraint in (21) can be written in the matrix form as

$$\boldsymbol{A}\begin{pmatrix}\ddot{\boldsymbol{q}}_k\\ \ddot{\boldsymbol{q}}_{k+1}\end{pmatrix} = \boldsymbol{b}$$

Then the closed form solution is the concatenation of the two states

$$\ddot{\boldsymbol{q}}^* = \begin{pmatrix}\ddot{\boldsymbol{q}}_k^T & \ddot{\boldsymbol{q}}_{k+1}^T\end{pmatrix}^T \tag{24}$$

In the implementation, only $\ddot{\boldsymbol{q}}_k$ is used at the time instant $t = t_k$ while $\ddot{\boldsymbol{q}}_{k+1}$ is discarded. The position and velocity at the next time instant are obtained by applying euler integration, using (9). Similar to the instantaneous torque minimization methods, PD feedback is applied to the term $\ddot{\boldsymbol{x}}_k$ (*only*) in (23) using (6).

Dynamic Damping in the Null Space

Instead of minimizing the torque norms like in the previous methods (i.e. placing the norm as close as possible to zero), one could minimize the difference in norm with respect to a suitable desired target torque. Define

$$\boldsymbol{\tau}_{D_k} = -\boldsymbol{D}_k\boldsymbol{M}_k\dot{\boldsymbol{q}}_k \tag{25}$$

where $\boldsymbol{D}_k$ is a non-negative diagonal (damping) gain matrix and $\boldsymbol{M}_k\dot{\boldsymbol{q}}_k$ is the generalized momentum of the robot at $t = t_k$.

When $\boldsymbol{\tau}_k = \boldsymbol{\tau}_{D_k}$ and the damping matrix is in the form $\boldsymbol{D}_k = d\boldsymbol{I} > 0$, from (1), the joint acceleration $\ddot{\boldsymbol{q}}_k$ becomes

$$\ddot{\boldsymbol{q}}_k = -\boldsymbol{M}_k^{-1}(\boldsymbol{S}_k\dot{\boldsymbol{q}}_k + \boldsymbol{g}_k) - d\dot{\boldsymbol{q}}_k \tag{26}$$

From (26), we see that the effect of the desired torque (25) is always opposite to the current joint velocity, thus acting as a damper on the joint motion. This effect will eliminate or reduce undesired behaviour (whipping and oscillations) in previous methods when the joint velocity becomes too large.

Therefore, the objective function in (4) is modified as

$$\begin{aligned}H_1' &= \frac{1}{2}\left\|\boldsymbol{\tau}_k - \boldsymbol{\tau}_{D_k}\right\|^2\\ &= \frac{1}{2}\left\|\boldsymbol{M}_k\ddot{\boldsymbol{q}}_k + (\boldsymbol{S}_k + \boldsymbol{D}_k\boldsymbol{M}_k)\dot{\boldsymbol{q}}_k + \boldsymbol{g}_k\right\|^2\end{aligned} \tag{27}$$

And the solution $\ddot{\boldsymbol{q}}^*$ is obtained from (5) by changing $\boldsymbol{n}_k$ to be

$$\boldsymbol{n}_k = (\boldsymbol{S}_k + \boldsymbol{D}_k\boldsymbol{M}_k)\dot{\boldsymbol{q}}_k + \boldsymbol{g}_k$$

This solution will be denoted as MTND (i.e., MTN with damping).

Similarly the objective function in (18) is modified as

$$H_4' = \frac{1}{2}\left(\omega_k\left\|\boldsymbol{\tau}_k - \boldsymbol{\tau}_{D_k}\right\|^2 + \omega_{k+1}\left\|\boldsymbol{\tau}_{k+1} - \boldsymbol{\tau}_{D_{k^+}}\right\|^2\right) \tag{28}$$

in which $\boldsymbol{\tau}_{D_{k^+}} = -\boldsymbol{D}_{k+1}\boldsymbol{M}_{k^+}(\dot{\boldsymbol{q}}_k + T\ddot{\boldsymbol{q}}_k)$. The solution $\ddot{\boldsymbol{q}}^*$ is again obtained from (24) by using the same substitutions for $\boldsymbol{A}, \boldsymbol{b}$ in (23) and replacing every $\boldsymbol{S}_k$ in $\boldsymbol{r}$ with $(\boldsymbol{S}_k + \boldsymbol{D}_k\boldsymbol{M}_k)$ and every $\boldsymbol{S}_{k^+}$ in $\boldsymbol{Q}, \boldsymbol{r}$ with $(\boldsymbol{S}_{k^+} + \boldsymbol{D}_{k+1}\boldsymbol{M}_{k^+})$. This solution will be denoted as MBPD. A stabilizing feedback control action on the task error is included as in non-damping cases.

Simulation Results

We consider a 3R planar arm ($n = 3$) moving on a horizontal plane (no influence of gravity). The robot has 3 uniform links, each of length $l = 1$ [m], mass $m_l = 10$ [kg], and moment of inertia $I_l = m_l l^2/12$. The simulated end-effector motion is a rest-to-rest straight-line cartesian trajectory, with bang-bang acceleration profile. The end-effector orientation is unconstrained, giving one degree of redundancy ($m = 2$). The upper and lower torque limits for joints 1-3 are $\pm 54, \pm 24$ *and* ± 6 [N.m] and the initial configuration is $\boldsymbol{q}_0 = (-45^\circ\ 135^\circ\ -135^\circ)^T$. The integration step was fixed to $T_s = 0.001$ [s]. The MBP method uses equal weights $\omega_k = \omega_{k+1} = 1$, and a preview window of $T = 100T_s = 0.1$ [s]. In all methods, the feedback gains on the cartesian task error were $\boldsymbol{K}_P = 10\boldsymbol{I}$ and $\boldsymbol{K}_D = \boldsymbol{I}$. The joint damping matrix is chosen as $\boldsymbol{D}_k = 10\boldsymbol{I}$, constant at all discrete times for both MTND and MBPD methods.

First, the case of the short movement is considered where the end-effector makes a movement of 0.2 [m] in both x and y directions ($\|L_1\| = 0.2828$ [m]), with an acceleration/deceleration of 2 $[m/s^2]$ in both x and y directions ($\|A_1\| = 2.8284$ $[m/s^2]$).

Torque profiles and continual motion configurations are shown in Fig. 1. We compare the 4 methods discussed previously (MTN, MTNB, PTR, MBP). We notice that PTR and MTNB make almost full or complete

usage of the torque for joint 3 in order to reduce peaks during the whole trajectory, while MTN and MBP use minimal torques for the first two joints, but exceed the limit for the last one.

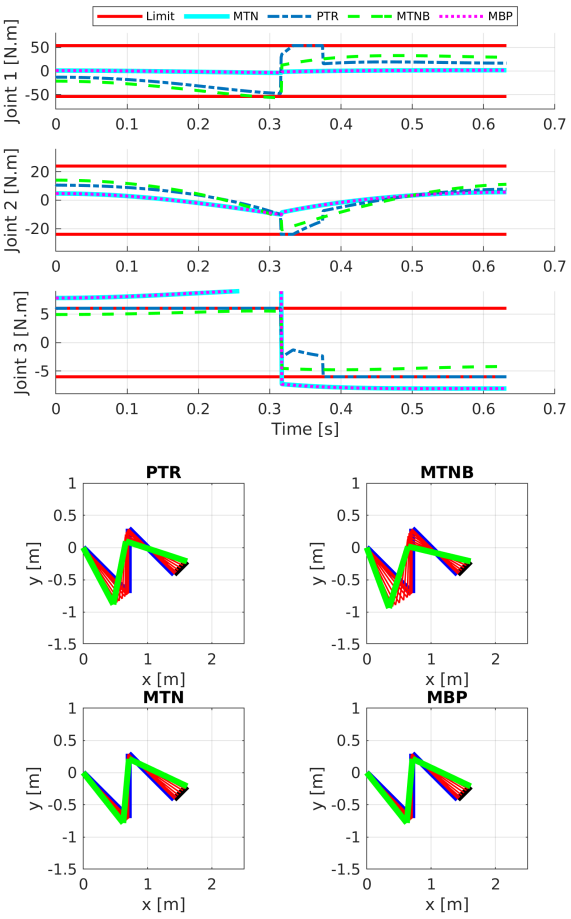

Figure 1: Torque and motion profiles for short movement

Second, the case of the long movement is considered where the end-effector makes a movement of 0.83 [m] in both x and y directions ($\|L_2\| = 1.1738$ [m]), with an acceleration/deceleration of 1 $[m/s^2]$ in both x and y directions ($\|A_2\| = 1.1412$ $[m/s^2]$).

Torque profiles and continual motion configurations are shown in Fig. 2. We notice the extremely large torques produced by MTNB and the whipping effect by MTN. On the other hand, since PTR and MBP use some information from a future state to compensate the locality of the algorithm, their performance is much more robust for longer movements (with the exception of MBP slightly exceeding the limit for joint 2). We cannot neglect the perturbation in torques realized by PTR in the beginning of the movement which shows indeed that MBP is better at addressing the torque instability issue. Also, for most of the motion MBP did not make full utilization of the torques for any joint.

In Fig. 3 we compare the torque profiles for the damping methods on both short and long movements. Since damping was not designed to deal with peak

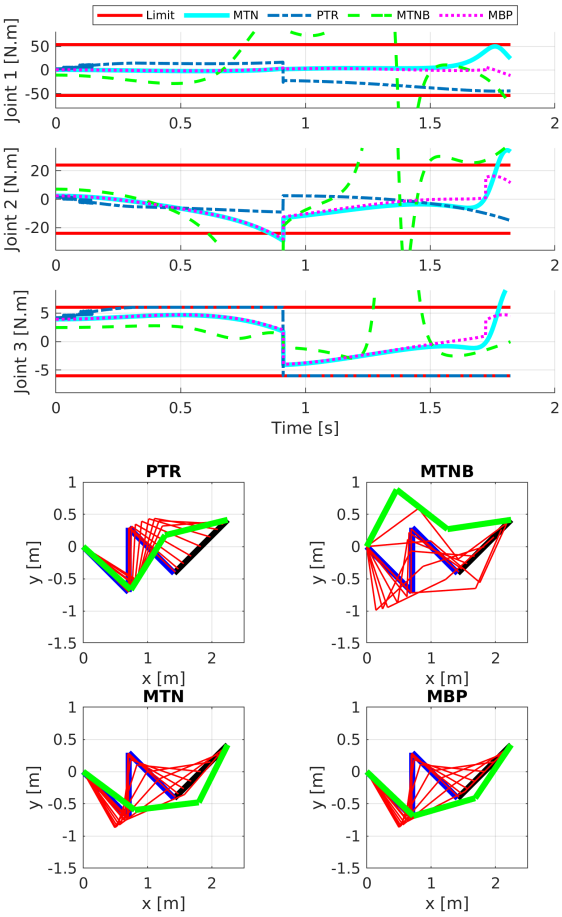

Figure 2: Torque and motion profiles for long movement

torques exceeding limits, we do not expect this behaviour to change. In fact, for the short movement, the torque profiles are exactly the same; with or without damping. For the long movement, we still notice the second joint still exceeding the limit (briefly), however at the end of the motion we notice the smoothness and stability of the torque command; compared with methods without damping which have a sudden whipping effect.

In all simulations, the cartesian tracking error was in the order of 10^{-4} [m]. Attempting to increase the preview window of MBP ($100T_s \leq T \leq 900T_s$) lead to similarly good behaviour by MBP.

Fig. 4 shows the norms of the joint velocities and torques for the short path using different methods. We notice that PTR and MTNB tend to use larger torques (thus, producing larger joint velocities) in order to keep all joints within torque limits at all times. While the main goal MTN and MBP is to minimize the torque norm (as observed) which results in exceeding torque limits at some intervals of the motion.

Fig. 5 shows the norms for the long path. We notice, again, the extremely large torques produced by MTNB and the whipping effect (near the end of the motion) by MTN. We observe again the ability of MBP to pre-

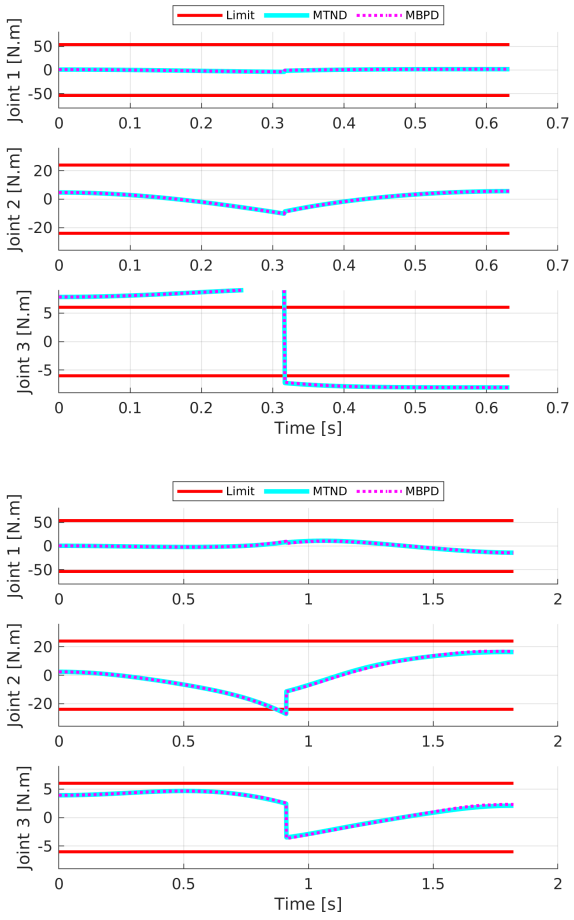

Figure 3: Torque profiles for damping methods on short movement (up) and long movement (down)

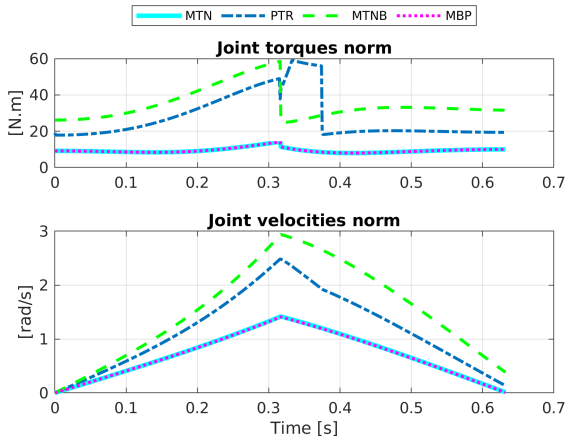

Figure 4: Joint torque and velocity norms for short movement

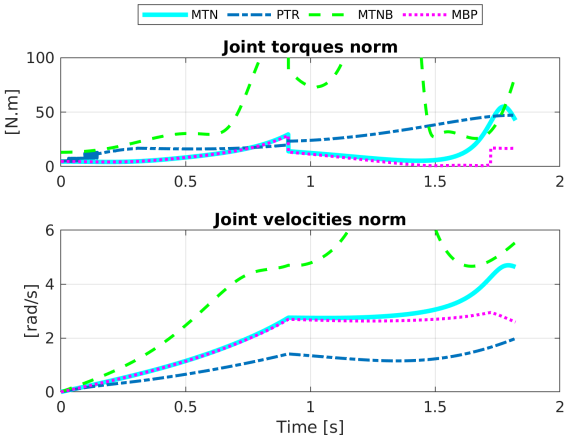

Figure 5: Joint torque and velocity norms for long movement

dict future torque requirements and producing minimal torque norm for the entire motion.

In Fig. 6 we compare the estimated torque at the future time instant by PTR ($p = 1$) and MBP ($p = 100$) for the short movement. As before, we see that the increased preview window allows MBP to better predict future requirements and thus reach minimal torque by the end of the motion, unlike PTR. It is worth to note that the dynamic approximations introduced to the future term in MBP's objective function (Eqn. 18) do not have a big effect here since the movement is short.

Contrarily, for the long movement (Fig. 7), we start to see the effect of the applied approximations in MBP. For $p = 100$, we notice the sudden whipping effect at the end of motion with one smaller peak near the middle of motion. For $p = 500$, we see more unstable behaviour with multiple increasing peaks during the motion. This estimated torque is expected to become more unstable as p increases (the approximations become less meaningful).

Finally, we visualized the motion using video simulations [1] that contrast the 4 methods against each other for both short and long movements. In case of the short movement, as we expect from the torque profiles, good utilization of the torques for joints 1 and 3 by PTR and MTNB clearly gives a different motion and prevents from exceeding the bounds. While for MTN and MBP, most of the desired motion is realized by actuating joint 3 (mainly) which resulted in passing the torque limits.

For the long movement, the unstable (and exceedingly large) torque behaviour by MTNB looks like '*impulses*' that translates to the weird motion seen in the video. For PTR we can interpret the movement as such: the end-effector is easing into the final position. Although MTN and MBP do realize the motion in a stable way, it feels borderline in the sense that if the straight-line trajectory was to be followed on some solid object placed on the horizontal plane, these methods would fail near the end since the third joint '*inflects*' in some sense and goes to the other side.

By looking at the all the results presented in this section, one could say that PTR produces the best results that can be applied in a *general* context (that does not require real-time control).

[1] Videos are attached to the report.

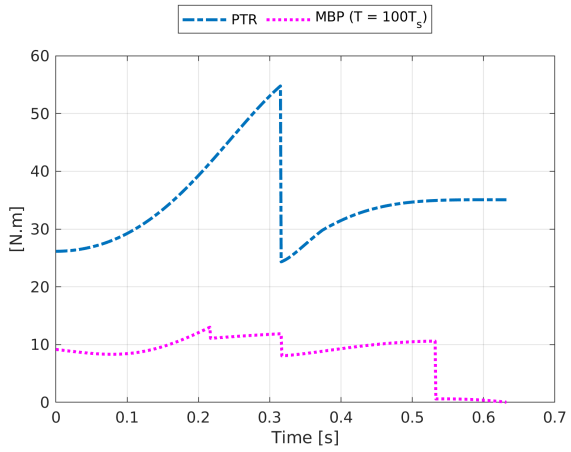

Figure 6: Future torque estimated by each method (short movement)

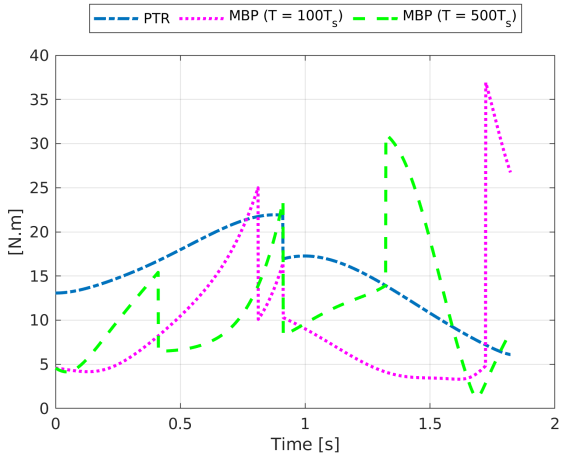

Figure 7: Future torque estimated by each method (long movement)

Conclusion

Different methods have been reviewed for kinematically redundant robots, with the aim of reducing joint torques consumption. Methods use local information (at current and/or future instants) about end-effector motion to minimize the joint torques. MTN and MTNB use only local information at current time instant leading to an unstable behaviour for longer movements. On the other hand, PTR and MBP attempt to *foresee* or *anticipate* future torque requirements and thus produce more stable results. Nonetheless, all methods fail to produce results that are stable and within bounds at the same time.

Future work will address the combination of MTNB and MBP methods to produce bounded and stable torque profiles for both short and long movements.

References

[1] Hollerbach, J. O. H. N. M., and Ki Suh. "Redundancy resolution of manipulators through torque optimization." IEEE Journal on Robotics and Automation 3.4 (1987): 308-316.

[2] Li, Degao, Andrew A. Goldenberg, and Jean W. Zu. "A new method of peak torque reduction with redundant manipulators." IEEE Transactions on Robotics and Automation 13.6 (1997): 845-853.

[3] Al Khudir, Khaled, et al. "Stable Torque Optimization for Redundant Robots using a Short Preview." IEEE Robotics and Automation Letters 4.2 (2019): 2046-2053.