On Maximizing Manipulability Index while Solving a Kinematics Task

Kévin Dufour, Wael Suleiman

Noname manuscript No.
(will be inserted by the editor)

On Maximizing Manipulability Index while Solving a Kinematics Task

Kévin Dufour · Wael Suleiman

the date of receipt and acceptance should be inserted later

Abstract In this paper, we investigate the problem of maximizing the manipulability index while solving a general Inverse Kinematics (IK) problem of a redundant industrial manipulator. Manipulability index has been extensively studied in the robotics literature and several formulae have been developed, nevertheless, they mainly only exploit the robot redundancy.

The general IK is formulated as a Quadratic Programming (QP) that can seamlessly incorporate inequality constraints, such as collision avoidance, and we propose two new formulae to integrate the manipulability index maximization into the QP-based IK solver. We then thoroughly analyze the performance of the proposed formulae in simulation and validate them on a real Baxter research robot.

The experimental results revealed the outperformance of the proposed formulae in comparison with the classical formula in the literature. Hence, providing a way to improve the manipulability index of a recorded trajectory, e.g. by learning by demonstration, or an offline generated one by a motion planning algorithm.

Keywords Manipulability index; Inverse Kinematics; QP-based IK solver; Collision avoidance

A preliminary version of this paper has been presented at 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2017) and published in [1].
K. Dufour and W. Suleiman
1- Electrical and Computer Engineering Department, Faculty of Engineering, University of Sherbrooke, Quebec, Canada
2- Institut Interdisciplinaire d'Innovation Technologique (3IT), University of Sherbrooke, Canada
Corresponding author: W. Suleiman
E-mail: Wael.Suleiman@USherbrooke.ca

1 Introduction

Collaborative robots (cobots) and their applications in manufacturing are very promising, as they are expected to allow small and medium-sized enterprises to regain competitiveness and efficiency thanks to the flexibility of their deployment. Cobots market is forecasted to grow at a compounded annual growth rate of 52.45% to reach 3 Billion USD by 2022 [2].

A major concern, however, is the safety of human operators of cobots. This is mainly because those robots operate without the conventional security fences usually used to separate heavy and dangerous industrial robots from humans, they are even designed to physically collaborate with humans to accomplish some tasks. To tackle the safety issue, several approaches have been proposed: I)- modifying the mechanical design of the robots and their material [3] or the actuation mechanisms [4,5,6], II)- adding a software layer to monitor the whole system to make it more human-friendly [7], III)- considering the safety criterion as an additional constraint in the planning and execution phases of the task.

Dealing with a dynamic environment while ensuring the safety of humans or not damaging the robot is also another challenging issue. A possible way to deal with this problem is by combining motion planning techniques, e.g. RRT* or PRM* [8,9], and IK methods. For instance, we generate at first a preliminary trajectory using motion planning methods to benefit from their probabilistic completeness and asymptotic optimality properties, and then modify it in real time by IK methods during the execution. The robot in this case could react to the presence of moving obstacles, such as a human or another robot, while satisfying some constraints, e.g. joint velocity or acceleration limits.

Besides, an interesting technique for programing a collaborative robot is Programing by Demonstration (PbD), which is a technique that provides a simple and intuitive way to program a collaborative robot from observations and demonstrations given by a human, who is not necessarily having programming abilities. Although PbD is one of the reasons behind the popularity of collaborative robots, the quality of resulting trajectories, e.g. being close to a singular configuration, cannot be easily figured out during the teaching phase.

Either the robot is executing a task in a dynamic environment or following a trajectory that has been taught by demonstrations, an important feature to assess is its ability to promptly react to unforeseen events. A way to measure this ability is by computing the manipulability index [10]. When this index becomes small and tends to zero that means that the robot is close to a singular configuration and its ability to move away from that pose is reduced. Hence, the objective of maximizing the manipulability index as much as possible during the motion execution is to keep the robot as far as possible from the singular configurations.

2 Contributions

The main contributions of this paper are:

- Proposing two new efficient formulae to solve the IK problem while maximizing the manipulability index. Those formulae are given in Section 4.2.
- Giving insights into the practical implementation of the new IK solvers as well as conducting experimental validation in simulation and on a real Baxter research robot. We also compare the performance of the proposed IK solvers against the conventional method in the literature. This contribution is detailed in Section 5.

3 Inverse Kinematics Problem Formulae

In this section, we overview the two main formulae for IK in the literature. Those formulae have been extensively studied and nowadays integrated in most of commercial control software for industrial manipulators.

3.1 Classical Formula

In order to accomplish a task defined by the user, IK methods convert a Cartesian pose of the robot end-effector into joint positions. The classical formula of IK problem is given as follows[1]:

$$\min_{\dot{\boldsymbol{q}}_t} \quad \|\boldsymbol{J}\dot{\boldsymbol{q}}_t - \dot{\boldsymbol{r}}_t\|^2 \tag{1}$$

where $\boldsymbol{J} \in \mathbb{R}^{6\times n}$ is the Jacobian matrix, $\dot{\boldsymbol{q}}_t \in \mathbb{R}^n$ is the joint velocity and $\dot{\boldsymbol{r}}_t = \begin{bmatrix}\boldsymbol{v}_t{}^T & \boldsymbol{\omega}_t{}^T\end{bmatrix}^T$ is the linear and angular desired velocity of the end-effector, whose components are defined such that:

$$\boldsymbol{v}_t = \frac{\boldsymbol{P}_d - \boldsymbol{P}_e}{T}$$
$$\boldsymbol{e}^{[\boldsymbol{\omega}_t\, T]^\wedge} = \boldsymbol{R}_d\, \boldsymbol{R}_e^T$$

where $\boldsymbol{P}_d$ and $\boldsymbol{P}_e$ are respectively the desired and actual Cartesian positions, and the rotation matrices $\boldsymbol{R}_d$ and $\boldsymbol{R}_e$ are defined in the same way. T is the sampling time, $\boldsymbol{e}^{\boldsymbol{M}}$ designs the matrix exponential of $\boldsymbol{M}$, and $[.\,]^\wedge$ is the skew operator defined as follows:

$$[.\,]^\wedge : \boldsymbol{\omega}_t = [\omega_x\, \omega_y\, \omega_z]^T \in \mathbb{R}^3 \rightarrow so(3)$$
$$[\boldsymbol{\omega}_t]^\wedge = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}$$

The optimization problem (1) can be efficiently solved using the pseudo-inverse technique [11,12] as follows:

$$\dot{\boldsymbol{q}}_t = \hat{\boldsymbol{J}}^\dagger \dot{\boldsymbol{r}}_t + (\boldsymbol{I}_6 - \hat{\boldsymbol{J}}^\dagger \boldsymbol{J})\boldsymbol{z} \tag{2}$$

where $\hat{\boldsymbol{J}}^\dagger = \boldsymbol{J}^T \left(\boldsymbol{J}\, \boldsymbol{J}^T\right)^{-1}$ is the well-known Moore-Penrose pseudo-inverse, $\boldsymbol{I}_6 \in \mathbb{R}^{6\times 6}$ is the identity matrix and $\boldsymbol{z}$ is an arbitrary vector. Recall that $\hat{\boldsymbol{J}}^\dagger \boldsymbol{J}$ is the orthogonal projector onto the kernel of $\boldsymbol{J}$. Eq. (2) clearly shows that the solution of the IK problem in the case of a redundant robot is not unique.

3.2 General Formula With Inequality Constraints

Even though the classical formula has a clear advantage, which is the low computational cost of its analytical solution, it cannot handle the case of inequality constraints, e.g. collision avoidance. In [1,13], we showed that allowing the robot to deviate from the pre-defined Cartesian trajectory, hence relaxing the constraint of following precisely that trajectory, increases the numerical robustness of the IK solver. Note that in some application, such as "pick-and-place", it is not necessary to precisely follow the end-effector pre-defined Cartesian trajectory as long as it reaches the goal pose.

Let $\boldsymbol{\delta} \in \mathbb{R}^6$ be a new variable, representing the deviation from the desired end-effector velocity. The general formula of IK with the trajectory relaxation is an optimization problem [1,13,14]:

[1] As a general rule, vectors and matrices are in **bold** font.

$$\begin{aligned} \min_{\dot{\boldsymbol{q}},\boldsymbol{\delta}} \quad & \frac{1}{2}\dot{\boldsymbol{q}}^T\boldsymbol{Q}\dot{\boldsymbol{q}} + \frac{1}{2}\boldsymbol{\delta}^T\boldsymbol{Q}_\delta\boldsymbol{\delta} \\ \text{subject to} \quad & \boldsymbol{J}\dot{\boldsymbol{q}} + \boldsymbol{\delta} = \dot{\boldsymbol{r}}_t \\ & \boldsymbol{b}^- \leq \boldsymbol{A}\dot{\boldsymbol{q}}_t \leq \boldsymbol{b}^+ \\ & \hat{\dot{\boldsymbol{q}}}^- \leq \dot{\boldsymbol{q}}_t \leq \hat{\dot{\boldsymbol{q}}}^+ \\ & \boldsymbol{\delta}^- \leq \boldsymbol{\delta} \leq \boldsymbol{\delta}^+ \end{aligned} \tag{3}$$

Where $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, $\boldsymbol{b}^+$ and $\boldsymbol{b}^- \in \mathbb{R}^m$ can be used for collision avoidance [15]. Refer to Appendix A for more details regarding the collision avoidance as a kinematics constraint.

$\boldsymbol{Q}_\delta(t) = f_e(t)\cdot\boldsymbol{I}_6$, where f_e is a function controlling the amplitude of the deviation. f_e must be very high at the beginning and at the end of the trajectory so the robot does not deviate from the trajectory and very low otherwise to allow the relaxation. Note that if the goal position is in collision with the obstacles, e.g. the end-effector is in collision, the robot will not reach the goal position and uses the relaxation parameter, $\boldsymbol{\delta}$, to avoid the collision while being as close as possible to the goal position.

In [13], we have studied several profile functions for f_e and the analysis revealed that a smooth joint acceleration can be achieved by a profile function that combines two polynomial functions as shown in Fig. 1. The value of Q_{high} is in the range $[10^4, 10^5]$ and in the range $[1, 10]$ for Q_{low}. Refer to [13] for detailed description and explanation.

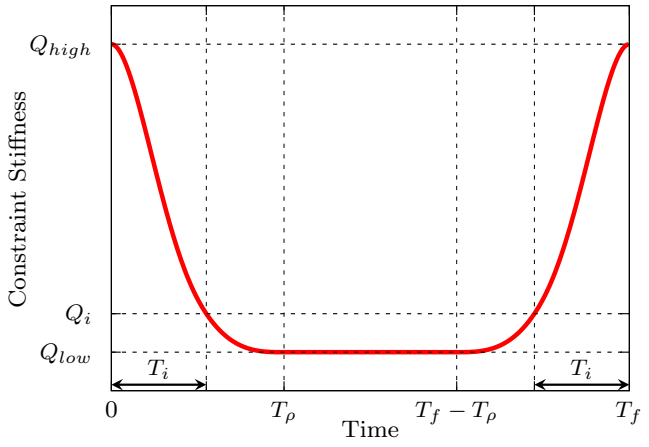


Fig. 1 Profile of constraint stiffness

Furthermore, the above formula can be easily adapted to consider the acceleration or even jerk limits of the joints [16].

The optimization problem (3) can be solved in an efficient way by transforming it into a QP problem as follows:

$$\begin{aligned} \min_{\boldsymbol{\mathcal{Z}}_t\in\mathbb{R}^{n+6}} \quad & \frac{1}{2}\boldsymbol{\mathcal{Z}}_t^T\boldsymbol{\mathcal{Q}}\boldsymbol{\mathcal{Z}}_t \\ \text{subject to} \quad & \boldsymbol{\mathcal{J}}\,\boldsymbol{\mathcal{Z}}_t = \dot{\boldsymbol{r}}_t \\ & \boldsymbol{\mathcal{B}}^- \leq \boldsymbol{\mathcal{A}}\boldsymbol{\mathcal{Z}}_t \leq \boldsymbol{\mathcal{B}}^+ \\ & \boldsymbol{\mathcal{Z}}^- \leq \boldsymbol{\mathcal{Z}}_t \leq \boldsymbol{\mathcal{Z}}^+ \end{aligned} \tag{4}$$

with

$$\boldsymbol{\mathcal{Z}}_t = \begin{bmatrix}\dot{\boldsymbol{q}}_t \\ \boldsymbol{\delta}\end{bmatrix}, \quad \boldsymbol{\mathcal{Q}} = \begin{bmatrix}\boldsymbol{Q} & \boldsymbol{0}_{n\times 6} \\ \boldsymbol{0}_{6\times n} & \boldsymbol{Q}_\delta\end{bmatrix}, \boldsymbol{\mathcal{J}} = \begin{bmatrix}\boldsymbol{J} & \boldsymbol{I}_6\end{bmatrix},$$
$$\boldsymbol{\mathcal{A}} = \begin{bmatrix}\boldsymbol{A} & \boldsymbol{0}_{m\times 6}\end{bmatrix}, \boldsymbol{\mathcal{B}}^+ = \boldsymbol{b}^+, \boldsymbol{\mathcal{B}}^- = \boldsymbol{b}^-,$$
$$\boldsymbol{\mathcal{Z}}^+ = \begin{bmatrix}\hat{\dot{\boldsymbol{q}}}^+ \\ \boldsymbol{\delta}^+\end{bmatrix}, \quad \boldsymbol{\mathcal{Z}}^- = \begin{bmatrix}\hat{\dot{\boldsymbol{q}}}^- \\ \boldsymbol{\delta}^-\end{bmatrix}$$

where $\boldsymbol{0}_{k\times l}$ designs the zero matrix in $\mathbb{R}^{k\times l}$.

4 Manipulability Index

The ability of a manipulator robot to escape unforeseen events, named manipulability, is defined by the following index [10]:

$$m(\boldsymbol{q}) = \sqrt{\det(\boldsymbol{J}\,\boldsymbol{J}^T)} = \sigma_1\sigma_2\cdots\sigma_n \tag{5}$$

with $(\sigma_i)_{1\leq i\leq n}$ are the singular values of $\boldsymbol{J}$.

The above index has been intensively studied in the robotics literature, it is very useful in practice as it gives an idea of the relative distance to singular configurations. It is obvious that $m(\boldsymbol{q}) \geq 0$ and $m(\boldsymbol{q}) = 0$ if and only if $\boldsymbol{q}$ is a singular configuration.

4.1 Integration into Classical IK Formula

In order to integrate the manipulability index into the solution of the classical formula (2), one can replace $\boldsymbol{z}$ by the following formula:

$$\begin{aligned} \boldsymbol{z} &= k_0\,\boldsymbol{\nabla} m \\ &= k_0\left(\frac{\partial m(\boldsymbol{q}_t)}{\partial \boldsymbol{q}_t}\right)^T \end{aligned} \tag{6}$$

where k_0 is a positive coefficient and $\boldsymbol{\nabla} m$ is the gradient of the manipulability index. As a result the analytical solution becomes:

$$\dot{\boldsymbol{q}}_t = \hat{\boldsymbol{J}}^\dagger\dot{\boldsymbol{r}}_t + k_0\left(\boldsymbol{I}_6 - \hat{\boldsymbol{J}}^\dagger\boldsymbol{J}\right)\left(\frac{\partial m(\boldsymbol{q}_t)}{\partial \boldsymbol{q}_t}\right)^T \tag{7}$$

4.2 Integration into General IK Formula

A direct way to integrate the manipulability index into (3) is to incorporate it into the objective function, thus we obtain:

$$\begin{aligned} \min_{\dot{\boldsymbol{q}}_t, \boldsymbol{q_t}, \boldsymbol{\delta}} \quad & \frac{1}{2}\,\dot{\boldsymbol{q}}_t^T Q \dot{\boldsymbol{q}}_t + \frac{1}{2}\boldsymbol{\delta}^T \boldsymbol{Q}_\delta \boldsymbol{\delta} - \alpha\, m(\boldsymbol{q}_t) \\ \text{subject to} \quad & \boldsymbol{J}\dot{\boldsymbol{q}}_t = \dot{\boldsymbol{r}}_t \\ & \boldsymbol{b}^- \leq \boldsymbol{A}\dot{\boldsymbol{q}}_t \leq \boldsymbol{b}^+ \\ & \hat{\dot{\boldsymbol{q}}}^- \leq \dot{\boldsymbol{q}}_t \leq \hat{\dot{\boldsymbol{q}}}^+ \\ & \boldsymbol{\delta}^- \leq \boldsymbol{\delta} \leq \boldsymbol{\delta}^+ \end{aligned} \tag{8}$$

where α is a positive coefficient, which is preceded by a negative sign in order to maximize $m(\boldsymbol{q}_t)$.

As $m(\boldsymbol{q}_t)$ is a nonlinear function in $\boldsymbol{q}_t$, Eq (8) becomes a nonlinear optimization problem which is hard to solve within a fixed time and therefore the hard real-time constraint might not be fulfilled.

In order to transform Eq (8) into a QP problem, the manipulability index is linearized as follows:

$$\begin{aligned} m(\boldsymbol{q}_t) &= m(\boldsymbol{q}_{t-1}) + (\boldsymbol{\nabla} m)^T \, \Delta \boldsymbol{q}_t + \frac{1}{2} \Delta \boldsymbol{q}_t^T \, \boldsymbol{H}_m \Delta \boldsymbol{q}_t \\ &= m(\boldsymbol{q}_{t-1}) + T \, (\boldsymbol{\nabla} m)^T \, \dot{\boldsymbol{q}}_t + \frac{1}{2} T^2 \, \dot{\boldsymbol{q}}_t^T \boldsymbol{H}_m \dot{\boldsymbol{q}}_t \end{aligned} \tag{9}$$

where $\boldsymbol{H}_m$ is the Hessian matrix of m, and recall that $\boldsymbol{\nabla} m$ is its gradient, T is the control loop sampling time of the robot.

By replacing (9) into (8), the following equivalent optimization problem is obtained:

$$\begin{aligned} \min_{\dot{\boldsymbol{q}}_t, \boldsymbol{\delta}} \quad & \frac{1}{2}\,\dot{\boldsymbol{q}}_t^T \boldsymbol{Q} \dot{\boldsymbol{q}}_t + \frac{1}{2}\boldsymbol{\delta}^T \boldsymbol{Q}_\delta \boldsymbol{\delta} \quad \dots \\ & - \alpha \left(T \, (\boldsymbol{\nabla} m)^T \, \dot{\boldsymbol{q}}_t + \frac{1}{2} T^2 \, \dot{\boldsymbol{q}}_t^T \boldsymbol{H}_m \dot{\boldsymbol{q}}_t \right) \\ \text{subject to} \quad & \boldsymbol{J}\dot{\boldsymbol{q}}_t + \boldsymbol{\delta} = \dot{\boldsymbol{r}}_t \\ & \boldsymbol{b}^- \leq \boldsymbol{A}\dot{\boldsymbol{q}}_t \leq \boldsymbol{b}^+ \\ & \hat{\dot{\boldsymbol{q}}}^- \leq \dot{\boldsymbol{q}}_t \leq \hat{\dot{\boldsymbol{q}}}^+ \\ & \boldsymbol{\delta}^- \leq \boldsymbol{\delta} \leq \boldsymbol{\delta}^+ \end{aligned} \tag{10}$$

By defining the following parameters:

$$\boldsymbol{\mathcal{Z}}_t = \begin{bmatrix} \dot{\boldsymbol{q}}_t \\ \boldsymbol{\delta} \end{bmatrix}, \quad \boldsymbol{\mathcal{Q}} = \begin{bmatrix} \boldsymbol{Q} - \alpha T^2 \boldsymbol{H}_m & \boldsymbol{0}_{n\times 6} \\ \boldsymbol{0}_{6\times n} & \boldsymbol{Q}_\delta \end{bmatrix},$$

$$\boldsymbol{\mathcal{G}} = \begin{bmatrix} -\alpha T \boldsymbol{\nabla} m \\ \boldsymbol{0}_{6\times 1} \end{bmatrix}, \boldsymbol{\mathcal{J}} = \begin{bmatrix} \boldsymbol{J} & \boldsymbol{I}_6 \end{bmatrix},$$
$$\boldsymbol{\mathcal{A}} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{0}_{m\times 6} \end{bmatrix}, \boldsymbol{\mathcal{B}}^+ = \boldsymbol{b}^+, \boldsymbol{\mathcal{B}}^- = \boldsymbol{b}^-,$$
$$\boldsymbol{\mathcal{Z}}^+ = \begin{bmatrix} \hat{\dot{\boldsymbol{q}}}^+ \\ \boldsymbol{\delta}^+ \end{bmatrix}, \quad \boldsymbol{\mathcal{Z}}^- = \begin{bmatrix} \hat{\dot{\boldsymbol{q}}}^- \\ \boldsymbol{\delta}^- \end{bmatrix}$$

The formula (10) is transformed into the following standard QP problem:

$$\begin{aligned} \min_{\boldsymbol{\mathcal{Z}}_t \in \mathbb{R}^{n+6}} \quad & \frac{1}{2}\boldsymbol{\mathcal{Z}}_{\boldsymbol{t}}^{\boldsymbol{T}} \boldsymbol{\mathcal{Q}} \boldsymbol{\mathcal{Z}}_{\boldsymbol{t}} + \boldsymbol{\mathcal{G}}^T \boldsymbol{\mathcal{Z}}_t \\ \text{subject to} \quad & \boldsymbol{\mathcal{J}}\, \boldsymbol{\mathcal{Z}}_t = \dot{\boldsymbol{r}}_t \\ & \boldsymbol{\mathcal{B}}^- \leq \boldsymbol{\mathcal{A}} \boldsymbol{\mathcal{Z}}_t \leq \boldsymbol{\mathcal{B}}^+ \\ & \boldsymbol{\mathcal{Z}}^- \leq \boldsymbol{\mathcal{Z}}_t \leq \boldsymbol{\mathcal{Z}}^+ \end{aligned} \tag{11}$$

Another way to integrate the manipulability index into (3) is as an equality constraint. Let us start by introducing the following lemma.

Lemma 1 *If $h : \mathbb{R}^n \to \mathbb{R}^+$, the following optimization problem yields to the maximization of h:*

$$\begin{aligned} \min_{\dot{\boldsymbol{q}}_t, \boldsymbol{\delta}, \epsilon_h} \quad & \frac{1}{2}\,\dot{\boldsymbol{q}}_t^T Q \dot{\boldsymbol{q}}_t + \frac{1}{2}\boldsymbol{\delta}^T \boldsymbol{Q}_\delta \boldsymbol{\delta} + \frac{1}{2}\,\alpha\,\epsilon_h^2 \\ \textit{subject to} \quad & \boldsymbol{J}\dot{\boldsymbol{q}}_t = \dot{\boldsymbol{r}}_t \\ & h(\boldsymbol{q}_t) - T\,(\boldsymbol{\nabla} h)^T \, \dot{\boldsymbol{q}}_t = \epsilon_h \\ & \boldsymbol{b}^- \leq \boldsymbol{A}\dot{\boldsymbol{q}}_t \leq \boldsymbol{b}^+ \\ & \hat{\dot{\boldsymbol{q}}}^- \leq \dot{\boldsymbol{q}}_t \leq \hat{\dot{\boldsymbol{q}}}^+ \\ & \boldsymbol{\delta}^- \leq \boldsymbol{\delta} \leq \boldsymbol{\delta}^+ \end{aligned} \tag{12}$$

where α is a positive coefficient.

Proof If we only consider the contribution of the function h in Eq (12), we obtain:

$$\begin{aligned} \min_{\dot{\boldsymbol{q}}_t, \epsilon_h} \quad & \frac{1}{2}\,\epsilon_h^2 \\ \text{subject to} \quad & h(\boldsymbol{q}_t) - T\,(\boldsymbol{\nabla} h)^T \, \dot{\boldsymbol{q}}_t = \epsilon_h \end{aligned} \tag{13}$$

We also have:

$$h(\boldsymbol{q}_t) \approx h(\boldsymbol{q}_{t-1}) + T\,(\boldsymbol{\nabla} h)^T \dot{\boldsymbol{q}}_t$$

Then $\epsilon_h = h(\boldsymbol{q}_t) - T\,(\boldsymbol{\nabla} h)^T \dot{\boldsymbol{q}}_t \approx h(\boldsymbol{q_{t-1}}) \geqslant 0$, this is because h is a positive function.

As $\epsilon_h \geqslant 0$, Eq (13) leads to the minimization of ϵ_h, i.e. leads to the minimization of $h(\boldsymbol{q}_t) - T\,(\boldsymbol{\nabla} h)^T \dot{\boldsymbol{q}}_t$. Because $h(\boldsymbol{q}_t)$ is a constant in the problem, Eq (13) minimizes $-T\,(\boldsymbol{\nabla} h)^T \dot{\boldsymbol{q}}_t$ and thus maximizes $T\,(\boldsymbol{\nabla} h)^T \, \dot{\boldsymbol{q}}_t$, which is the variation of $h(\boldsymbol{q}_t)$. □

By applying Lemma 1 to the case of manipulability index, we obtain the following optimization problem:

$$
\begin{aligned}
\underset{\dot{\boldsymbol{q}}_t,\boldsymbol{\delta},\epsilon_m}{\text{minimize}} \quad & \frac{1}{2}\,\dot{\boldsymbol{q}}_t^T \boldsymbol{Q}\dot{\boldsymbol{q}}_t + \frac{1}{2}\boldsymbol{\delta}^T\boldsymbol{Q}_\delta\boldsymbol{\delta} + \frac{1}{2}\,\alpha\,\epsilon_m^2 \\
\text{subject to} \quad & \boldsymbol{J}\dot{\boldsymbol{q}}_t + \boldsymbol{\delta} = \dot{\boldsymbol{r}}_t \\
& m(\boldsymbol{q}_t) - T\,(\boldsymbol{\nabla} m)^T\,\dot{\boldsymbol{q}}_t = \epsilon_m \\
& \boldsymbol{b}^- \le \boldsymbol{A}\dot{\boldsymbol{q}}_t \le \boldsymbol{b}^+ \\
& \hat{\dot{\boldsymbol{q}}}^- \le \dot{\boldsymbol{q}}_t \le \hat{\dot{\boldsymbol{q}}}^+ \\
& \boldsymbol{\delta}^- \le \boldsymbol{\delta} \le \boldsymbol{\delta}^+
\end{aligned}
\tag{14}
$$

In a similar way, the above optimization problem can be transformed into the following standard QP problem:

$$
\begin{aligned}
\min_{\boldsymbol{\mathcal{Z}}_t\in\mathbb{R}^{n+7}} \quad & \frac{1}{2}\boldsymbol{\mathcal{Z}}_{\boldsymbol{t}}^{\boldsymbol{T}}\boldsymbol{\mathcal{Q}}\boldsymbol{\mathcal{Z}}_{\boldsymbol{t}} \\
\text{subject to} \quad & \boldsymbol{\mathcal{J}}\,\boldsymbol{\mathcal{Z}}_t = \begin{bmatrix} \dot{\boldsymbol{r}}_t \\ m(\boldsymbol{q}_t) \end{bmatrix} \\
& \boldsymbol{\mathcal{B}}^- \le \boldsymbol{\mathcal{A}}\boldsymbol{\mathcal{Z}}_t \le \boldsymbol{\mathcal{B}}^+ \\
& \boldsymbol{\mathcal{Z}}^- \le \boldsymbol{\mathcal{Z}}_t \le \boldsymbol{\mathcal{Z}}^+
\end{aligned}
\tag{15}
$$

where:

$$
\boldsymbol{\mathcal{Z}}_t = \begin{bmatrix} \dot{\boldsymbol{q}}_t \\ \boldsymbol{\delta} \\ \epsilon_m \end{bmatrix}, \quad \boldsymbol{\mathcal{Q}} = \begin{bmatrix} \boldsymbol{Q} & \boldsymbol{0}_{n\times 6} & \boldsymbol{0}_{n\times 1} \\ \boldsymbol{0}_{6\times n} & \boldsymbol{Q}_\delta & \boldsymbol{0}_{6\times 1} \\ \boldsymbol{0}_{1\times n} & \boldsymbol{0}_{1\times 6} & \alpha \end{bmatrix},
$$

$$
\boldsymbol{\mathcal{J}} = \begin{bmatrix} \boldsymbol{J} & \boldsymbol{I}_6 & \boldsymbol{0}_{6\times 1} \\ T\,(\boldsymbol{\nabla} m)^T & \boldsymbol{0}_{1\times 6} & 1 \end{bmatrix},
$$

$$
\boldsymbol{\mathcal{A}} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{0}_{m\times 7} \end{bmatrix}, \boldsymbol{\mathcal{B}}^+ = \boldsymbol{b}^+, \boldsymbol{\mathcal{B}}^- = \boldsymbol{b}^-,
$$

$$
\boldsymbol{\mathcal{Z}}^+ = \begin{bmatrix} \hat{\dot{\boldsymbol{q}}}^+ \\ \boldsymbol{\delta}^+ \end{bmatrix}, \quad \boldsymbol{\mathcal{Z}}^- = \begin{bmatrix} \hat{\dot{\boldsymbol{q}}}^- \\ \boldsymbol{\delta}^- \end{bmatrix}
$$

Claim 1 *Even though both formulae* (14) *and* (10) *lead to the maximization of* $m(\boldsymbol{q})$*, formula* (14) *is numerically more robust than formula* (10)*.*

Proof In order to compare the formulae (14) and (10), without loss of generality, we suppose that $\boldsymbol{Q}_{\boldsymbol{\delta}} = \boldsymbol{0}$. Let us first integrate the equality constraint on ϵ_m into the objective function of (14), that leads to the following equivalent optimization problem:

$$
\begin{aligned}
\min_{\dot{\boldsymbol{q}}_t} \quad & \frac{1}{2}\,\dot{\boldsymbol{q}}_t^T\,\left(\boldsymbol{Q} + \alpha T^2 \boldsymbol{\nabla} m\,(\boldsymbol{\nabla} m)^T\right)\dot{\boldsymbol{q}}_t - \alpha\, m\, T(\boldsymbol{\nabla} m)^T\,\dot{\boldsymbol{q}}_t \\
\text{subject to} \quad & \\
& \boldsymbol{J}\dot{\boldsymbol{q}}_t = \dot{\boldsymbol{r}}_t \\
& \boldsymbol{b}^- \le \boldsymbol{A}\dot{\boldsymbol{q}}_t \le \boldsymbol{b}^+ \\
& \hat{\dot{\boldsymbol{q}}}^- \le \dot{\boldsymbol{q}}_t \le \hat{\dot{\boldsymbol{q}}}^+
\end{aligned}
\tag{16}
$$

It is then easy to verify that, $\forall \alpha > 0$, the matrix $\boldsymbol{Q} + \alpha T^2 \boldsymbol{\nabla} m\,(\boldsymbol{\nabla} m)^T$ is always positive definite, and the optimization problem is therefore convex that can be solved in a weakly polynomial time. On the other hand, the matrix $\boldsymbol{Q} - \alpha\, T^2\, \boldsymbol{H}_m$ in (10) can become indefinite by increasing α, as a result the QP problem becomes non-convex and NP-hard. Our empirical numerical analysis in Section 5 has confirmed this claim, moreover, it revealed that finding a quasi-optimal and trajectory independent value for α is possible for (14), while in the case of (10) the quasi-optimal value for α depends on the desired trajectory. □

4.3 Gradient and Hessian Computation

The gradient of m ($\boldsymbol{\nabla} m$) can be computed either numerically or analytically. The numerical computation of the gradient is straightforward:

$$
\begin{aligned}
(\boldsymbol{\nabla} m)_i &= \frac{\partial m(\boldsymbol{q}_t)}{\partial q_i} \\
&= \frac{m(\boldsymbol{q}_t + \delta q_i \boldsymbol{E}_i) - m(\boldsymbol{q}_t - \delta q_i \boldsymbol{E}_i)}{2\delta q_i}
\end{aligned}
\tag{17}
$$

where $(\boldsymbol{\nabla} m)_i$ is the i^{th} element of vector $\boldsymbol{\nabla} m(\boldsymbol{q}_t)$, and $\boldsymbol{E}_i \in \mathbb{R}^n$ is defined as follows:

$$
\boldsymbol{E}_i = [0\ \ldots\ 0\ \underset{\uparrow\ i}{1}\ \ 0\ \ldots\ \ 0]^T
$$

A practical value for δq_i is in the range $[10^{-4}, 10^{-3}]$ rad. As one can figure out, it is easy to numerically compute $\boldsymbol{\nabla} m$ since only the manipulability index for different configurations is computed, which indirectly implies, according to (5), Jacobian matrix calculations.

$\boldsymbol{\nabla} m$ can also be computed analytically using the following formula (refer to Appendix B for more details):

$$
\frac{\partial m}{\partial q_i} = m \cdot tr\left(\left(\boldsymbol{J}\,\boldsymbol{J}^T\right)^{-1}\,\frac{\partial \boldsymbol{J}}{\partial \boldsymbol{q}_i}\,\boldsymbol{J}^T\right) \tag{18}
$$

where $tr(\boldsymbol{M})$ is the trace operator of matrix $\boldsymbol{M}$.

Note that the above analytical derivative is undefined at $m = 0$, while the numerical derivative is always defined $\forall m$.

Regarding the Hessian matrix, a direct way to compute it is the following:

$$
\begin{aligned}
(\boldsymbol{H}_m)_{i,j} &= \frac{\partial^2 m(\boldsymbol{q}_t)}{\partial q_i \partial q_j} \\
&= \frac{(\boldsymbol{\nabla} m)_j\,(\boldsymbol{q}_t + \delta q_i \boldsymbol{E}_i) - (\boldsymbol{\nabla} m)_j\,(\boldsymbol{q}_t - \delta q_i \boldsymbol{E}_i)}{2\delta q_i}
\end{aligned}
\tag{19}
$$

where:

$$(\boldsymbol{\nabla} m)_j \left(\boldsymbol{q}_t + \delta q_i \boldsymbol{E}_i\right) = \frac{m(\boldsymbol{q}_t + \delta q_i \boldsymbol{E}_i + \delta q_j \boldsymbol{E}_j) - m(\boldsymbol{q}_t + \delta q_i \boldsymbol{E}_i - \delta q_j \boldsymbol{E}_j)}{2\delta q_j}$$
$$(\boldsymbol{\nabla} m)_j \left(\boldsymbol{q}_t - \delta q_i \boldsymbol{E}_i\right) = \frac{m(\boldsymbol{q}_t - \delta q_i \boldsymbol{E}_i + \delta q_j \boldsymbol{E}_j) - m(\boldsymbol{q}_t - \delta q_i \boldsymbol{E}_i - \delta q_j \boldsymbol{E}_j)}{2\delta q_j}$$

One drawback of (19) is its expensive computational cost, as it implies the evaluation of $4\,n^2$ Jacobian matrices which seriously increases the total calculation time. An alternative method to significantly reduce the computational cost is to use Broyden-Fletcher-Goldfarb-Shanno (BFGS) method. The computation procedure using BFGS can be summarized as follows:

$$\begin{aligned}
\boldsymbol{y}_t &= \boldsymbol{\nabla} m(\boldsymbol{q}_t) - \boldsymbol{\nabla} m(\boldsymbol{q}_{t-1}) \\
\boldsymbol{s}_t &= \boldsymbol{q}_t - \boldsymbol{q}_{t-1} \\
\boldsymbol{\rho}_t &= \frac{1}{\boldsymbol{y}_t^T\, \boldsymbol{s}_t} \\
\boldsymbol{H_m}(\boldsymbol{q}_t) &= \left(\boldsymbol{I}_n - \boldsymbol{\rho}_t\, \boldsymbol{s}_t\, \boldsymbol{y}_t^T\right) \boldsymbol{H_m}(\boldsymbol{q}_{t-1}) \left(\boldsymbol{I}_n - \boldsymbol{\rho}_t\, \boldsymbol{y}_t\, \boldsymbol{s}_t^T\right) + \boldsymbol{\rho}_t\, \boldsymbol{s}_t\, \boldsymbol{s}_t^T
\end{aligned} \tag{20}$$

5 Experimental Results

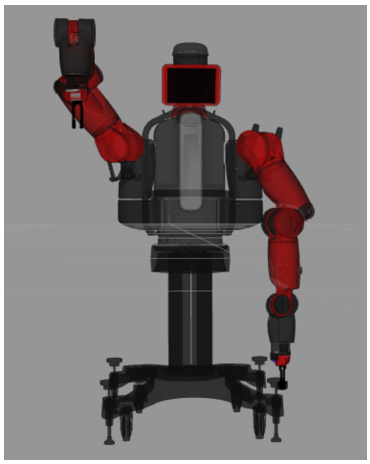
In order to evaluate the performance of the different proposed formulae, we have carried out several scenarios. In the first two scenarios, we compare the maximization of manipulability index using the proposed formula (10) with the conventional method in the literature (formula (7)).

In the third scenario we handle the case of obstacle avoidance while maximizing the manipulability index, this situation cannot be handled by the classical formula, hence pointing out one of the advantages of the proposed general formulae.

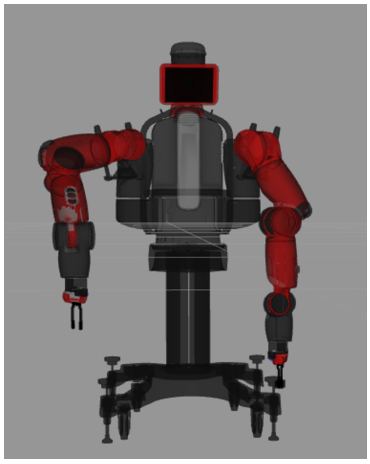
Finally, we compare in the fourth scenario the formulae (10) and (14) regarding numerical robustness and parameters tuning.

The different approaches have been tested in simulation using a Baxter research robot from Rethink Robotics and then validated on the real robot. A preliminary test has been conducted offline to measure the manipulability index for all possible configurations of the robot, thus to find an approximation of its maximal value. Note that all the manipulability indexes shown in this section are expressed relatively to that maximal value. The matrix $\boldsymbol{Q}$ in all formulae is equal to the identity matrix, and $T = 10\,\text{ms}$. The QP problems are solved by qpOASES [17], which has been proven to be efficient and real-time compatible. All experiments, in simulation and on the real robot, are performed on Intel® Core™ i5-3470 CPU @ 3.20 GHz PC with 8 GB RAM.

In the first scenario, the robot executes a simple trajectory in the Cartesian space along the vertical axis as shown in Fig. 2. The trajectory is obtained by cubic interpolation.



(a) Initial pose



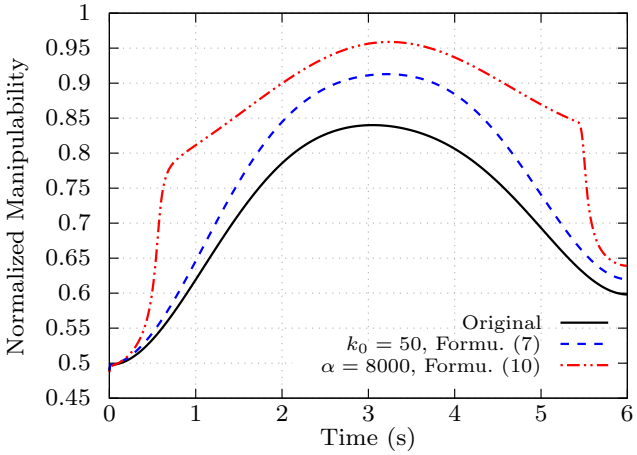
(b) Final pose

Fig. 2 Scenario 1: Initial and final poses of the robot

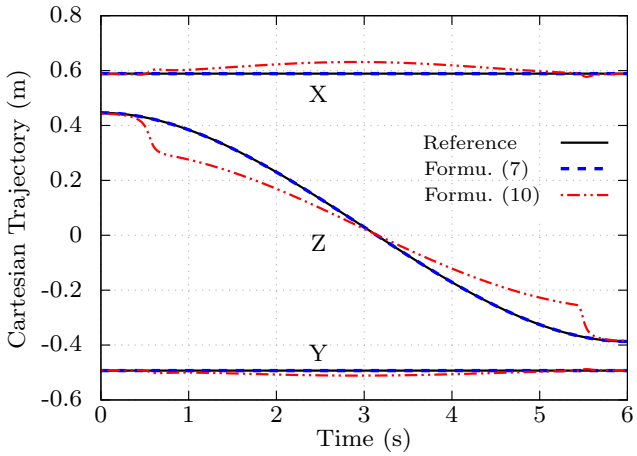
We compare in Fig. 3(a) the performance of the classical formula (7) and the proposed general formula (10), as can be seen the later formula significantly improve the manipulability index. This is because the classical formula only exploits the robot redundancy, while the general formation relaxes the constraint of following precisely the desired trajectory to maximize the manipulability index as much as possible. Note that the reported results are the best obtained results and the corresponding parameters (k_0 and α) are given in Fig. 3(a).

Fig. 3(b) shows the reference Cartesian trajectory of the end-effector and the obtained trajectories using the classical and general formulae, it points out that the reference trajectory is precisely tracked using the classical formula while small deviations can be observed in the case of general formula, however, the robot resumes

the task at the end to successfully reach the goal pose for the end-effector.

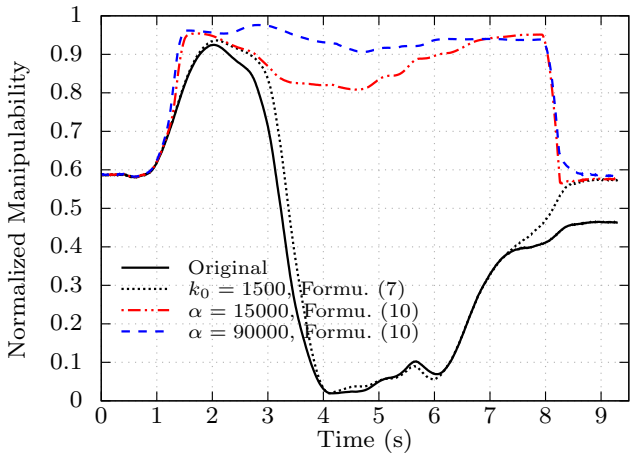


(a) Normalized manipulability index evolution

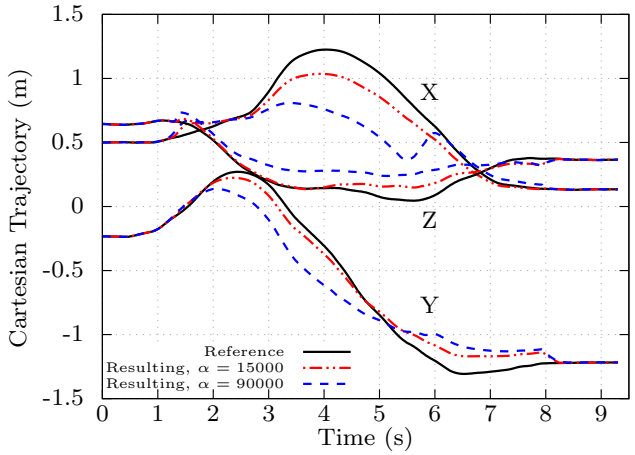


(b) Reference and obtained trajectories

Fig. 3 Scenario 1: Comparing formulae (7) and (10)



(a) Normalized manipulability index evolution



(b) Reference and obtained trajectories

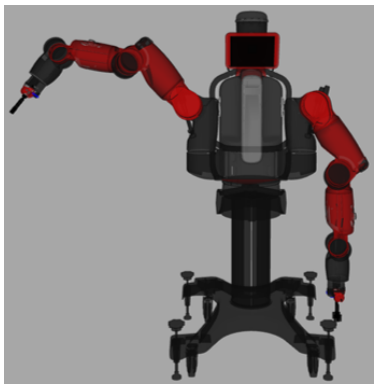
Fig. 4 Scenario 2: Comparing formulae (7) and (10)

In the second scenario, a trajectory is generated using the technique of programming by demonstrations. Although the robot end-effector was able to follow the pre-defined trajectory, the manipulability index significantly decreases during the motion and comes close to a singularity as can be seen in Fig. 4(a).

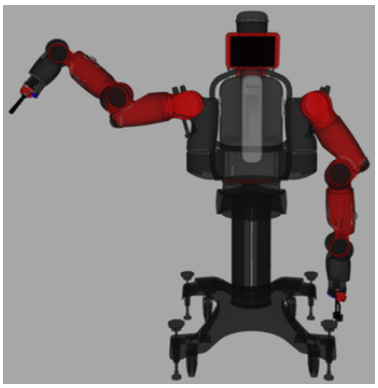
Fig. 4(a) also shows that a little improvement of the manipulability index is obtained by applying the classical formula, while the proposed formula (10) yields a very important improvement of the manipulability index. Moreover, the proposed formula allows the robot to completely escape from the singular configuration, hence improving the quality of the obtained trajectory. Fig. 4(b) presents the influence of α on the relaxation of the trajectory, and as it can be seen increasing α increases the deviations. However, even limited deviations, as with $\alpha = 15000$, leads to a high profile of manipulability index. Note that the trajectory obtained by applying the classical formula tracks precisely the reference, and it is not shown in Fig. 4(b) for a clarity reason.

Fig. 5 illustrates the difference between the final configurations in scenario 2 by applying the classical formula and proposed one (10), it shows that the configuration of the robot has been changed to a more convenient one regarding the manipulability, while having the same end-effector pose. The computation time of our implementation in the case of scenarios 1 and 2 is reported in Fig. 6. It clearly shows that the algorithm is real-time compatible as its computational time is less than the control loop sampling time $T = 10$ ms.

The case of collision avoidance is then addressed, as shown in Fig. 7, where a spherical obstacle that is located close to the elbow joint during the movement has been added. The desired trajectory is the same as



(a) formula (7)



(b) formula (10)

Fig. 5 Scenario 2: Final pose of the robot

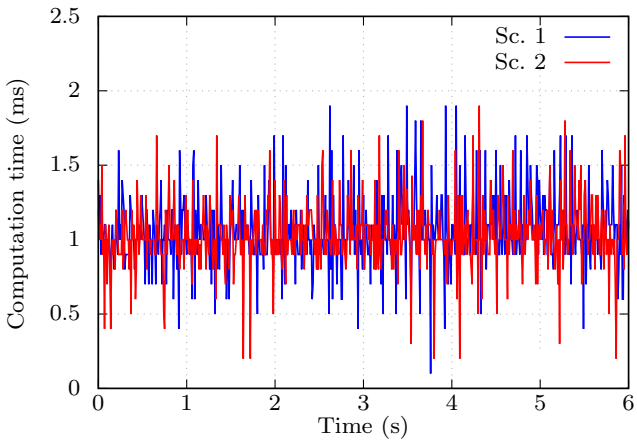
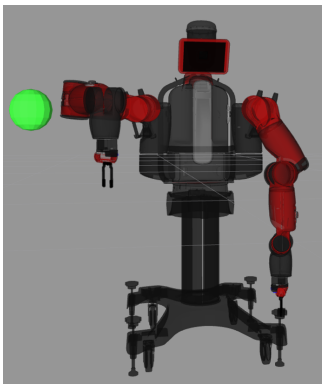
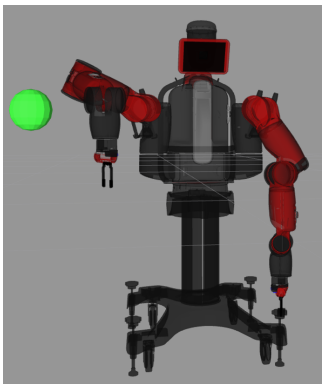


Fig. 6 Scenarios 1&2: Total computation time of our implementation of (10).

in Scenario 1. Fig. 7 illustrates the effect of the collision avoidance constraint on the configuration of the robot for a given time during the motion: the elbow, which becomes too close to the obstacle represented by a green sphere, is pushed away to satisfy the security distance. It should be noted that the end-effector has the same pose in both configurations as the Cartesian trajectory constraint is always satisfied.



(a) Without obstacle constraint



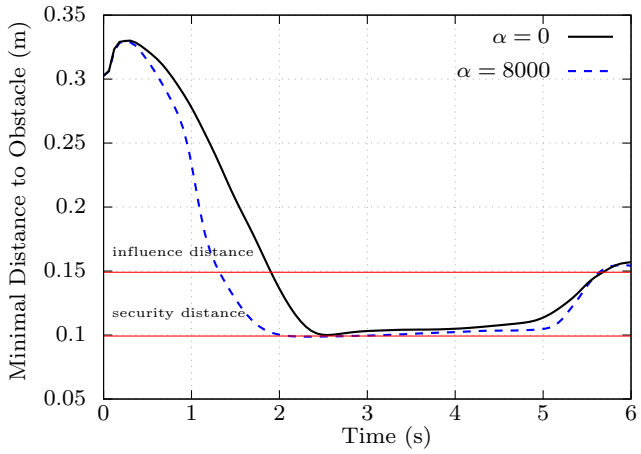
(b) With obstacle constraint

Fig. 7 Scenario 3: Effect of the obstacle avoidance constraint on the configuration of the robot for a given state by applying formula (10).

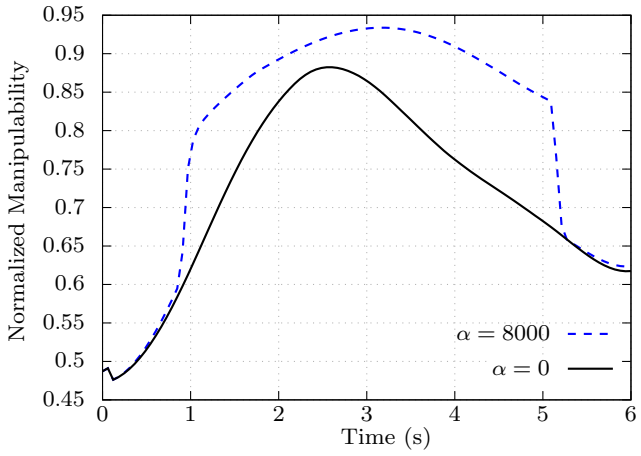
We applied the formula (10) to solve the kinematics task while avoiding the collision with the obstacle. The minimal distance between the robot and the obstacle is given in Fig. 8(a) for two values of α, $\alpha = 0$ means that the manipulability index is not maximized. It shows that the robot successfully avoids the obstacle and does not cross the security distance d_s while still being able to reach the goal. Fig. 8(b) points out that formula (10) succeeds in maximizing the manipulability index, hence providing a general framework to maximize the manipulability index while efficiently incorporating inequality constraints, such as collision avoidance.

Fig. 8(c) shows the computational time of the algorithm while considering the collision avoidance constraint, the algorithm is still real-time compatible despite the additional time of distance computations.

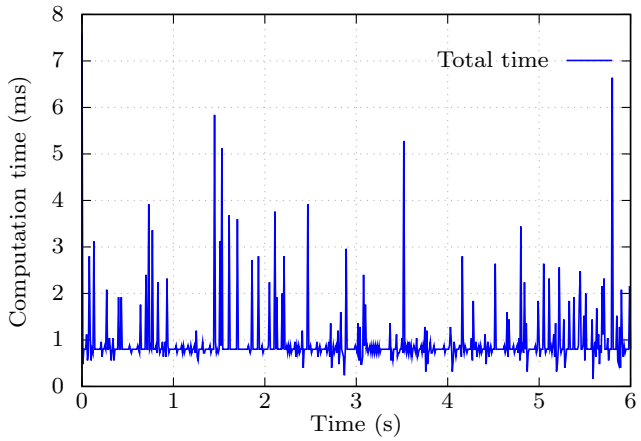
To compare the performance of formulae (14) and (10), five kinematics tasks have been defined. Each kinematics task consists in following a pre-defined trajectory of the robot end-effector, which was either obtained by using MoveIt! [18] or by recording a trajectory



(a) Minimal distance between the robot and the obstacle



(b) Normalized manipulability index evolution



(c) Total computation time of our implementation

Fig. 8 Scenario 3: Collision avoidance by applying formula (10).

of the robot joints while moving the robot arm between an initial and a goal positions of the robot end-effector (learning by demonstration technique).

Experiments have shown that both formulae successfully maximize the manipulability index. However, as pointed out in **Claim** 1, increasing α in formula (10) produced velocity oscillations and the QP problem became infeasible. On the other hand, formula (14) was more numerically stable and always feasible for the same values of α.

Table 1 Optimal coefficients for manipulability index maximization

Trajectory	$\mathcal{T}_1$	$\mathcal{T}_2$	$\mathcal{T}_3$	$\mathcal{T}_4$	$\mathcal{T}_5$
α (in thousands), Formu. (10)	50	70	70	120	90
α, Formu. (14)	5×10^5				

In Table 1, we investigated the identification of the optimal value of α using formulae (10) and (14). For both formulae, we have noticed that while increasing the value of α improves the manipulability profile, an optimal value can be identified and further increase of α does not much improve the manipulability index. Table 1 points out that α is non-unique using formula (10). On the other hand, a single coefficient for all five kinematics tasks has been identified, $\alpha = 5 \times 10^5$, for formula (14).

It is worth mentioning that using the unique value of α in formula (14) yielded comparable normalized manipulability profiles to those produced by formula (10) using the trajectory-dependent optimal α, but with inferior values of around 3% to 5% as can be seen in Fig. 9.

Therefore, from a practical point of view, formula (10) can be used for repetitive tasks where the optimal value of α can be determined offline for a given robot and a specific kinematics task, while formula (14) can be used for a given robot regardless of the kinematics task.

We also validated formula (14) on the real Baxter research robot as shown in Fig. 10. Our experiment with the real robot confirmed that formula (14) is real time compatible, and the robot successfully followed the computed joint trajectories. Moreover, that experiment pointed out that maximizing the manipulability index while relaxing the trajectory yielded a smoother trajectory than the initial provided one and the exerted torques by the joint actuators were also less.

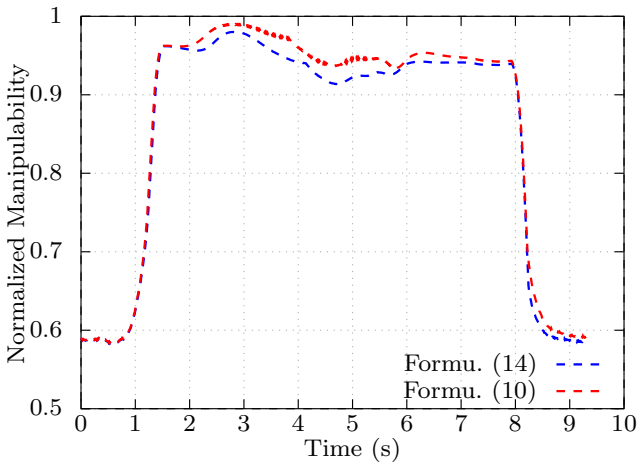


Fig. 9 Comparing the performance between formulae (14) and (10) for the trajectory $\mathcal{T}_3$.

6 Conclusion and Future Work

In this paper, we presented two formulae to integrate the manipulability index maximization into an optimization-based IK solver. The new proposed formulae significantly improved the manipulability index profile with a reasonable deviation from the desired trajectory of the end-effector. They outperformed the classical formula while being able to seamlessly integrate additional constraints such as collision avoidance.

Our experimental validation also revealed that integrating the manipulability index as an equality constraint is numerically more robust than directly integrating it into the objective function.

We have also verified that the proposed formulae are real-time compatible. Our experiments in simulation and on the real Baxter robot were carried out at a control frequency of 100 Hz. However, preliminary results using a PC with more computational power revealed that our proposed formulae can easily run with a control frequency of 1 kHz.

An application of proposed IK solvers is post-processing offline generated trajectories, such as trajectories generated through Learning from Programming by Demonstration (PbD) or motion planning techniques, to improve the trajectory quality regarding the manipulability index.

Future work will focus on integrating other security-related constraints such as taking into account human awareness and dealing with the presence of a human beside the robot.

A Collision avoidance as kinematics constraint

Let us define d as the distance between two convex objects, one being a part of the robot ($\mathcal{O}_1$) and the other is an obstacle ($\mathcal{O}_2$). In order to avoid the collision with the obstacle, we define two parameters, d_i and d_s, which are respectively influence and security distances. The following constraint is then activated if and only if d becomes smaller than d_i:

$$-\boldsymbol{n}^T \boldsymbol{J}(\boldsymbol{q}_t, \boldsymbol{p}_1)\dot{\boldsymbol{q}}_t \leq \xi \frac{d - d_s}{d_i - d_s} \quad \text{if} \quad d \leq d_i \tag{A.1}$$

where $\boldsymbol{n}$ is the unit vector between the pair of closest points $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$, as described in Fig. 11, ξ is a positive coefficient.

Let k be the number of distances between the robot and a set of obstacles satisfying $d \leq d_i$, the inequality constraint in (3) can then be rewritten as:

$$\boldsymbol{A}\dot{\boldsymbol{q}}_t \leq \boldsymbol{b}^+$$

where $\boldsymbol{A} \in \mathbb{R}^{k \times n}$ and $\boldsymbol{b}^+ \in \mathbb{R}^k$, and their j^{th} line is defined as follows:

$$\begin{aligned} \boldsymbol{A}_j &= -\boldsymbol{n}_j^T \boldsymbol{J}_j(\boldsymbol{q}_t, \boldsymbol{p}_1) \\ b_j^+ &= \xi \frac{d_j - d_s}{d_i - d_s} \end{aligned} \tag{A.2}$$

B Proof of Eq. (18)

If $\boldsymbol{A}(\boldsymbol{X}) \in \mathbb{R}^{m \times m}$ is a matrix that is a function of $\boldsymbol{X} = [x_1\, x_2 \cdots x_n]^T \in \mathbb{R}^n$, using Jacobi's formula we have:

$$\frac{\partial \det \boldsymbol{A}}{\partial x_i} = \left(\text{adj}\,(\boldsymbol{A})\, \frac{\partial \boldsymbol{A}}{\partial x_i} \right)$$

where $\text{adj}\,(\boldsymbol{A})$ is the adjugate operator of matrix $\boldsymbol{A}$.

Applying the above formula to $m^2 = \det(\boldsymbol{J}\boldsymbol{J}^T)$, we obtain:

$$\begin{aligned} 2m \frac{\partial m}{\partial q_i} &= tr \left(\text{adj}\,(\boldsymbol{J}\boldsymbol{J}^T)\, \frac{\partial (\boldsymbol{J}\boldsymbol{J}^T)}{\partial q_i} \right) \\ &= 2\, tr \left(\text{adj}\,(\boldsymbol{J}\,\boldsymbol{J}^T)\, \frac{\partial \boldsymbol{J}}{\partial q_i}\, \boldsymbol{J}^T \right) \end{aligned}$$

if $m \neq 0$, i.e. the robot is not in a singular configuration, we obtain the following formula:

$$\begin{aligned} \frac{\partial m}{\partial q_i} &= \frac{1}{m}\, tr \left(\text{adj}\,(\boldsymbol{J}\,\boldsymbol{J}^T)\, \frac{\partial \boldsymbol{J}}{\partial q_i}\, \boldsymbol{J}^T \right) \\ &= m \cdot tr \left((\boldsymbol{J}\,\boldsymbol{J}^T)^{-1}\, \frac{\partial \boldsymbol{J}}{\partial q_i}\, \boldsymbol{J}^T \right) \end{aligned}$$

References

1. K. Dufour and W. Suleiman, "On integrating manipulability index into inverse kinematics solver," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, Canada, 2017, pp. 6967–6972.
2. Technavio, "Global collaborative robots market 2018-2022," Market Research Reports, Tech. Rep., Sep. 2018.

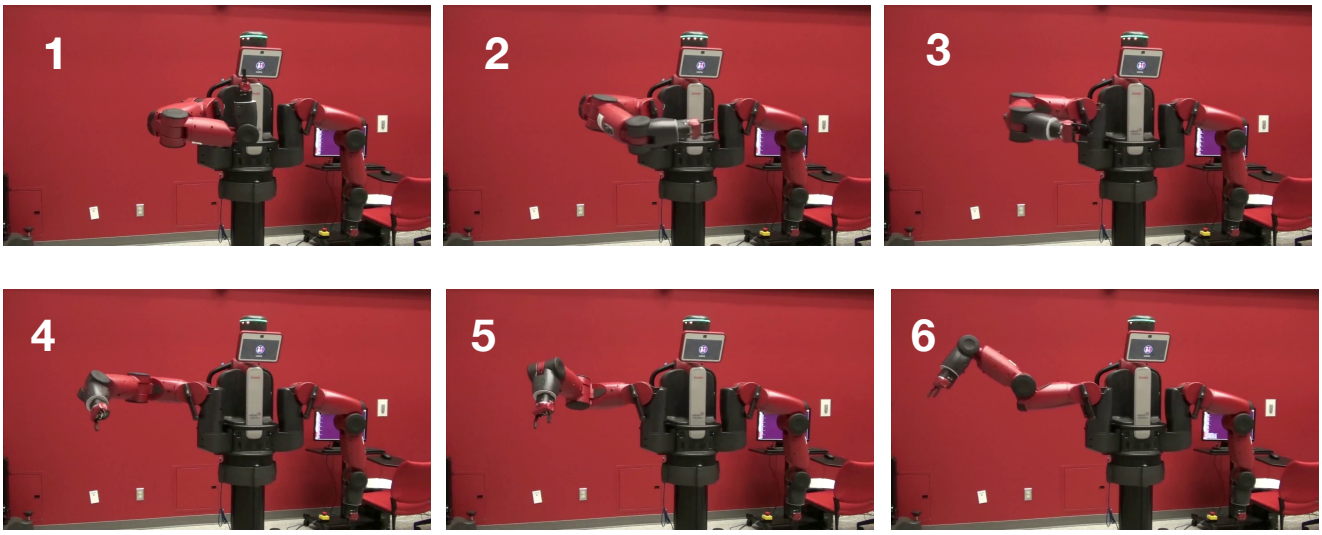


Fig. 10 Validation of Scenario 2 on the real Baxter research robot

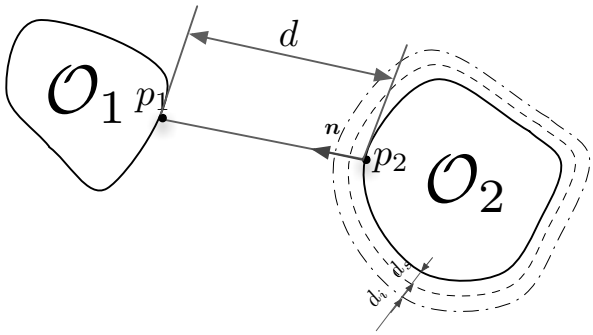


Fig. 11 Parameters for collision avoidance

3. A. Albu-Schäffer, S. Haddadin, C. Ott, A. Stemmer, T. Wimböck, and G. Hirzinger, "The DLR lightweight robot: design and control concepts for robots in human environments," *Industrial Robot: An International Journal*, vol. 34, no. 5, pp. 376–385, 2007.
4. G. A. Pratt and M. M. Williamson, "Series elastic actuators," in *International Conference on Intelligent Robots and Systems (IROS). Workshop 'Human Robot Interaction and Cooperative Robots'*, Pittsburgh, USA, 1995, pp. 399–406.
5. D. Ragonesi, S. Agrawal, W. Sample, and T. Rahman, "Series elastic actuator control of a powered exoskeleton," in *IEEE Annual International Conference of the Engineering in Medicine and Biology Society, EMBC, 2011*, Boston, USA, 2011, pp. 3515–3518.
6. G. Durandau and W. Suleiman, "User-safe orthosis based on compliant actuators: Mechanical design and control framework," in *55th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, Tsukuba, Japan, 2016, pp. 1508–1513.
7. D. Kulić and E. Croft, "Pre-collision safety strategies for human-robot interaction," *Autonomous Robots*, vol. 22, no. 2, pp. 149–164, 2007.
8. S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.
9. S. Klemm, J. Oberl, A. Hermann, A. Roennau, and T. Schamm, "RRT*-Connect : Faster , Asymptotically Optimal Motion Planning," in *IEEE Conference on Robotics and Biomimetics*, Zhuhai, China, 2015, pp. 1670–1677.
10. T. Yoshikawa, "Manipulability of Robotic Mechanisms," *The International Journal of Robotics Research*, vol. 4, no. 2, pp. 3–9, 1985.
11. Y. Nakamura, *Advanced Robotics: Redundancy and Optimization*. Addison-Wesley, 1991.
12. B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. Springer, 2009.
13. K. Dufour and W. Suleiman, "On inverse kinematics with nonlinear criteria: Trajectory relaxation," in *15th International Workshop on Advanced Motion Control (AMC)*, Tokyo, Japan, 2018, pp. 102–107.
14. F. Kanehiro, W. Suleiman, K. Miura, M. Morisawa, and E. Yoshida, "Feasible pattern generation method for humanoid robots," in *IEEE-RAS 9th International Conference on Humanoid Robots*, Paris, France, 2009, pp. 542–548.
15. B. Faverjon and P. Tournassoud, "A local based approach for path planning of manipulators with a high number of degrees of freedom," in *IEEE International Conference on Robotics and Automation (ICRA)*, North Carolina, USA, 1987, pp. 1152–1159.
16. W. Suleiman, "On inverse kinematics with inequality constraints: new insights into minimum jerk trajectory generation," *Advanced Robotics*, vol. 30, no. 17-18, pp. 1164–1172, 2016.
17. H. Ferreau, C. Kirches, A. Potschka, H. Bock, and M. Diehl, "qpOASES: A parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, no. 4, pp. 327–363, 2014.
18. I. A. Sucan and S. Chitta, "Moveit!" 2019. [Online]. Available: http://moveit.ros.org