2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

On Locally Optimal Redundancy Resolution using the Basis of the Null Space

Eugenio Monari[1], Yi Chen[1] and Rocco Vertechy[1]

Abstract— **This paper presents two methods for the computation of the null space velocity command in redundant robots. Both these methods resort to the solution of a constrained optimization problem. The first one is a formalization of the traditional Gradient Projection Method (GPM) which guarantees the respect of the joint bounds and a gradual activation/deactivation of the null space command. The second one, called Null Space Basis Optimal Linear Combination Method (NSBM), finds the optimal coefficients of a basis of the null space of the Jacobian, ensuring in turn that the joint bounds are respected and that the null space is activated and deactivated gradually. The two methods are applied to the case study of a welding application in which the null space command must avoid the collision between the robot and an obstacle. The comparison of the results of the case study shows that NSBM performs better than GPM. The proposed algorithms are also tested on a real robotic platform to demonstrate that their computational time is compatible with the real-time requirements of the robot.**

I. Introduction

Nowadays, redundant robots are frequently employed in the industrial field. In fact, these robots provide the possibility to follow the end effector trajectory while at the same time performing some additional task, such as obstacle, joint limits or singularity avoidance, as the number of degrees of freedom (i.e., dimension of the joints space) is larger than the number of variables needed to define the end effector task (i.e., dimension of the operational space). Moreover, the increasing number of collaborative robotics applications is making redundant robots even more important, because of the shared working area between the robot and one or more human operators, whose motions are unpredictable. In such a context, a good online handling of redundancy is crucial in order for the application to be safe and to avoid that the end effector movement is unnecessarily slowed down.

The most common approach to solve the inverse kinematics of a redundant robot is the Gradient Projection Method (GPM), originally introduced by Liegeois [1]. GPM computes the null space command as the product of the opposite of the gradient of some cost function by a matrix which projects it on the null space of the Jacobian, with the projection matrix being such that the obtained null space vector is the most similar to the opposite of the gradient.

Several cost functions are presented in the literature, for example in [1] and [2] a function of the distance from the central joint position is used for joint position limit avoidance; in [3], [4] and [5] obstacle avoidance is obtained using different types of functions of the distance between the robot and obstacles in the workspace; [6] and [7] propose to use different types of measures of the manipulability in order to ensure singularity avoidance.

Other methods ([8], [9]) recur to the solution of a quadratic programming problem, enforcing constraints on joint limits and on the distance from obstacles in the workspace; [10] extends [8] to include several objectives with different levels of priority. Workspace augmentation ([11]) uses a method for kinematic inversion based on the transpose of the Jacobian and extends it including constraints in the form of an error which is made to converge to zero. Full Space Parametrization and Parametrization through Null Space ([12] and [13]) seek a linear combination of the elements of a basis of the null space of the Jacobian to satisfy the constraints.

Also [14], [15] and [16] recur to the solution of an optimization problem in which the decision variables are the coefficients of the linear combination of the null space of the Jacobian, but [14], when applied to obstacle avoidance, finds the solution that is most similar to a desired "escape velocity", chosen without taking into account the joint limits, while [15] and [16] aim at finding the time-optimal trajectory respecting the joint limits.

While it would also be possible to approach redundancy resolution using inverse dynamics, such as in [17], in this paper only inverse kinematics is considered, with the assumption that the resulting desired trajectory is fed to an underlying torque controller which tracks that trajectory with sufficient accuracy. This paper proposes a formulation of GPM as a constrained optimization problem, which also ensures a gradual activation/deactivation of the null space command. Additionally, the Null Space Basis Optimal Linear Combination Method (NSBM) is presented. This method is also formalized as a constrained optimization problem, but does not require the computation of the gradient of the cost function and uses an approach which is similar to [14], [15] and [16], but has a simpler and more general formulation and can be applied for real-time computation. The proposed methods are then applied to an obstacle avoidance case study and their results are compared.

The article is organized as follows: Section II introduces the formalization of GPM and NSBM as constrained optimization problems; Section III describes the case study and presents the results of GPM and NSBM in MATLAB; Section IV describes how GPM and NSBM were implemented on a real robotic system satisfying the real-time constraints; Section V presents the conclusions of the work and its future

[1]Eugenio Monari, Yi Chen and Rocco Vertechy are with the Department of Industrial Engineering, University of Bologna, 40136 Bologna, Italy (email: eugenio.monari3@unibo.it; yi.chen4@unibo.it; rocco.vertechy@unibo.it)

developments.

II. GPM AND NSBM AS CONSTRAINED OPTIMIZATION PROBLEMS

A. Formalization of GPM with optimal gain selection

As already introduced, GPM solves the joint velocities inverse kinematics problem of a redundant manipulator in the following way

$$\dot{\mathbf{q}} = \mathbf{J}^{\dagger}\mathbf{v}_e - k\left(\mathbf{I}_n - \mathbf{J}^{\dagger}\mathbf{J}\right)\left(\frac{\partial w\left(\mathbf{q}\right)}{\partial \mathbf{q}}\right)^{T} \tag{1}$$

where $\mathbf{J} \in \mathbb{R}^{mxn}$ is the geometric Jacobian of the robot (m is the dimension of the operational space and n is the dimension of the joint space), $\mathbf{I}_n$ is the nxn identity matrix, $\mathbf{J}^{\dagger}$ is its Moore-Penrose pseudo-inverse (appropriately damped to avoid numerical errors near singularities), $\mathbf{v}_e \in \mathbb{R}^m$ is the vector containing the desired translational and/or rotational velocities of the end effector, $w\left(\mathbf{q}\right)$ is the function whose gradient is projected onto the null space and k is a user-defined positive gain. Since it is desired to formalize GPM as an optimization problem, it is necessary to identify an objective function that needs to be minimized and a set of constraints.

As for the objective function, we consider that k should be chosen as large as possible in order to minimize $w\left(\mathbf{q}\right)$, provided that the joint limits are respected, as stated in [18][1]. As a consequence, the objective function is as simple as $-k$, with k as the decision variable.

As for the implementation of joint limits, we consider a robotic system which imposes asymmetric position bounds $\mathbf{q}_{max}$/$\mathbf{q}_{min}$ and symmetric velocity, acceleration and jerk bounds $\dot{\mathbf{q}}_{max}$/$-\dot{\mathbf{q}}_{max}$, $\ddot{\mathbf{q}}_{max}$/$-\ddot{\mathbf{q}}_{max}$ and $\dddot{\mathbf{q}}_{max}$/$-\dddot{\mathbf{q}}_{max}$. It has to be observed that it is not sufficient to impose that the velocity command is such that the position, velocity, acceleration and jerk bounds are not violated at the next iteration. In fact, if one of the joints is very close to its maximum velocity, depending on the acceleration at the previous iteration, it may happen that imposing the acceleration value not to overcome the velocity limit at the next iteration generates a jerk that violates the jerk limit. So, in the proximity of the velocity limit, the acceleration must be reduced gradually to take also jerk bounds into account. An analogous consideration can be made regarding the proximity of the position bounds, to avoid violating the acceleration and jerk limits.

The case of the maximum acceleration limit is now treated. Similarly to the approach presented in [20], we consider that the joint acceleration limit must be such that, at any iteration, it is possible to reduce the acceleration to 0 applying a constant minimum jerk without exceeding the maximum velocity.

[1]Choosing k as large as possible is always the choice guaranteeing the maximum decrease of w, unless the configuration for which w is minimal is near. In that case, the best option would be to choose k so that the robot reaches the optimal configuration: if instead the maximum possible k is chosen, the robot starts oscillating around the optimal configuration, as observed in [19]. This case, however, will never happen in practice in the proposed framework, because the null space will be deactivated when the cost function exceeds a certain threshold.

At each iteration h and for each joint $i = 1, \ldots, n$ the velocity at the next iteration is approximated with backward Euler method as

$$\dot{q}_{i,h+1} = \dot{q}_{i,h} + \ddot{q}_{i,h+1}T \tag{2}$$

where T is the sampling time. Based on these considerations, and given the initial velocity $\dot{q}_{i,0}$ and acceleration $\ddot{q}_{i,0}$ (which is assumed to be positive) of a sequence of iterations, it is desired to find the value of acceleration $\ddot{q}_{i,1}$ such that the maximum velocity value in the sequence is $\dot{q}_{i,max}$ if the acceleration decreases at a rate of $-\dddot{q}_{i,max}$.

In this case, the values of acceleration and velocity at the h-th iteration are

$$\ddot{q}_{i,h} = \ddot{q}_{i,1} - (h-1)\,\dddot{q}_{i,max}T \tag{3}$$

$$\dot{q}_{i,h} = \dot{q}_{i,0} + h\ddot{q}_{i,1}T - \frac{(h-1)\,h}{2}\,\dddot{q}_{i,max}T^2. \tag{4}$$

So the acceleration reaches zero (and the velocity reaches its maximum value) at

$$h_{max} = \frac{\ddot{q}_{i,1}}{\dddot{q}_{i,max}T} + 1. \tag{5}$$

As a result, the maximum allowable value for $\ddot{q}_{i,1}$ is the solution of the equation

$$\dot{q}_{i,h_{max}} = \dot{q}_{i,max} \tag{6}$$

namely

$$\ddot{q}_{i,1,max} = -\dddot{q}_{i,max}\left(\frac{T}{2} - \sqrt{\frac{\dddot{q}_{i,max}T - 8\dot{q}_{i,0} + 8\dot{q}_{i,max}}{4\,\dddot{q}_{i,max}}}\right). \tag{7}$$

Approximating the position at the next iteration with forward Euler method it is possible to obtain maximum/minimum velocity values in an analogous way to (7).

Ultimately, the velocity and acceleration limits to be used for the computation of $\dot{\mathbf{q}}_{h+1}$ can be cast as

$$\dot{q}_{i,max_{safe},h+1} = \min\left(\dot{q}_{i,max}, \max\left(0, -\ddot{q}_{i,max}\left(\frac{T}{2} - \sqrt{\frac{\ddot{q}_{i,max}T^2 - 8q_{i,h+1} + 8\left(q_{i,max} - q_{i,mar}\right)}{4\ddot{q}_{i,max}}}\right)\right)\right) \tag{8}$$

$$\dot{q}_{i,min_{safe},h+1} = \max\left(-\dot{q}_{i,max}, \min\left(0, \ddot{q}_{i,max}\left(\frac{T}{2} - \sqrt{\frac{\ddot{q}_{max,i}T^2 + 8q_{i,h+1} + 8\left(q_{i,min} - q_{i,mar}\right)}{4\ddot{q}_{i,max}}}\right)\right)\right) \tag{9}$$

$$\ddot{q}_{i,max_{safe},h+1} = \min\left(\ddot{q}_{i,max}, \max\left(0, -\dddot{q}_{i,max}\left(\frac{T}{2} - \sqrt{\frac{\dddot{q}_{i,max}T^2 - 8\dot{q}_{i,h} + 8\left(\dot{q}_{i,max} - \dot{q}_{i,mar}\right)}{4\,\dddot{q}_{i,max}}}\right)\right)\right) \tag{10}$$

$$\ddot{q}_{i,min_{safe},h+1} = \max\left(-\ddot{q}_{i,max}, \min\left(0,\ \dddot{q}_{i,max}\left(\frac{T}{2}\right.\right.\right.$$
$$\left.\left.\left.-\sqrt{\frac{\dddot{q}_{max,i}T^2 + 8\dot{q}_{i,h} + 8\left(\dot{q}_{i,max} - \dot{q}_{i,mar}\right)}{4\,\dddot{q}_{i,max}}}\right)\right)\right) \quad (11)$$

where $q_{i,mar}$ and $\dot{q}_{i,mar}$ are safety margins that take into account the fact that the number of iterations to reach 0 velocity/acceleration as expressed in (6) may not be an integer number and that some additional space is needed to bring the acceleration from its initial value to $-\ddot{q}_{i,max}$ without violating the jerk limit to enforce the position limit. Additionally, in order to guarantee a gradual activation/deactivation of the null space, more limits are set on the null space command

$$\dot{\mathbf{q}}_{null} = -k\left(\mathbf{I}_n - \mathbf{J}^{\dagger}\mathbf{J}\right)\left(\frac{\partial w\left(\mathbf{q}\right)}{\partial \mathbf{q}}\right)^T \quad (12)$$

so that the minimum/maximum values of the null space command are the minimum/maximum velocity when $w\left(\mathbf{q}\right)$ is lower than a threshold $w_{th,1}$ and then linearly decrease until they become 0 when a second threshold $w_{th,2}$ is crossed ($w_{th,1} < w_{th,2}$). $w_{th,1}$ is the value below which it is desired to fully exploit the robot potential in terms of decrease of the cost function; $w_{th,2}$, instead, is tuned to avoid abruptly and repeatedly switching between the activation and deactivation of the null space. The limits are expressed in the form

$$-s\left(w\left(\mathbf{q}\right)\right)\dot{\mathbf{q}}_{max} \leq \dot{\mathbf{q}}_{null} \leq s\left(w\left(\mathbf{q}\right)\right)\dot{\mathbf{q}}_{max} \quad (13)$$

where $s\left(w\left(\mathbf{q}\right)\right)$ is a factor ranging from 0 to 1 defined as

$$s\left(w\left(\mathbf{q}\right)\right) = \begin{cases} 1 & \text{if } w\left(\mathbf{q}\right) \leq w_{th,1} \\ m_{th}w\left(\mathbf{q}\right) + q_{th} & \text{if } w_{th,1} \leq w\left(\mathbf{q}\right) \leq w_{th,2} \\ 0 & \text{if } w\left(\mathbf{q}\right) \geq w_{th,2} \end{cases} \quad (14)$$

where $m_{th} = \frac{1}{(w_{th,1} - w_{th,2})}$ and $q_{th} = \frac{w_{th,2}}{w_{th,2} - w_{th,1}}$.
Based on all the previous considerations, with $\dot{\mathbf{q}}_{h+1}$ defined as in (1) and approximating $\ddot{\mathbf{q}}_{h+1}$ and $\dddot{\mathbf{q}}_{h+1}$ with backward Euler method as in (2), the gain k of the null space command is the solution of the following optimization problem

$$\min_{k} \quad -k \quad (15a)$$
$$\text{s.t.} \quad \dot{\mathbf{q}}_{min_{safe},h+1} \leq \dot{\mathbf{q}}_{h+1} \leq \dot{\mathbf{q}}_{max_{safe},h+1} \quad (15b)$$
$$\ddot{\mathbf{q}}_{min_{safe},h+1} \leq \ddot{\mathbf{q}}_{h+1} \leq \ddot{\mathbf{q}}_{max_{safe},h+1} \quad (15c)$$
$$-\dddot{\mathbf{q}}_{max} \leq \dddot{\mathbf{q}}_{h+1} \leq \dddot{\mathbf{q}}_{max} \quad (15d)$$
$$-s\dot{\mathbf{q}}_{max} \leq \dot{\mathbf{q}}_{null,h+1} \leq s\dot{\mathbf{q}}_{max}. \quad (15e)$$

B. Formalization of NSBM and comparison with GPM

As introduced in [16], a solution of the inverse kinematics problem which is alternative to the one of (1) is

$$\dot{\mathbf{q}} = \mathbf{J}^{\dagger}\mathbf{v}_e + \mathbf{B}\mathbf{a} \quad (16)$$

where $\mathbf{B} \in \mathbb{R}^{nxr}$ is a matrix whose columns are the vectors of a basis of the null space of $\mathbf{J}$ ($r = n - m$ is the dimension of the null space) and $\mathbf{a} \in \mathbb{R}^r$ is a vector of arbitrary coefficients, so that the null space command is a linear combination of the vectors of the null space basis. In this framework it is possible to define

$$\mathbf{q}_{new} = \mathbf{q}_{previous} + \left(\mathbf{J}^{\dagger}\mathbf{v}_e + \mathbf{B}\mathbf{a}\right)T \quad (17)$$

which is the joint configuration that will be obtained at the next iteration for a certain choice of $\mathbf{a}$. Remembering that the optimization problem that is being solved is a local one, namely it considers only a one-iteration horizon, it can be concluded that the best possible null space command is the one for which $w\left(\mathbf{q}_{new}\right)$ is minimized.
We therefore define NSBM optimization problem with the same constraints as in (15), with $\dot{\mathbf{q}}_{h+1}$ defined as in (16) and the null space command used in (15e) defined as

$$\dot{\mathbf{q}}_{null} = \mathbf{B}\mathbf{a} \quad (18)$$

but substituting (15a) with

$$\min_{\mathbf{a}} \quad w\left(\mathbf{q}_{new}\right). \quad (19)$$

The reason why an alternative method to GPM is introduced is that, as stated in [21], GPM constrains the null space vector to have the same direction of $\left(\mathbf{I}_n - \mathbf{J}^{\dagger}\mathbf{J}\right)\left(\frac{\partial w(\mathbf{q})}{\partial \mathbf{q}}\right)^T$, namely GPM allows exploring only a one-dimensional subspace of the null space of $\mathbf{J}$, but the best possible solution does not necessarily belong to this subspace. In fact, the vector $-\left(\frac{\partial w(\mathbf{q})}{\partial \mathbf{q}}\right)^T$ is along the direction of maximum decrease of the cost function, but since this vector has to be projected onto the null space of $\mathbf{J}$ and the limits (15b)-(15e) have to be enforced, it is not guaranteed that the direction $-\left(\mathbf{I}_n - \mathbf{J}^{\dagger}\mathbf{J}\right)\left(\frac{\partial w(\mathbf{q})}{\partial \mathbf{q}}\right)^T$ is the optimal one. NSBM, instead, spans the entire solution space and therefore intrinsically provides the locally optimal solution, and in case GPM already provides the optimal solution, the solution of NSBM is eventually the same as that of GPM.

III. Simulation results

A. Test description

To provide an intuitive interpretation of the improvement of NSBM over GPM, some tests were performed with the seven-degree-of-freedom Franka Emika Panda robot. The manipulator is shown in Fig. 1, while joint position, velocity, acceleration and jerk limits are shown in Table I. If the velocity commands sent to the joints respect these limits, an internal torque controller provided by the developers is able to track accurately the desired trajectories. This is the reason why inverse kinematics is used in this case study instead of inverse dynamics.
The considered case study is a welding task, in which the welding tip attached to the end effector must maintain its position and orientation with respect to the surface, but can rotate freely around its axis. At the same time a plane moves in the direction of the robot and the null space command must be such to avoid the collision, as is shown in Fig. 2.
In this case the movement of the plane is pre-defined, but in a real-world scenario GPM and NSBM would be useful when

TABLE I

JOINT BOUNDS: THE LIMITS ARE THOSE OF THE REAL ROBOT [22], WITH THE ONLY EXCEPTION OF THE MAXIMUM VELOCITY OF JOINT 2, WHICH IS TAKEN AS 0.5 RAD/S INSTEAD OF 2.175 RAD/S TO EMPHASIZE THE DIFFERENCE BETWEEN THE TWO ALGORITHMS AND THE BETTER PERFORMANCE OF NSBM WITH RESPECT TO GPM.

	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6	Joint 7	Unit
q_{max}	2.8973	1.7628	2.8973	-0.0698	2.8973	3.7525	2.8973	rad
q_{min}	-2.8973	-1.7628	-2.8973	-3.0718	-2.8973	-0.0175	-2.8973	rad
$\dot{q}_{max}$	2.175	0.5	2.175	2.175	2.61	2.61	2.61	rad/s
$\ddot{q}_{max}$	15	7.5	10	12.5	15	20	20	rad/s^2
$\dddot{q}_{max}$	7500	3750	5000	6250	7500	10000	10000	rad/s^3

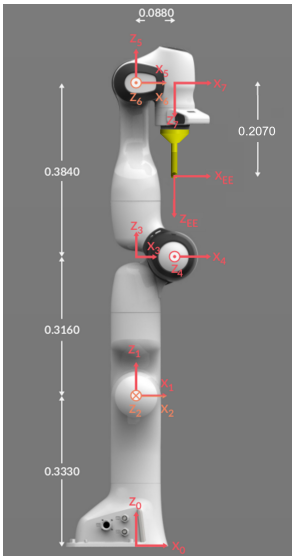

Fig. 1. Franka Emika Panda robot and its reference frames, defined according to Craig's convention [22] [23]. The original picture was modified including a schematic representation of an end effector.

the trajectory of the obstacle is not known in advance but is acquired by sensors (e.g. cameras) and as a consequence it would be impossible to perform obstacle avoidance through a priori path planning.

The function $w(\mathbf{q})$ that is used in the GPM and NSBM algorithms is chosen as the opposite of the distance between the center of the fourth reference frame and the plane, which is an approximation of the distance between the robot and the plane[2].

The dimension of the operational space is 5 instead of 6: the velocity command is set as $\mathbf{v}_e = \mathbf{0}$ and does not include the angular velocity component around y_{EE}, while the Jacobian does not include the row related to the rotation around y_{EE} as well. In this way the orientation around y_{EE} is left free

[2]Considering only the center of the fourth reference frame is not a good estimation of the distance between the robot and the plane in general. However, in the considered case studies, the point of the robot which is nearest to the plane is always close to the center of the fourth reference frame, so this approximation is reasonable for the considered tests. Additionally, the focus of the article is not on finding a cost function that allows to obtain the best possible obstacle avoidance, but on the algorithms to compute null space commands.

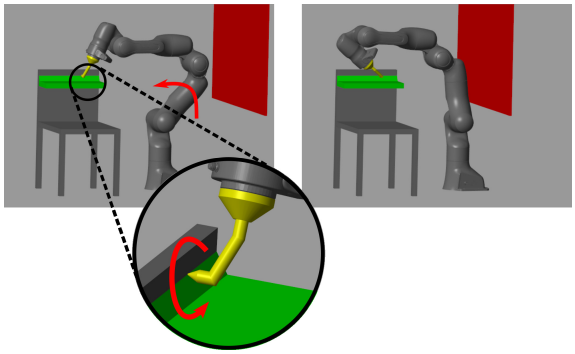

Fig. 2. Two frames from the welding application: the null space command enables the tip of the welder to rotate about its axis, as shown in the right frame

and can be modified by the null space command. In this case, GPM and NSBM can provide different results. Instead, if there were only one redundant degree of freedom, the solution of NSBM would necessarily coincide with the one of GPM, because the null space would be a one-dimensional subspace[3].

The initial joint angles are $\mathbf{q}_{init} = [0, \text{-}0.7854, 0, 2.3562, 0, 2.0071, 0]^T$ rad, while the plane is parallel to the y_0-z_0 plane, is initially at x_0 = -0.5 m and moves at a constant velocity of 0.4 m/s along axis x_0 until it reaches the position x_0 = -0.1 m. Finally, the thresholds $w_{th,1}$ and $w_{th,2}$ for the activation of the null space are set to 0.15 m and 0.25 m respectively. The tests were performed in the MATLAB environment, and the `fmincon` function was used to solve the optimization problems, setting SQP as the solver algorithm.

B. Results of the case study

GPM and NSBM are applied to the described case study, and their results are presented in the following.

[3]As a sidenote, it can be specified that this statement is always true unless $\left(\frac{\partial w(\mathbf{q})}{\partial \mathbf{q}}\right)^T$ belongs to the null space of $(\mathbf{I}_n - \mathbf{J}^\dagger\mathbf{J})$. In such a case, any choice of k will always produce a null space command equal to zero: there would exist null space directions that could lead to a decrease of the value of the cost function, but these directions are not explored by the GPM formulation. NSBM, instead, would be able to find the best null space command even in this situation without any inconvenience. This case is actually very rare, but it can be argued that in configurations which are close to this condition the value of k should be very high, which may lead to some numerical issues. This can therefore be considered as an additional advantage of NSBM over GPM.

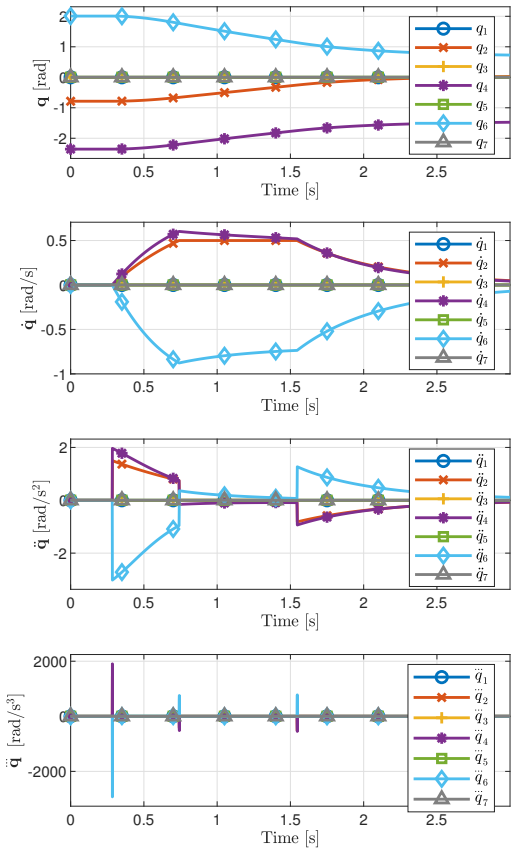

Fig. 3. Position, velocity, acceleration and jerk plots using GPM.

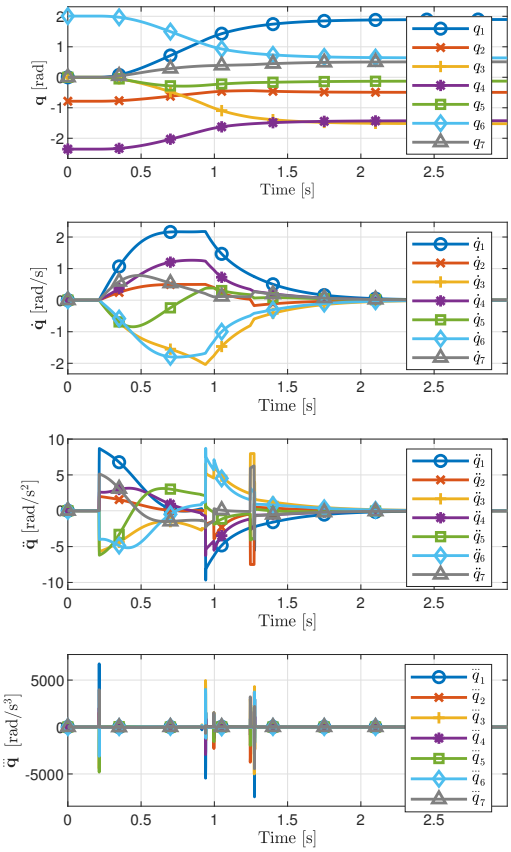

Fig. 4. Position, velocity, acceleration and jerk plots using NSBM.

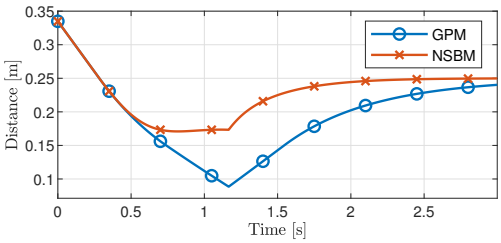

Fig. 5. Comparison of the value of the distance using GPM and NSBM.

Figs. 3 and 4 show the position, velocity, acceleration and jerk plots using GPM and NSBM, while Fig. 5 shows the distance between the center of the fourth frame and the plane when using GPM and NSBM. It can be observed that NSBM performs much better than GPM. In fact, the projection of the gradient has a large component on the second joint, but the speed limit of this joint is soon reached, and therefore GPM performs rather poorly in this case, being unable to increase the distance while the plane is moving in the positive x direction. Actually, there exist null space directions which could guarantee a better performance than the projection of the gradient, but GPM is "unaware" of the fact that the joint bounds may deteriorate the performances along the gradient projection direction and cannot explore those directions. NSBM, instead, exploits the knowledge of the joint bounds to choose the direction of the null space command, generating a motion of all other joints, which allows not to saturate the velocity limit on joint 2 while increasing successfully the distance between the center of the fourth reference frame and the plane.

IV. Real-time implementation

To implement the outlined algorithms on the real robotic system, the joint velocity interface, provided by the developers of Franka, was used. This interface allows the user to provide the commanded velocities in the joint space through a custom C++ controller library.

In addition, it is needed to meet the real-time constraints of the robot, i.e., the sampling time is 1 ms, so, at each iteration, all the computations needed to provide the joint velocity command must be executed within this period. The tests were performed on a PC with a 6-core Intel i7-5820K CPU, running at 3.3 GHz.

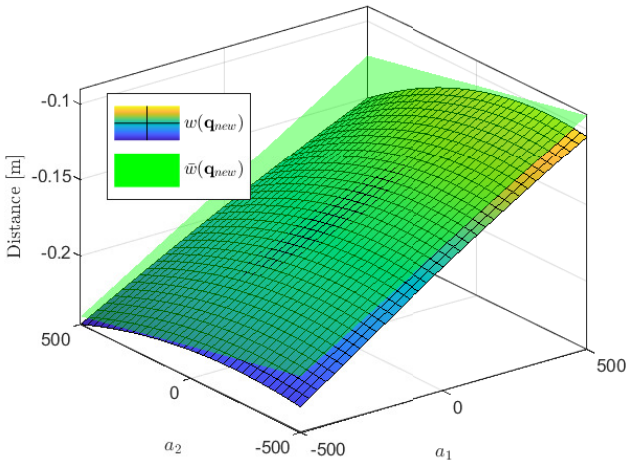

Fig. 6. $w(\mathbf{q}_{new})$ and $\bar{w}(\mathbf{q}_{new})$ as a function of $\mathbf{a}$, at time $t = 0.5$ s.

The GPM algorithm is intrinsically a linear optimization problem, because both the function (15a) to be minimized and the constraints (15b)-(15e) are linear in the decision variable k. As a result, using the linear programming package of the ALGLIB library [24], the problem was solved obtaining the same solutions as in MATLAB and meeting the requirements.

As for the NSBM algorithm, the function (19) to be minimized is non-linear in $\mathbf{a}$. However, it can be observed that, in the proximity of the origin of $\mathbb{R}^2$, which is the space containing all the possible choices for $\mathbf{a}$, the function can be approximated with a linear function, namely a plane.

So (19) is substituted with its linear approximation

$$\bar{w}(a_1, a_2) = w(0,0) + \frac{\partial w(0,0)}{\partial a_1} a_1 + \frac{\partial w(0,0)}{\partial a_2} a_2 \quad (20)$$

where a_1 and a_2 are the components of $\mathbf{a}$ and the distance function was written only as a function of $\mathbf{a}$ because $\mathbf{q}_{previous}$ and $\mathbf{v}_e$ are known and $\mathbf{J}$ and $\mathbf{B}$ of (17) are functions of $\mathbf{q}_{previous}$, so all these quantities can be considered as constants at each iteration.

Fig. 6 shows the behavior of the distance function and of its linear approximation at time $t = 0.5$ s. It can be observed that for large values of a_1 and a_2 the approximation error is noticeable, but in the proximity of the origin the planar approximation is very accurate. More specifically, considering the whole time history of the case study, the maximum percentage approximation error is 24.53% considering values of a_1 and a_2 ranging from -500 to 500, but decreases to $2.88 \cdot 10^{-4}\%$ considering values ranging from -5 to 5. So, since in the application under consideration the values of the components of $\mathbf{a}$ stay in the contour of the origin (their absolute values are always lower than 5 for both components), the linear approximation can safely be used. A further proof of the appropriateness of the approximation is given by the percentage error on the velocity command computed with the linearization with respect to its non-linearized version: this error takes a maximum value of $7.05 \cdot 10^{-3}\%$.

As a result, also NSBM was computed in real-time using the ALGLIB linear optimization solver, thus meeting the real-time requirements and reducing the computational cost. It should also be mentioned that the calculation of a null space basis of a matrix is computationally not problematic, and can be performed by several linear algebra libraries without raising real-time issues (in the tests reported here the Armadillo library was used [25]).

The overall computation time was always lower than 0.5 ms for both algorithms, therefore no errors due to the violation of the 1 ms time constraint happened during the tests.

V. Conclusion

This paper presented a framework for the optimal gain selection of GPM and, most importantly, the NSBM algorithm for the computation of the null space command in redundant robots, whose performance is always better or at least equal with respect to the commonly used GPM, and whose computational cost is not significantly higher than that of GPM and is compatible with the real-time constraints of most robots. Moreover, a case study was shown in which NSBM is used to solve the inverse kinematics of the Franka Emika Panda robot so as to perform obstacle avoidance in a locally optimal way, without interfering with the end effector task.

Existing techniques available in the literature which use a linear combination of the elements of the null space basis either require long computational times because they were developed for a priori path planning ([15] and [16]) or are not suitable for usage with a general form of the cost function and require to choose a desired operational space "escape velocity" for obstacle avoidance ([14]). Instead, NSBM uses as objective function the value of a desired criterion at the next iteration for a certain choice of the decision variables, and can therefore easily be applied to any case, such as joint limits or singularity avoidance.

Finally, unlike NSBM, the approaches that treat obstacle avoidance using constraints ([8]-[13]), do not necessarily provide the solution that causes the greatest possible decrease of the cost function.

Future developments of this work will involve implementing obstacle avoidance in a more complete and accurate formulation, applying the NSBM approach to fulfill other objectives, such as singularity avoidance, verifying that the described linearization is accurate enough for any type of movement and cost function and increasing the time horizon of the solution by introducing an optimal control problem instead of a local optimization problem.

References

[1] A. Liegeois, "Automatic supervisory control of the configuration and behavior of multibody mechanisms," *IEEE Trans. Sys., Man and Cyber.*, vol. 7, no. 12, pp. 868–871, 1977.

[2] H. Zghal, R. V. Dubey, and J. A. Euler, "Efficient gradient projection optimization for manipulators with multiple degrees of redundancy," in *Proc. IEEE Int. Conf. Rob. and Autom.* IEEE, 1990, pp. 1006–1011.

[3] A. A. Maciejewski and C. A. Klein, "Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments," *Int. J. Rob. Res.*, vol. 4, no. 3, pp. 109–117, 1985.

[4] S. I. Choi and B. K. Kim, "Obstacle avoidance control for redundant manipulators using collidability measure," *Robotica*, vol. 18, no. 2, pp. 143–151, 2000.

[5] L. Zhu, H. Mao, X. Luo, and J. Xiao, "Determining null-space motion to satisfy both task constraints and obstacle avoidance," in *IEEE Int. Sym. Assembly and Manufact.* IEEE, 2016, pp. 112–119.

[6] T. Yoshikawa, "Manipulability of robotic mechanisms," *Int. J. Rob. Res.*, vol. 4, no. 2, pp. 3–9, 1985.

[7] C. A. Klein and B. E. Blaho, "Dexterity measures for the design and control of kinematically redundant manipulators," *Int. J. Rob. Res.*, vol. 6, no. 2, pp. 72–83, 1987.

[8] B. Faverjon and P. Tournassoud, "A local based approach for path planning of manipulators with a high number of degrees of freedom," in *Proceedings. 1987 IEEE international conference on robotics and automation*, vol. 4. IEEE, 1987, pp. 1152–1159.

[9] D. Guo and Y. Zhang, "Acceleration-level inequality-based man scheme for obstacle avoidance of redundant robot manipulators," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 12, pp. 6903–6914, 2014.

[10] O. Kanoun, F. Lamiraux, and P.-B. Wieber, "Kinematic control of redundant manipulators: Generalizing the task-priority framework to inequality task," *IEEE Transactions on Robotics*, vol. 27, no. 4, pp. 785–792, 2011.

[11] L. Sciavicco and B. Siciliano, "A solution algorithm to the inverse kinematic problem for redundant manipulators," *IEEE J. Rob. and Autom.*, vol. 4, no. 4, pp. 403–410, 1988.

[12] F. G. Pin and F. A. Tulloch, "Resolving kinematic redundancy with constraints using the FSP (full space parameterization) approach," in *Proc. of IEEE Int. Conf. Rob. and Autom.*, vol. 1. IEEE, 1996, pp. 468–473.

[13] Z. Kemeny, "Redundancy resolution in robots using parameterization through null space," *IEEE Trans. on Ind. Electronics*, vol. 50, no. 4, pp. 777–783, 2003.

[14] R. P. Podhorodeski, A. A. Goldenberg, and R. G. Fenton, "Resolving redundant manipulator joint rates and identifying special arm configurations using Jacobian null-space bases," *IEEE Trans. Rob and Autom.*, vol. 7, no. 5, pp. 607–618, 1991.

[15] A. Reiter, H. Gattringer, and A. Müller, "Redundancy resolution in minimum-time path tracking of robotic manipulators," in *Int. Conf. Informatics in Control, Autom. and Rob.*, vol. 3. SCITEPRESS, 2016, pp. 61–68.

[16] A. Reiter, A. Müller, and H. Gattringer, "On higher order inverse kinematics methods in time-optimal trajectory planning for kinematically redundant manipulators," *IEEE Trans. Ind. Informatics*, vol. 14, no. 4, pp. 1681–1690, 2018.

[17] A. Del Prete, F. Nori, G. Metta, and L. Natale, "Prioritized motion–force control of constrained fully-actuated robots:"task space inverse dynamics"," *Robotics and Autonomous Systems*, vol. 63, pp. 150–157, 2015.

[18] R. V. Dubey, J. A. Euler, and S. M. Babcock, "An efficient gradient projection optimization scheme for a seven-degree-of-freedom redundant robot with spherical wrist," in *Proc. IEEE Int. Conf. Rob. and Autom.* IEEE, 1988, pp. 28–36.

[19] J. A. Euler, R. V. Dubey, S. M. Babcock, and W. R. Hamel, "A comparison of two real-time control schemes for redundant manipulators with bounded joint velocities," in *Proc. IEEE Int. Conf. Rob. and Autom.* IEEE, 1989, pp. 106–112.

[20] F. Flacco, A. De Luca, and O. Khatib, "Control of redundant robots under hard joint constraints: Saturation in the null space," *IEEE Trans. on Rob.*, vol. 31, no. 3, pp. 637–654, 2015.

[21] T. F. Chan and R. V. Dubey, "A weighted least-norm solution based scheme for avoiding joint limits for redundant joint manipulators," *IEEE Trans. Rob. and Autom.*, vol. 11, no. 2, pp. 286–292, 1995.

[22] Franka Control Interface Documentation. (2022). [Online]. Available: https://frankaemika.github.io/docs/

[23] J. J. Craig, *Introduction to robotics: mechanics and control*. Pearson Educacion, 2005.

[24] S. Bochkanov. ALGLIB. (2022). [Online]. Available: https://www.alglib.net/

[25] C. Sanderson and R. Curtin, "Armadillo: a template-based C++ library for linear algebra," *J. Open Source Soft.*, vol. 1, no. 2, p. 26, 2016.