A Redundancy-resolution Algorithm for Five-degree-of-freedom Tasks via Sequential Quadratic Programming

J. Léger* McGill University Montreal, Canada

J. Angeles† McGill University Montreal, Canada

Abstract— *Five-degree-of-freedom (five-dof) tasks are of particular interest in industry, since machining, arc-welding and deburring operations all fall into this category. These tasks, normally conducted with industrial six-dof robots, render the robot functionally redundant. Upon exploiting this redundancy to minimize the condition number of the Jacobian matrix, it is expected that the accuracy of the performed task will be increased. Traditional methods for redundancy-resolution are normally used to solve the more popular intrinsic redundancy; however, they cannot be used to resolve the functional redundancy. Five-dof tasks are formulated using an approach that leads to a system of six velocity-level kinematics relations in six unknowns, with a* 6×6 *Jacobian matrix, of nullity 1, which reflects the functional redundancy of the problem at hand. To resolve the foregoing redundancy, a method based on sequential quadratic programming (SQP) is proposed. An efficient method to compute the gradient of the condition number is also discussed as it is a key element for finding the posture of minimum condition number using a gradient method. An example then shows how the SQP algorithm can be applied to offline robot trajectory-planning for five-dof tasks.*

Keywords: sequential quadratic programming, redundancy-resolution, functional redundancy, trajectory planning, condition number

I. Introduction

The growing popularity of robots for manufacturing operations has brought about the need of robots with higher accuracy than what is currently available. The need for accurate robots, however, has not only been reported for manufacturing tasks. Recently, a robot for surgical tasks was proposed [1]. For high-accuracy robots, not only the components of the robot must be precisely built and assembled, but the path-planing and control algorithms must also be taken into consideration to improve accuracy. In some instances, the robot might be redundant, in which case more *degrees of freedom* (dof) are available than needed; therefore, a secondary task can be accomplished. This is the case in arc-welding and machining operations involving an axisymmetric tool [2–5].

Two types of redundancy can be identified, *functional redundancy* and *intrinsic redundancy*. To properly define these two, it is worthwhile to define three spaces: the *joint space* $\mathcal{J}$ is the space of joint variables; the *operational space* $\mathcal{O}$ is the reachable Cartesian space of the end-effector; and the *task space* $\mathcal{T}$ is the Cartesian space of the task. For the robot to be able to accomplish a task, the relations below should be observed:

$$\mathcal{T} \subseteq \mathcal{O} \tag{1}$$

$$\dim(\mathcal{T}) \leq \dim(\mathcal{O}) \leq \dim(\mathcal{J}) \tag{2}$$

Serial robots that have a joint-space dimension greater than their operational-space dimension are termed intrinsically redundant, their degree of intrinsic redundancy being computed as

$$r_i = \dim(\mathcal{J}) - \dim(\mathcal{O}) \tag{3}$$

On the other hand, *functionally redundant robots* have an operational-space dimension greater than their task-space dimension; their degree of functional redundancy is then computed as

$$r_f = \dim(\mathcal{O}) - \dim(\mathcal{T}) \tag{4}$$

the total degree of redundancy thus being

$$r_t = r_i + r_f \tag{5}$$

Intrinsic redundancy has been discussed extensively in the literature [6, 7]. Redundancy-resolution algorithms are generally based on the generalized inverse of the rectangular Jacobian matrix using the gradient-projection method (GPM), first introduced by Liégeois [8]. Crucial to the GPM, the Jacobian matrix must be of $m \times n$, with $m > n$, to be able to exploit the Jacobian matrix null space. This is, however, not the case for all types of redundancy, as pointed out by Sciavicco and Siciliano [9].

Functional redundancy, on the other hand, can yield a non-singular square Jacobian matrix. This is the case of machining and welding robots. To solve the functional redundancy, Baron proposed to insert a virtual joint [10], thus adding a column to the Jacobian. Using the modified Jacobian, the GPM can then be used to resolve the redundancy.

*jeremie.leger@mail.mcgill.ca
†angeles@cim.mcgill.ca

A more geometrically intuitive method, termed the *twist decomposition algorithm* (TWA) [3], makes use of projection matrices in the operational space to find the null space of the problem. TWA was shown to be faster than other methods for functional redundancy-resolution [11]. Here a redundancy-resolution scheme using sequential quadratic programming (SQP) [12] via the orthogonal decomposition algorithm (ODA) [13] is proposed.

SQP has been shown to work as a redundancy-resolution algorithm, but only for intrinsically redundant robots [14]. The novelty of this work lies in applying SQP to the functionally redundant robot for which the null space of the robot Jacobian is empty.

As a secondary objective, performance criteria have been developed to: avoid obstacles [15]; avoid joint limits [10,16]; minimize joint velocities and joint torques [17]; increase power transmission [4]; avoid singularities [10, 18]; or even a combination of multiple criteria [2, 10, 19]. In this work, the focus is on avoiding singularities.

In singularity avoidance, the two most popular performance criteria are manipulability [20] and condition number [18]. Manipulability suffers from not being able to measure distance form singularity, but rather only being able to identify when the robot is in a singular posture, i.e., when its manipulability is null. The condition number, on the other hand, is a measure of distance from singularity [21] and will be used as a performance index.

The condition number depends on the norm chosen; however, its significance is independent of the chosen norm. Since SQP requires the objective function to be twice continuous differentiable, the Frobenius norm is preferred, as it is an analytical function of its argument. This condition number guarantees the continuity of the second derivative. The Frobenius-norm condition number of the normalized Jacobian matrix and the Frobenius norm are, respectively, defined as

$$\kappa_F(\mathbf{J}) = ||\mathbf{J}_n||_F||\mathbf{J}_n^{-1}||_F \tag{6a}$$

$$||\mathbf{J}_n||_F = \sqrt{\frac{\mathrm{tr}(\mathbf{J}_\mathrm{n}\mathbf{J}_\mathrm{n}^\mathrm{T})}{n}} \tag{6b}$$

To calculate the condition number, a normalized Jacobian matrix must be used in order to prevent comparing numbers with units of length to dimensionless numbers. The characteristic length, introduced by Angeles [22] and defined as the length that minimizes the condition number, is one way of normalizing the Jacobian matrix. The normalized Jacobian matrix using the characteristic length L is defined as

$$\mathbf{J}_n = \begin{bmatrix} L\mathbf{A} \\ \mathbf{B} \end{bmatrix} \tag{7}$$

where **A** is the sub-matrix that maps joint velocities into angular velocities and **B** is the sub-matrix that maps joint velocities into the velocity of the end-effector (EE) operation point (OP). Gosselin [23] proposed a different method, by redefining the Jacobian matrix using only point velocities. In this method, multiple points of the end effector are used to fully define the motion, instead of the more traditional method of using the velocity of one point of the end-effector and the angular velocity of the same. The idea of using multiple points was then further investigated for parallel manipulators [24, 25]. In this method, caution must be used in the selection of the location of the points, as these locations could influence the condition number.

This work will, in a first section, investigate the SQP method via ODA for use in the functional redundancy-resolution. In the second section we discuss the normality conditions of the Frobenius-norm condition number. In the third section we show, by an example, how SQP compares to other functional-redundancy-resolution methods.

II. SQP Redundancy-resolution

A. SQP via ODA

The problem being solved with SQP takes the form:

$$f(\mathbf{x}) \rightarrow \min_{\mathbf{x}}, \quad \text{s.t.} \quad \mathbf{h}(\mathbf{x}) = 0 \tag{8}$$

where $f(\mathbf{x})$ is the objective function to be minimized, **h** is a vector of constraints and **x** is a set of design variables. In SQP, the function $f(\mathbf{x})$ to minimize is approximated at each iteration by a quadratic function, which leeds to the problem:

$$\begin{aligned} f(\mathbf{x}^k + \Delta\mathbf{x}^k) \sim f(\mathbf{x}^k) + \nabla\mathbf{f}_k^T\Delta\mathbf{x}^k \\ + \frac{1}{2}(\Delta\mathbf{x}^k)^T(\nabla\nabla f)_k\Delta\mathbf{x}^k \rightarrow \min_{\Delta\mathbf{x}^k} \end{aligned} \tag{9}$$

Similar to the ODA [13], the $\Delta\mathbf{x}^k$ step vector is decomposed into two orthogonal components.

$$\Delta\mathbf{x}^k = \Delta\mathbf{v}^k + \mathbf{L}_k\Delta\mathbf{u}^k \tag{10}$$

where $\mathbf{L}_k$ is an orthogonal complement of the Jacobian of $\mathbf{h}^k$.

An improvement $\Delta\mathbf{v}^k$ in satifying the constraint is computed as the minimum-norm solution of an underdetermined system, namely,

$$\mathbf{H}_k\Delta\mathbf{v}^k = -\mathbf{h}^k \tag{11}$$

where $\mathbf{H}_k$ is the Jacobian of **h** at the k^{th} iteration and $\mathbf{h}^k$ is the constraint vector at this same iteration. With the minimum-norm solution $\Delta\mathbf{v}^k$ of eq. (11), the optimization problem is then rewritten in terms of the design vector $\Delta\mathbf{u}^k$. Upon approximating the objective function up to second order terms, a solution for $\Delta\mathbf{u}^k$ is found as

$$\begin{aligned} \Delta\mathbf{u}^k = -(\mathbf{L}_k^T(\nabla\nabla f)_k\mathbf{L}_k)^{-1}\mathbf{L}_k^T \\ ((\nabla\nabla f)_k\Delta\mathbf{v}^k + (\nabla f)_k) \end{aligned} \tag{12}$$

or, by using approximation $\mathbf{B}_k$ of the Hessian:

$$\Delta\mathbf{u}^k = -(\mathbf{L}_k^T\mathbf{B}_k\mathbf{L}_k)^{-1}\mathbf{L}_k^T(\mathbf{B}_k\Delta\mathbf{v}^k + (\nabla f)_k) \tag{13}$$

Different approximations to the Hessian have been proposed in the literature [12, 26]. In this work the Broydon-Fletcher-Goldfarb-Shanno (BFGS) approximation of the Hessian is used [26]. Accordingly, for a given approximation $\mathbf{B}_k$, an update $\mathbf{B}_{k+1}$ is computed as

$$\mathbf{B}_{k+1} = \mathbf{B}_k - \frac{\mathbf{B}_k\mathbf{s}_k\mathbf{s}_k^T\mathbf{B}_k}{\mathbf{s}_k^T\mathbf{B}_k\mathbf{s}_k} + \frac{\mathbf{y}_k\mathbf{y}_k^T}{\mathbf{s}_k^T\mathbf{y}_k} \tag{14}$$

where $\mathbf{B}_0$ is defined as the $n \times n$ identity matrix, while

$$\mathbf{y}_k = \nabla f(\mathbf{x}^{k+1}) - \nabla f(\mathbf{x}^k) \tag{15}$$

and

$$\mathbf{s}_k = \mathbf{x}^{k+1} - \mathbf{x}^k \tag{16}$$

B. SQP Functional Redundancy-resolution

The five-dof constrained problem can be formulated, as suggested by Angeles [27], by means of two points separated by a distance d. Choosing this approach, a rotation about the axis that passes trough these two points does not change the location of these two points, this rotation being the functional redundancy. The constraint of the five-dof problem is then defined as

$$\begin{bmatrix} \mathbf{p}_1 - \mathbf{a}_1 \\ \mathbf{p}_2 - \mathbf{a}_2 \end{bmatrix} = \mathbf{0} \tag{17}$$

where $\mathbf{p}_1$ and $\mathbf{p}_2$ are the current position vectors of points 1 and 2, respectively, while $\mathbf{a}_1$ and $\mathbf{a}_2$ are their desired position vectors. When the unit vector $\mathbf{e}_{\rm act}$ is parallel to the axis on which points $\mathbf{p}_1$ and $\mathbf{p}_2$ lie, $\mathbf{p}_2$ can be found as

$$\mathbf{p}_2 = \mathbf{p}_1 + d\mathbf{e}_{\rm act} \tag{18}$$

A similar relation is found for the desired points with $\mathbf{e}_{\rm des}$ denoting the vector parallel to the desired direction of the axis. Using the absolute position of point 1 and the relative position of point 2 with respect to point 1 leads to a second form of the constraints, namely,

$$\mathbf{h} = \begin{bmatrix} d\mathbf{e}_{act} - d\mathbf{e}_{des} \\ \mathbf{p}_1 - \mathbf{a}_1 \end{bmatrix} = \mathbf{0} \tag{19}$$

the first three components representing the error in the relative position. The Jacobian of the constraint of eq. (19) is

$$\mathbf{H} = \frac{d\mathbf{h}}{d\mathbf{x}} = \begin{bmatrix} \mathbf{B}_2 - \mathbf{B}_1 \\ \mathbf{B}_1 \end{bmatrix} \tag{20}$$

where $\mathbf{B}_1$ and $\mathbf{B}_2$ are the lower blocks of the robot Jacobian, as defined in eq. (7). To gain insight into the first three rows of $\mathbf{H}$, a second form of the result is given here

$$\begin{aligned} \frac{d(\mathbf{p}_2 - \mathbf{p}_1)}{dt} &= (\mathbf{B}_2 - \mathbf{B}_1)\dot{\boldsymbol{\theta}} \\ &= d\frac{d\mathbf{e}_{\rm act}}{dt} \\ &= \boldsymbol{\omega} \times d\mathbf{e}_{\rm act} \\ &= -d\,\mathrm{CPM}(\mathbf{e}_{\rm act})\mathbf{A}\dot{\boldsymbol{\theta}} \end{aligned} \tag{21}$$

where $\mathrm{CPM}(\cdot)$ is the cross product matrix of $(\cdot)$ [28], $\boldsymbol{\omega}$ the angular velocity and $\mathbf{A}$ the upper block of the robot Jacobian. This leads to an alternative form of the Jacobian:

$$\mathbf{H} = \begin{bmatrix} -\mathrm{CPM}(\mathbf{e}_{act}) & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{1}_{3\times3} \end{bmatrix} \begin{bmatrix} d\mathbf{A} \\ \mathbf{B}_1 \end{bmatrix} \tag{22}$$

whose first factor is singular, given that its upper-left block is a 3×3 skew-symmetric matrix, and hence, of rank equal to 2. The significance of the distance d, separating the two points, discussed below, cannot be neglected. Too small or too big a length will render the constraint Jacobian poorly scaled. In fact, the problem of choosing the proper length is very similar to the one of choosing the characteristic length, as the condition number of the constraint Jacobian depends on it in the same way. For this reason, it is recommended to use the characteristic length as the distance between the two points. The Jacobian expressed as a function of the robot normalized Jacobian matrix is

$$\mathbf{H} = \begin{bmatrix} -\mathrm{CPM}(\mathbf{e}_{act}) & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{1}_{3\times3} \end{bmatrix} \mathbf{J}_n \tag{23}$$

From eq. (23), the orthogonal complement $\mathbf{L}$ of $\mathbf{H}$, whose columns span the null space of the latter, is found as

$$\mathbf{L} = \mathbf{J}_n^{-1} \begin{bmatrix} \mathbf{e}_{\rm des} \\ \mathbf{0} \end{bmatrix} \tag{24}$$

The other possible directions of the null space would arise from the null space of the robot Jacobian itself, for robots with more than six joints. This is, however, not the focus of this work, as we have assumed that the robot in question has as many joints as its operational-space dimension (only functional redundancy is considered).

Writting eq. (11) for this particular case, the linear system to be solved for $\Delta\mathbf{v}$ is

$$\mathbf{H}\Delta\mathbf{v} = \begin{bmatrix} -\mathrm{CPM}(\mathbf{e}_{act}) & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{1}_{3\times3} \end{bmatrix} \mathbf{J}_n\Delta\mathbf{v} = \mathbf{h} \tag{25}$$

Note that $\mathbf{h}$ does not necessarily lie in the range of $\mathbf{H}$; however, the directions that lie outside of the range are known and well defined in operational-space. In fact, they are the redundant directions of the robot. One way to make sure that the right-hand side of eq. (25) lies in the range of $\mathbf{H}$ consists in multiplying both sides of the equation by a singular matrix to yield a projection matrix $\mathbf{P}$ on the left, with

its singular directions in the same directions of those of **H**. This eliminates the components of **h** that do not lie in the range of **H**:

$$\mathbf{P}\mathbf{J}_n\Delta\mathbf{v} = \begin{bmatrix} \mathrm{CPM}(\mathbf{e}_{act}) & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{1}_{3\times3} \end{bmatrix}\mathbf{h} \tag{26}$$

where **P** is the matrix that projects a vector onto the range of $\mathbf{J}_p$, namely

$$\mathbf{P} = \begin{bmatrix} -\mathrm{CPM}^2(\mathbf{e}_{act}) & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{1}_{3\times3} \end{bmatrix} \tag{27}$$

A solution $\Delta\mathbf{v}$ can now be found as

$$\Delta\mathbf{v} = \mathbf{J}_n^{-1}\mathbf{t}_e \tag{28}$$

where

$$\mathbf{t}_e = \begin{bmatrix} -\mathrm{CPM}(\mathbf{e}_{act}) & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{1}_{3\times3} \end{bmatrix}\mathbf{h} \tag{29}$$

the projection matrix **P** having no effect on the solution of the system, since the right-hand side of eq. (26) lies entirely in the range of **P**. If matrix **P** has no effect in solving eq. (26) and eq. (28) can be used, then $\mathbf{t}_e$ can be expressed in the form of a normalized twist. This is shown by expanding **h** and computing the product.

$$\begin{aligned}\mathbf{t}_e &= \begin{bmatrix} -\mathbf{CPM}(\mathbf{e}_{act}) & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{1}_{3\times3} \end{bmatrix}\begin{bmatrix} d\mathbf{e}_{act} - d\mathbf{e}_{des} \\ \mathbf{p}_1 - \mathbf{a}_1 \end{bmatrix} \\ &= \begin{bmatrix} d(\mathbf{e}_{act} \times \mathbf{e}_{des}) \\ \mathbf{p}_1 - \mathbf{a}_1 \end{bmatrix}\end{aligned} \tag{30}$$

which bears the form of a normalized twist, the first three components of $\mathbf{t}_e$ providing a measure of the error in the orientation of the end-effector. The product $\mathbf{e}_{act} \times \mathbf{e}_{des}$ gives the direction of the axis about which the end-effector should rotate, its magnitude being the sine of the angle that separates the two axes.

A potential pitfall in using the sine of the angle is its inability to distinguish angles in the first quadrant from angles in the second quadrant. A more intuitive way of choosing the magnitude is via the angle between the two axes directly. In fact, when the angle is small, both methods should perform equally well, since $\sin(\theta) \sim \theta$ in this case. The angle itself is chosen here, instead of its sine, as the angle gives the constraint error a more direct physical meaning.

In the unlikely case that both axes are collinear but in opposite directions, the algorithm would also fail to recognize that the orientation error is at its maximum since the cross product would vanish. A simple means to avoid this problem is by introducing additionally the cosine of the angle. When this situation occurs, any axis normal to $\mathbf{e}_{\mathrm{des}}$ can be used to define the error.

Having solved for the constraint, the performance index can then be minimised using eq. (13). Finally, the step $\Delta\mathbf{x}$ at each iteration is given by eq. (10).

III. Condition Number Normality Conditions

The normality conditions are of the utmost importance in the problem of finding the minimum condition number using a gradient method. The square of the Frobenius-norm condition number is recalled here as it is easer to work with than the condition number itself.

$$\kappa(\mathbf{J}_n)_F^2 = \frac{1}{n^2}\mathrm{tr}(\mathbf{J}_n^T\mathbf{J}_n)\mathrm{tr}(\mathbf{J}_n^{-1}\mathbf{J}_n^{-T}) \tag{31}$$

Differentiating the square of the condition number with respect to an arbitrary variable, the following expression is found:

$$\begin{aligned}\frac{\partial\kappa^2}{\partial x_i} &= \frac{1}{n^2}\frac{\partial\mathrm{tr}(\mathbf{J}_n^T\mathbf{J}_n)}{\partial x_i}\mathrm{tr}(\mathbf{J}_n^{-1}\mathbf{J}_n^{-T}) \\ &\quad + \mathrm{tr}(\mathbf{J}_n^T\mathbf{J}_n)\frac{\partial\mathrm{tr}(\mathbf{J}_n^{-1}\mathbf{J}_n^{-T})}{\partial x_i} \\ &= \frac{1}{n^2}\left[\mathrm{tr}\left(\frac{\partial\mathbf{J}_n^T}{\partial x_i}\mathbf{J}_n\right)\right. \\ &\quad \left.+\mathrm{tr}\left(\mathbf{J}_n^T\frac{\partial\mathbf{J}_n}{\partial x_i}\right)\right]\mathrm{tr}(\mathbf{J}_n^{-1}\mathbf{J}_n^{-T}) \\ &\quad - \mathrm{tr}(\mathbf{J}_n^T\mathbf{J}_n)\left[\mathrm{tr}\left(\mathbf{J}_n^{-1}\frac{\partial\mathbf{J}_n}{\partial x_i}\mathbf{J}_n^{-1}\mathbf{J}_n^{-T}\right)\right. \\ &\quad \left.+\mathrm{tr}\left(\mathbf{J}_n^{-1}\mathbf{J}_n^{-T}\frac{\partial\mathbf{J}_n^T}{\partial x_i}\mathbf{J}_n^{-T}\right)\right] \\ &= \frac{2}{n^2}\mathrm{tr}\left(\frac{\partial\mathbf{J}_n^T}{\partial x_i}\mathbf{J}_n\right)\mathrm{tr}(\mathbf{J}_n^{-T}\mathbf{J}_n^{-1}) \\ &\quad - 2\mathrm{tr}(\mathbf{J}_n^T\mathbf{J}_n)\mathrm{tr}\left(\mathbf{J}_n^{-1}\mathbf{J}_n^{-T}\mathbf{J}_n^{-1}\frac{\partial\mathbf{J}_n}{\partial x_i}\right)\end{aligned} \tag{32}$$

where, depending on the chosen variable, x_i, $\partial\mathbf{J}_n/\partial x_i$ will change. For existing industrial robots, the variables of interest are generaly the joint angles and the characteristic length. It should be noted that the joint variable θ_1 does not influence the condition number [1]; therefore, the derivative of the condition number with respect to θ_1 is null. This is so because the Frobenius norm, and, by consequence, the Frobenius-norm condition number, is invariant to a change of frame. For the other joint angles, the normalized Jacobian matrix partial derivatives can be found as

$$\frac{\partial\mathbf{J}_n(\mathbf{j})}{\partial\theta_k} = \begin{cases} \begin{bmatrix} L\mathbf{e}_k \times \mathbf{e}_i \\ \mathbf{e}_k \times (\mathbf{e}_i \times \mathbf{r}_i) \end{bmatrix} & \text{if } i > k \\ \begin{bmatrix} \mathbf{0} \\ \mathbf{e}_i \times (\mathbf{e}_k \times \mathbf{r}_k) \end{bmatrix} & \text{if } i \leq k \end{cases} \tag{33}$$

where **j** denotes the i^{th} column of the Jacobian, $\mathbf{e}_j$ the vector parallel to the j^{th} axis, and $\mathbf{r}_j$ the vector from a point

[1] A change in θ_1 amounts to a rotation of the whole robot about its first axis, which amounts in turn to changing the viewpoint.

on the j^{th} axis to the OP. The partial derivative of the normalized Jacobian with respect to the characteristic length is

$$\frac{\partial \mathbf{J}_n}{\partial L} = \begin{bmatrix} \mathbf{e}_1 & \dots & \mathbf{e}_6 \\ \mathbf{0} & \dots & \mathbf{0} \end{bmatrix} \tag{34}$$

Using eqs.(32), (33) and (34), the condition number normality conditions are found.

IV. Implementation and results

To better understand how SQP can be used to minimize the condition number, an example is shown here. In this example, the FANUC 710ic-50 is to perform a milling task on a cylinder. The helicoidal trajectory on the cylinder, displayed in Fig. 1, is described by the parametrisation below:

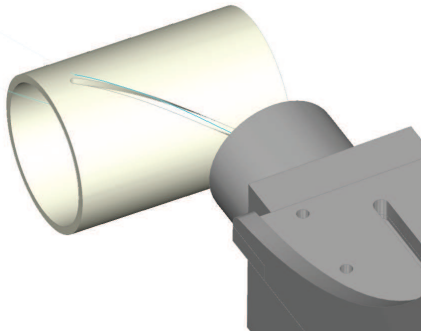

Fig. 1. Desired trajectory

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} R\cos(\Delta\phi - \phi) \\ p\phi \\ R\sin(\Delta\phi - \phi) \end{bmatrix} \tag{35}$$

where the coordinates (x, y, z), which represent the desired position of the tool, are given in a frame attached to the cylinder, and

$$\mathbf{e}_{\mathrm{n}} = \begin{bmatrix} -\cos(\Delta\phi - \phi) \\ 0 \\ -\sin(\Delta\phi - \phi) \end{bmatrix} \tag{36}$$

is a vector in the cylinder frame, normal to the cylinder wall, playing the role of the desired direction of the tool axis. Describing the pose of the cylinder in the robot-base frame by mean of a rotation matrix $\mathbf{R}_{cyl}$ and a position vector $\mathbf{p}_{cyl}$, the position vector of the operation point and the direction of the tool axis for the trajectory in robot-base frame are

$$\mathbf{p} = \mathbf{p}_{\mathrm{cyl}} + \mathbf{R}_{\mathrm{cyl}} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{37}$$

$$\mathbf{e}_{\mathrm{des}} = \mathbf{R}_{\mathrm{cyl}}\mathbf{e}_{\mathrm{n}} \tag{38}$$

For this example, the data in Table I were used to describe the trajectory.

radius of cylinder R (mm)	250
start/end angle ϕ (°)	0/90
offset angle $\Delta\phi$ (°)	180
Velocity $\dot{\phi}$ (°/s)	9
pitch of helix p (mm/rad)	$\frac{500}{\pi}$
$\mathbf{p}_{cyl}$ (mm)	$\begin{bmatrix} 1300\ 1000\ 0 \end{bmatrix}^T$
$\mathbf{R}_{cyl}$	**1**

TABLE I. Helicoidal trajectory data

The Denavit-Hartenberg parameters [29] of the FANUC 710ic-50 robot with the milling tool are shown in Table II. The operation point is considered to be at the tip of the milling tool for this particular robot.

joint i	a_i(mm)	b_i(mm)	α_i(°)
1	150	0	-90
2	870	0	180
3	170	0	-90
4	0	-1016	90
5	0	0	-90
6	-287.692	-607.777	120

TABLE II. Denavit-Hartenberg parameters of of the FANUC 710ic-50 with milling tool

The process of minimizing the condition number of the Jacobian matrix must be performed in two parts. In a first part, the characteristic length of the robot should be found by solving the unconstrained optimum posture problem for the robot in question. In a second part, the characteristic length is used to calculate de condition number and its gradient in solving the the trajectory constrained problem. It is in this second part that SQP can be utilized to resolve the redundancy.

A. Finding the Characteristic Length

To find the characteristic length of the FANUC 710ic-50, an unconstrained optimization problem must be solved:

$$\min_{\mathbf{x}} \kappa(\mathbf{J}_n) \tag{39}$$

where the design vector $\mathbf{x}$ is

$$\mathbf{x} = \begin{bmatrix} L & \theta_2 & \dots & \theta_6 \end{bmatrix}^T \tag{40}$$

Note that θ_1 is not included in $\mathbf{x}$, as it has no affect on the condition number. Using the quasi-Newton method with BFGS update of the Hessian matrix a solution can then be found. As a stopping criterion, the normalized-step size was used. The normalization was done by dividing the current estimate of the characteristic length by the root-mean-square value of the distances from each axis to the operation point. Table III summarizes the results of the optimization.

initial guess $\mathbf{x}_0$	$\begin{bmatrix} 20^\circ \\ -20^\circ \\ 0 \\ -90^\circ \\ 0 \\ 298.5933\ \text{mm} \end{bmatrix}$
optimum $\mathbf{x}$	$\begin{bmatrix} 0.4424^\circ \\ -35.7223^\circ \\ 0 \\ -118.5801^\circ \\ 0 \\ 485.5933\ \text{mm} \end{bmatrix}$
number of iterations	31
condition number	6.5046
stopping criteria (step size)	10^{-6}

TABLE III. Minimum condition number posture optimization results

B. Optimizing a Trajectory

The characteristic length now known, the optimal trajectory can be found, with initial guess

$$\mathbf{x}_0 = \begin{bmatrix} 20.6 \\ 22.6 \\ -12.8 \\ -33.8 \\ -79.8 \\ 263.0 \end{bmatrix} \tag{41}$$

which places the robot on the first trajectory point. The minimum condition number posture for the first point of the trajectory is found via SQP; for comparison the same is found via TWA. The redundancy-resolution for TWA yields

$$\Delta\mathbf{x} = \mathbf{J}^{-1}(\mathbf{1} - \mathbf{T})\mathbf{t}_e + \mathbf{J}^{-1}\mathbf{T}\mathbf{J}c\nabla\kappa \tag{42}$$

where $\mathbf{T}$, described below for this particular example, is a matrix that projects a twist onto a space orthogonal to the task space:

$$\mathbf{T} = \begin{bmatrix} \mathbf{e}_{des}\mathbf{e}_{des}^T & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{43}$$

The constant c in eq. (43) must be carefully chosen; for this example c was selected to be 5×10^{-2}. The results of the optimization using both SQP and TWA are shown below.

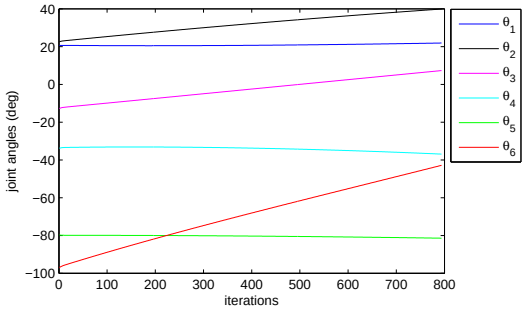

Fig. 2. Joint values progression for TWA

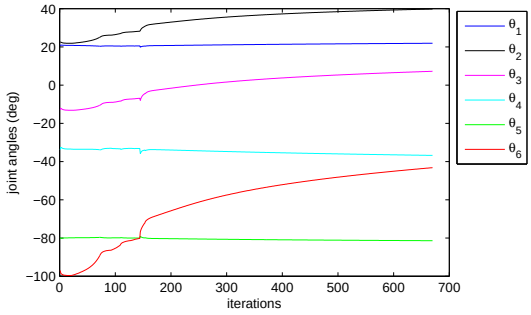

Fig. 3. Joint values progression for SQP

From Figs. 2 and 3 it is clear that both the proposed SQP algorithm and TWA converge to the same solution, however SQP does this in less iterations. Table IV summarizes the results of the two optimization procedures.

	SQP	TWA
joint solution	$\begin{bmatrix} 21.89 \\ 39.81 \\ 7.22 \\ -36.79 \\ -81.43 \\ -43.21 \end{bmatrix}$	$\begin{bmatrix} 21.90 \\ 39.90 \\ 7.31 \\ -36.91 \\ -81.43 \\ -42.84 \end{bmatrix}$
total iterations	670	794

TABLE IV. Summary of results for the optimal first point

By comparing the results of the iterations, as shown in Fig. 4, it is apparent that SQP converges faster than TWA to the minimum. This was expected, since the SQP algorithm uses a second-order approximation of the condition number, whereas TWA uses only a first-order approximation. While the convergence of the latter is linear, that of the former is superlinear.

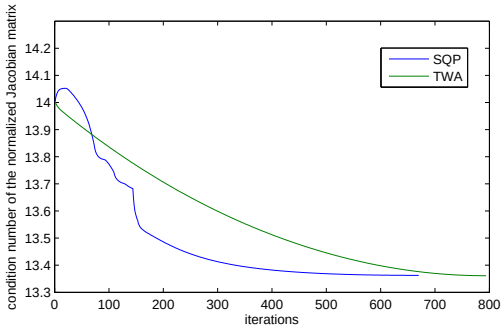

Fig. 4. Condition number progression for SQP and TWA

To confirm that the joint solution found by SQP and TWA is truly the minimum condition number posture of the robot, the inverse condition number is plotted below.

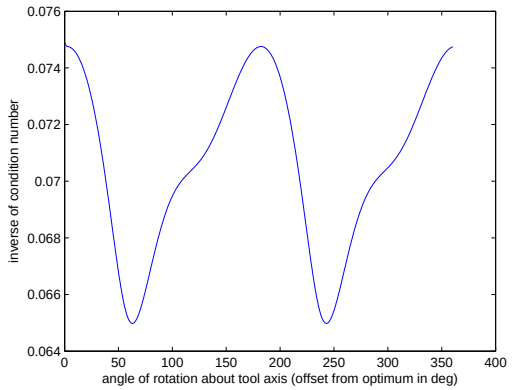

Fig. 5. Condition number as a function of the angle of rotation about the redundant axis

In Fig. 5, the abscissae represent the angle of rotation of the end-effector about the tool axis from the optimum posture. An angle of 0° or 360° thus corresponds to the solution found by SQP. The ordinates represent the inverse of the Frobenius-norm condition number. For this particular trajectory point, the end-effector can undergo a full rotation of 360° without passing trough a singularity and the condition number is π-periodic with respect to a rotation about the tool axis. From this figure, the solution found by SQP is apparently the one that maximizes the inverse condition number, and therefore, minimizes the condition number.

After having found the optimal first point of the joint trajectory, the remainder of the trajectory follows in a similar way using the previous trajectory point as an initial guess. The optimal joint trajectory along with the corresponding condition number are displayed in Figs. 6 and 7, respectively. A total of 100 trajectory points were used to describe the joint path.

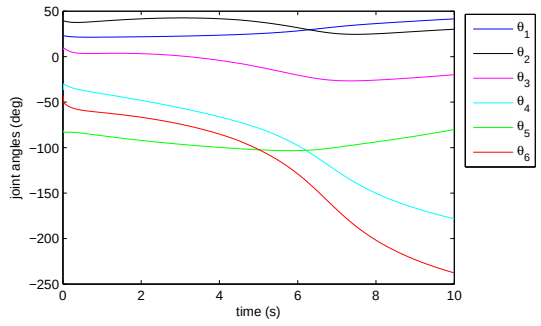

Fig. 6. Optimum trajectory

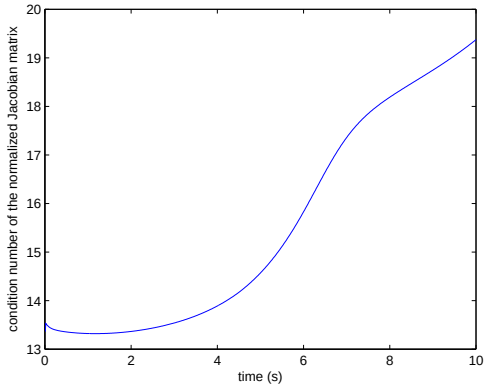

Fig. 7. Condition number of trajectory

The trajectory was then verified using RobotMaster in order to check for potential problems such as interferences and joint limits.

V. Conclusions

This work has demonstrated how SQP can be used as a redundancy-resolution algorithm for functionally redundant manipulators. By restating the constraints of the problem as a five-dimensional task, a null space of the constraint Jacobian could be found, which then allowed the use of SQP to resolve the redundancy. In an example, the method was compared to TWA and showed an improvement on the total number of iterations. In solving the example, it was also demonstrated how SQP could be utilized for offline trajectory planning of functionally redundant robots.

References

[1] Z. Li, D. Glozman, D. Milutinovic, and J. Rosen, "Maximizing dexterous workspace and optimal port placement of a multi-arm surgical robot," in *Proc. 2011 IEEE International Conference on Robotics and Automation (ICRA)*, (Shanghai, China), pp. 3394–3399, May 9-13 2011.

[2] W. Zhu, W. Qu, L. Cao, D. Yang, and Y. Ke, "An off-line programming system for robotic drilling in aerospace manufacturing," *The International Journal of Advanced Manufacturing Technology*, vol. 68, pp. 2535–2545, 2013.

[3] L. Huo and L. Baron, "Kinematic inversion of functionally redundant serial manipulators: application to arc-welding," *Trans. of the Canadian Society for Mech. Eng.*, vol. 29, pp. 679–690, 2005.

[4] S. H. H. Zargarbashi, W. Khan, and J. Angeles, "Posture optimization in robot-assisted machining operations," *Mechanism and Machine Theory*, vol. 51, pp. 74–86, 2012.

[5] S. H. H. Zargarbashi, W. Khan, and J. Angeles, "The jacobian condition number as a dexterity index in 6r machining robots," *Robotics and Computer Integrated Manufacturing*, vol. 28, pp. 694–699, 2012.

[6] N. Arenson, J. Angeles, and L. Slutski, "Redundancy-resolution algorithms for isotropic robots," *Advances in Robot Kinematics: Analysis and Control*, pp. 425–434, 1998.

[7] T. Yoshikawa, "Basic optimization methods of redundant manipulators," *Laboratory robotics and automation*, vol. 8, pp. 49–60, 1996.

[8] A. Liégeois, "Automatic supervisory control of the configuration and behavior of multibody mechanisms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 12, pp. 868–871, 1977.

[9] L. Sciavicco and B. Siciliano, *Modelling and Control of Robot Manipulators*. London: Springer, 2000.

[10] L. Baron, "A joint-limits avoidance strategy for arc-welding robots," in *Proc. International Conference on Integrated Design and Manufacturing in Mechanical Engineering*, (Montreal, Canada), May 16-19 2000.

[11] L. Baron and L. Huo, "Inverse kinematics of functionally-redundant serial manipulators: A comparaison study," in *Proc. 12th IFToMM World Congress*, (Besançon, France), 2007.

[12] J. Nocedal and S. J. Wright, *Numerical Optimization*. New York: Springer, 1999.

[13] J. Angeles, K. Anderson, and C. Gosselin, "Constrained design optimization using orthogonal decomposition," *Mechanical Design*, vol. 112, pp. 255–256, 1990.

[14] J. Xu, W. Wang, and Y. Sun, "Two optimization algorithms for solving robotics inverse kinematics with redundancy," *Control Theory Applications*, vol. 8, pp. 166–175, 2010.

[15] S. Khadem and R. Dubey, "A global cartesian space obstacle avoidance sheme for redundant manipulators," *Optimal Control Applications and Methods*, vol. 12, pp. 279–286, 1991.

[16] J. Andres, L. Gracia, and T. Josep, "Implementation and testing of a cam postprocessor for an industrial redundant workcell with evaluation of several fuzzified redundancy resolution schemes," *Robotics and Computer-Integrated Manufacturing*, vol. 28, pp. 265–274, 2012.

[17] R. Dubey and J. Y. S. Luh, "Redundant robot control using task based performance measures," *Journal of Robotic Systems*, vol. 5, pp. 409–432, 1988.

[18] J. K. Salisbury and J. J. Craig, "Articulated hands: Force control and kinematic issues," *The International Journal of Robotics Research*, vol. 1, pp. 4–17, 1982.

[19] L. Huo and L. Baron, "The self-adaptation of weights for joint-limits and singularity avoidances of functionally redundant robotic-task," *Robotics and Computer-Integrated Manufacturing*, vol. 27, pp. 367–376, 2011.

[20] T. Yoshikawa, "Manipulability of robotic mechanisms," *The International Journal of Robotics Research*, vol. 4, pp. 3–9, 1985.

[21] D. S. Watkins, *Fundamentals of Matrix Computations*. New York: Wiley, 2010.

[22] J. Angeles, "The design of isotropic manipulator architectures in the presence of redundancies," *The International Journal of Robotics Research*, vol. 11, pp. 196–201, 1992.

[23] C. M. Gosselin, "The optimum design of robotic manipulators using dexterity indices," *Robotics and Autonomous Systems*, vol. 9, pp. 213–226, 1992.

[24] G. Pond and J. Carretero, "Formulating jacobian matrices for the dexterity analysis of parallel manipulators," *Mechanism and Machine Theory*, vol. 41, pp. 1505–1519, 2006.

[25] S.-G. Kim and J. Ryu, "New dimensionally homogeneous jacobian matrix formulation by three end-effector points for optimal design of parallel manipulators," *IEEE Transactions on Robotics and Automation*, vol. 19, 2003.

[26] H. Yabe, H. Ogasawara, and M. Yoshino, "Local and superlinear convergence of quasi-newton methods based on modified secant conditions," *Computational and Applied Mathematics*, vol. 205, pp. 617–632, 2007.

[27] J. Angeles, "Iterative kinematic inversion of general five-axis robot manipulators," *The International Journal of Robotics Research*, vol. 4, pp. 37–46, 1986.

[28] J. Angeles, *Fundamentals of Robotic Mechanical Systems. Theory, Methods, and Algorithms*. New York: Springer, 2007.

[29] R. S. Hartenberg and J. Denavit, *Kinematic Synthesis of Linkages*. New York: McGraw-Hill, 1964.