DOI: https://doi.org/10.1007/978-3-030-19648-6_33

Bounded Self-Motion of Functional Redundant Robots

Leon Žlajpah and Tadej Petrič

Jožef Stefan Institute
Ljubljana, Slovenia
{leon.zlajpah,tadej.petric}@ijs.si *

Abstract. In this paper, we consider a problem how to exploit a task space motion for lower-priority tasks when the end-effector motion allows some deviation of the motion for the primary task. Using a common redundancy resolution methods, the self-motion is only possible in the null-space. Therefore, we propose a novel combination of controllers in two spaces: in the task space and in the reduced task space, where DOFs corresponding to spatial directions allowing the deviations are excluded. The motion generated by the controller in the reduced task space is mapped into the main task and by properly selecting the controller parameters the resulting motion does not violate the constraints. To demonstrate the effectiveness of the proposed control we show simulation examples where motion in a constraint region is used to avoid joint limits.

Keywords: Redundant robots, functional redundancy, null-space control, tracking relaxation

1 Introduction

Robot tasks are usually defined as a motion of the robot's end-effector. Typically, the motion of the end-effector is restricted by different constraints arising from the task itself or the working environment. In general, to perform the task without violating any constraints the robot has to be redundant. In this case, there are infinite joint configurations which result in the desired task-space pose of the end-effector. The problem is how to find such joint motions that realize the desired end-effector motion while guaranteeing that no constraints are violated.

Typically, the redundancy resolution is done by selecting a suitable objective function and then a feasible solution is found by using a local or a global optimization method. In most cases, it is assumed that the robot is kinematically redundant. However, there are tasks like arc welding or spray painting, where not all six Cartesian dimensions are important to accomplish the task. These unused DOFs can then contribute to the degree-of-redundancy (DOR) of the

* This work was supported by EU Horizon 2020 RIA Programme grant 820767 CoLLaboratE, and by Slovenian Research Agency grant P2-0076.

robot and they can be exploited for the self-motion. This type of redundancy has been recognized as the *functional redundancy* [8, 2, 5]. To control robots with functional redundancy, Baron [2] proposed to add a virtual joint to the robot and used the usual redundancy resolution with the extended Jacobian. A more general version has been proposed in [7], where a robot has been virtually extended by three prismatic joints and three rotational joints at the end-effector allowing flexible redundancy control when the task is not fully constraint in the Cartesian space. Huo and Baron [5] proposed the orthogonal decomposition of the task-space without considering the null-space of the Jacobian matrix. In [9] the use of the extended Jacobian is proposed for tasks where axis-symmetric tools are used (like a spray-painting gun) and the orientation of the tool along one axis is not important. For 5 DOF tasks, a redundancy resolution algorithm based on sequential quadratic programming is proposed in [6] and based on convex optimization in [4].

These approaches do not consider the possibility that tasks may allow some freedom regarding the motion of controlled DOF, i.e. the task can be performed with a set of possible end-effector positions and/or orientations. For example, for spray-painting and welding tasks it is common to assume that the tool is axis-symmetric and hence, it has one additional rotational DOR. However, the task can be performed also when the orientation of the tool-axis is not exactly oriented but in some region around the object. In [3] a concept of Task Space Regions is proposed, where the idea is that the end-effector pose is not fixed but can belong to a region of admissible positions and/or orientations. These regions are then used for motion planning. In [1] a method for optimizing the trajectories considering allowed deviations around the path is presented.

All path planning algorithms require apriori knowledge of constraints. If this is not the case, a common approach is to perform redundancy resolution in the control. The simplest approach to solve the kinematic redundancy is to resolve it on the velocity level, where the differential kinematic equation is solved by using a kind of generalized inverse, and the projection of an arbitrary vector onto the null-space of the Jacobian is used to exploit the self-motion of the robot. When the motion in one spatial direction has to be controlled, but the task allows some deviation in this direction then we can treat this freedom-of-motion as *bounded functional redundancy*, i.e. this DOF belongs to the task space and allows simultaneously some motion which can be used to perform lower priority tasks.

This paper aims at contributing to this field by presenting efficient kinematic control strategies for bounded functional redundancy. The main goal is to improve the capabilities of robots using intrinsic and functional redundancy in the same framework. For that, we propose a control framework which allows exploiting freedom-of-motion in the task DOFs for lower priority tasks like joint limit avoidance or obstacle avoidance.

2 Modelling

We deal with robot manipulators, which can be represented by a serial kinematic chain. Let the configuration of the robot manipulator be represented by the n-dimensional vector $\boldsymbol{q}$ of joint positions. The robot's end-effector pose $\boldsymbol{x} \in SE(3)$, the six-dimensional Cartesian space, is represented by the position $\boldsymbol{p} \in \mathbb{R}^3$ and orientation $\mathcal{Q} \in SO(3)$ (quaternion representation), which can be expressed as a function of joint coordinates using the direct kinematic equations, $\boldsymbol{p} = \boldsymbol{p}(\boldsymbol{q})$ and $\mathcal{Q} = \mathcal{Q}(\boldsymbol{q})$.

In general, the spatial end-effector velocity $\boldsymbol{v}$ is expressed as

$$\boldsymbol{v} = [\dot{\boldsymbol{p}}^T \boldsymbol{\omega}^T]^T \ , \tag{1}$$

where $\dot{\boldsymbol{p}} \in \mathbb{R}^3$ and $\boldsymbol{\omega} \in \mathbb{R}^3$ are the linear and angular velocity of the end-effector, respectively. The relation between joint velocities $\dot{\boldsymbol{q}}$ and end-effector velocity $\boldsymbol{v}$ is represented by the differential forward kinematics in the form

$$\boldsymbol{v} = \begin{bmatrix} \dot{\boldsymbol{p}} \\ \boldsymbol{\omega} \end{bmatrix} = \mathbf{J}(\boldsymbol{q})\dot{\boldsymbol{q}} \ . \tag{2}$$

where $\mathbf{J}(\boldsymbol{q}) \in \mathbb{R}^{6\times n}$ is a geometric Jacobian matrix.

3 Redundancy

A redundant robot has more DOFs than it is necessary to perform a task. This implies that the redundancy is not only a feature of the robot structure but depends also on the task. To identify different types of redundancy, it is beneficial to define the following spaces:

- the *configuration space* $\mathcal{C} \in \mathbb{R}^n$ where the joint variables $\boldsymbol{q}$ are defined;
- the *operational space* $\mathcal{O} \triangleq SE(3)$ is the 6-dimensional Cartesian space, where the positions/orientations of the robot end-effector are defined; and
- the *task space* $\mathcal{T}$ is a Cartesian subspace where the task is defined, $\mathcal{T} \subseteq \mathcal{O}$.

Serial robots for which $\dim(\mathcal{O}) < \dim(\mathcal{C})$ are *intrinsic redundant*. When the task does not require that all 6 spatial DOFs are controlled, i.e. $\dim(\mathcal{T}) < \dim(\mathcal{O})$, the robot becomes *functional redundant*.

The task can be viewed as constraints of the end-effector pose $\boldsymbol{x}$ along the path. In some cases, the task allows that the constraints for one or more DOFs are relaxed, i.e. the end-effector can be in some region $\mathcal{A} \subset \mathcal{T}$ around the nominal path. Let the task be defined as a path in task space $\mathcal{T}$ as

$$\boldsymbol{x}_d = \boldsymbol{x}_d(s) \in \mathcal{T} \ , \tag{3}$$

where s is a strictly increasing path parameter, $s \subseteq [0, 1]$ and let $\boldsymbol{\xi} \in \mathbb{R}^k$ be a set of coordinates in $\mathcal{T}$ corresponding to spatial directions in which the task allows

some free motion. Then the free-motion region $\mathcal{A}(s)$ is defined as a subspace of $\mathcal{T}$ spanned by $\boldsymbol{\xi}$ with origin in $\boldsymbol{x}_d(s)$ and with bounds

$$B_i = [\underline{\boldsymbol{\xi}}_i(s,t), \overline{\boldsymbol{\xi}}_i(s,t)], \qquad i = 1, \ldots, k \,, \tag{4}$$

where $\underline{\boldsymbol{\xi}}_i(s)$ and $\overline{\boldsymbol{\xi}}_i(s)$ are minimal and maximal deviations from $\boldsymbol{x}_d(s)$ in spatial direction $\boldsymbol{\xi}_i$, respectively. The region $\mathcal{A}$ can change along the path or in time. In [3] these regions are termed Task Space Regions (TSRs) and are used for manipulation planning. In the following, we will present how the motion in $\mathcal{A}$ can be included in the redundancy resolution at the control level. We denote that when $\mathcal{A}$ exists, the robot becomes *bounded functional redundant*.

4 Kinematic control

To this day many different control approaches for robots have been proposed. As robot manipulators are highly nonlinear dynamical systems, most of the proposed control strategies use some kind of inner-loop inverse dynamic control like computed torque technique or operational space control to compensate nonlinearities. Here, we focus on the redundancy resolution at the velocity level and we assume that the inner-loop controller already compensates the nonlinear dynamics of the robot.

The main idea of the redundancy resolution at the velocity level is to compute the necessary outer-loop control velocity $\dot{\boldsymbol{q}}_c$ by inverting Eq. (2)

$$\dot{\boldsymbol{q}}_c = \mathbf{J}^{\#}(\boldsymbol{q})\dot{\boldsymbol{x}}_c + \left(\mathbf{I} - \mathbf{J}^{\#}(\boldsymbol{q})\mathbf{J}(\boldsymbol{q})\right)\dot{\boldsymbol{q}}_n \,, \tag{5}$$

where $\dot{\boldsymbol{x}}_c$ represents the desired task-space control velocity and $\dot{\boldsymbol{q}}_n$ is an arbitrary joint velocity, which is projected into the null-space of $\mathbf{J}$ and can be used to perform some additional lower priority subtasks.

It is common to select the task-space control velocity $\dot{\boldsymbol{x}}_c$ in (5) as

$$\dot{\boldsymbol{x}}_c = \boldsymbol{v}_d + \mathbf{K}_p \boldsymbol{e} \,, \tag{6}$$

where $\boldsymbol{v}_d$ is the desired end-effector position/rotation velocity, $\mathbf{K}_p$ is the diagonal gain matrix, which define the close-loop behavior, and $\boldsymbol{e}$ is the end-effector position/rotation error. It is defined as

$$\boldsymbol{e} = \begin{bmatrix} \boldsymbol{p}_d - \boldsymbol{p} \\ 2\log(\mathcal{Q}_d \mathcal{Q}^{-1}) \end{bmatrix} \,, \tag{7}$$

where $\boldsymbol{x}_d = \{\boldsymbol{p}_d, \mathcal{Q}_d\}$ represents the desired pose of the end-effector. Note that the velocity $\dot{\boldsymbol{x}}_c$ and errors $\boldsymbol{e}$ can be expressed in any task space as long as the Jacobian matrix $\mathbf{J}$ relates the robot configuration space and the selected task space $\mathcal{T}$.

The task space $\mathcal{T}$ and consequently the Jacobian matrix $\mathbf{J}$ are defined considering intrinsic and/or functional redundancy of the robot. However, when bounded functional redundancy is present, then (5) does not make possible to

use the motion in $\mathcal{A}$ for additional lower priority subtasks. As $\mathcal{A} \subset \mathcal{T}$ the spatial directions $\boldsymbol{\xi}$ are also included in $\mathcal{T}$. Therefore, projecting $\dot{\boldsymbol{q}}_n$ into the null-space of $\mathbf{J}$ using $(\mathbf{I}-\mathbf{J}^{\#}\mathbf{J})$ does not produce any motion in $\mathcal{T}$, meaning that the available motion in $\mathcal{A}$ cannot be exploited.

To exploit the motion in $\mathcal{A}$ for secondary tasks, it is necessary that spatial directions $\boldsymbol{\xi}$ are excluded from $\mathcal{T}$ and included in null-space projector $(\mathbf{I}-\mathbf{J}^{\#}\mathbf{J})$. Let denote $\mathcal{T}_{\mathcal{A}}$ the task space spanned over spatial directions excluding directions $\boldsymbol{\xi}$ and the corresponding Jacobian matrix as $\mathbf{J}_{\mathcal{A}}$. Then, projected velocities $(\mathbf{I} - \mathbf{J}_{\mathcal{A}}^{\#}\mathbf{J}_{\mathcal{A}})\dot{\boldsymbol{q}}_n$ have components also in directions $\boldsymbol{\xi}$. However, using $\mathbf{J}_{\mathcal{A}}$ in (5) is not allowed as this would allow moving along the task path. To overcome this problem, we propose to use the following kinematic control, which combines redundancy resolution in space $\mathcal{T}$ and in task space $\mathcal{T}_{\mathcal{A}}$.

The affordance that free-motion region around the path exists can be expressed in (6) by applying dead-zones to the components of $\boldsymbol{e}$ that are corresponding to $\boldsymbol{\xi}$

$$\widehat{\boldsymbol{e}}_i = \begin{cases} (\boldsymbol{e}_i - \overline{\boldsymbol{e}}_i) & \text{for } \boldsymbol{e}_i > \overline{\boldsymbol{e}}_i \\ 0 & \text{otherwise} \\ (\boldsymbol{e}_i - \underline{\boldsymbol{e}}_i) & \text{for } \boldsymbol{e}_i < \underline{\boldsymbol{e}}_i \end{cases} \quad \text{and} \quad i \in \boldsymbol{\xi} \tag{8}$$

Next, we augment the control (6) with a term corresponding to the task space velocities generated by the projection of $\dot{\boldsymbol{q}}_n$ in the null-space of $\mathbf{J}_{\mathcal{A}}$

$$\dot{\boldsymbol{x}}_{\mathcal{A}} = \mathbf{J}(\mathbf{I} - \mathbf{J}_{\mathcal{A}}^{\#}\mathbf{J}_{\mathcal{A}})\dot{\boldsymbol{q}}_n \tag{9}$$

yielding

$$\dot{\boldsymbol{x}}_c = \boldsymbol{v}_d + \mathbf{K}_p\widehat{\boldsymbol{e}} + \mathrm{sat}(\dot{\boldsymbol{x}}_{\mathcal{A}}, \overline{\dot{\boldsymbol{x}}}_i, \underline{\dot{\boldsymbol{x}}}_i)\,, \tag{10}$$

where $\mathrm{sat}(a)$ is the saturation function

$$\mathrm{sat}(a, a_{max}, a_{min}) = \begin{cases} a_{max} & \text{for } a > a_{max} \\ a & \text{for } a_{min} < a < a_{max} \\ a_{min} & \text{for } a < a_{min} \end{cases}. \tag{11}$$

To prevent the violation of constraints defined by $\mathcal{A}$, the dead-zone parameters in (8) are selected as

$$\overline{\boldsymbol{e}}_i = \overline{\boldsymbol{\xi}}_i - \frac{1}{\mathbf{K}_{p,i}}\,\overline{\dot{\boldsymbol{x}}}_i \qquad \text{and} \qquad \underline{\boldsymbol{e}}_i = \underline{\boldsymbol{\xi}}_i - \frac{1}{\mathbf{K}_{p,i}}\,\underline{\dot{\boldsymbol{x}}}_i\,, \tag{12}$$

where $\overline{\dot{\boldsymbol{x}}}_i$ and $\underline{\dot{\boldsymbol{x}}}_i$ are the limits for the task velocities generated by (9), respectively.

5 Case study

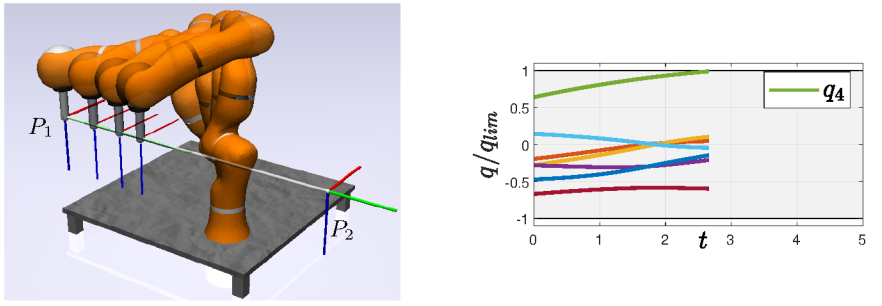
The application of the proposed control scheme does not depend on the complexity of the motion the robot has to perform. It solely allows some freedom-of-motion around the task trajectory, which depends on the task requirements.

Hence, to illustrate the usefulness of the proposed control scheme we have selected a simple line tracking task with a 7-DOF robot KULA LWR. The task was to move along line from the point P_1, $\boldsymbol{p}_1 = [0.25,\ -0.5,\ 0.4]^T$, to the point P_2, $\boldsymbol{p}_2 = [0.25,\ 0.5,\ 0.4]^T$ with a constant orientation $\mathcal{Q} = \mathcal{Q}(R_y(\pi))$ (rotation around y-axis) and with velocity $\boldsymbol{v}_d = [0,\ 0.125,\ 0,\ 0,\ 0,\ 0]^T$ (see Fig. 1a)). The robot has one redundant DOF, which has been used to avoid joint limits. For that, we propose to use the following null-space velocity control

$$\dot{q}_{n,i} = K_{n,i} \begin{cases} (a\, q_{i,max} - q_i) & \text{for } \ q_i > a\, q_{i,max} \\ 0 & \text{otherwise} \\ (a\, q_{i,min} - q_i) & \text{for } \ q_i < a\, q_{i,min} \end{cases} \qquad i = 1, \ldots, n\,, \tag{13}$$

where $q_{i,max}$ and $q_{i,min}$ are the joint limits and a has been used to select the critical distance to the joint limits.

The path was intentionally selected so, that the typical robot control algorithm could not move along the path without violating constraints. Fig. 1 shows the response of the system with control (5). We can see that the end-effector tracks the desired path but during the motion one joint has reached the limit (q_4) (see Fig. 1b) and hence, the robot has stopped and aborted the task execution.



a) 3D path tracking

b) Normalized joint positions (regarding joint limits)

Fig. 1. KUKA LWR robot moving along straight line without exploiting bounded functional redundancy; $\mathbf{K}_p = 100$, $a = 0.9$.

If some deviations from the desired path are allowed, then the task can be completed. Here we assumed that when moving along the path a rotation around y-axis is allowed in a bounded region. Note that no deviations are allowed at the end of the path at P_2, where the robot has to be in the desired pose. This free rotation forms a one dimensional subspace $\mathcal{A}_1$ spanned over $\boldsymbol{\xi} = [r_y]$ (we denote with r_a the axis of spatial rotation around a-axis). The bounds of $\mathcal{A}$ have been selected as $\overline{\boldsymbol{\xi}} = -\underline{\boldsymbol{\xi}} = \min(2\|\boldsymbol{p}(s) - \boldsymbol{p}(1)\|, 0.4)$rad. Fig. 2 shows the simulation results. We can see that using motion in $\mathcal{A}$ enables the robot to complete the

task. When q_4 is close to the joint limit, the null-space control uses motion in direction $\boldsymbol{\xi}_{r_y}$ to avoid joint limit.

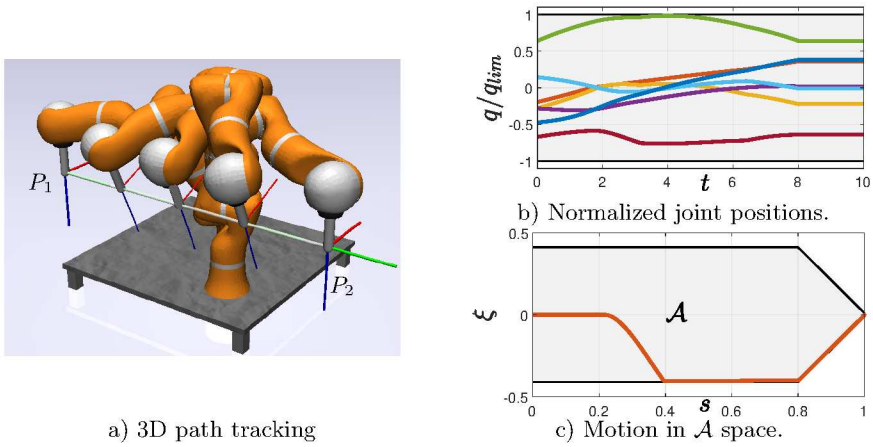
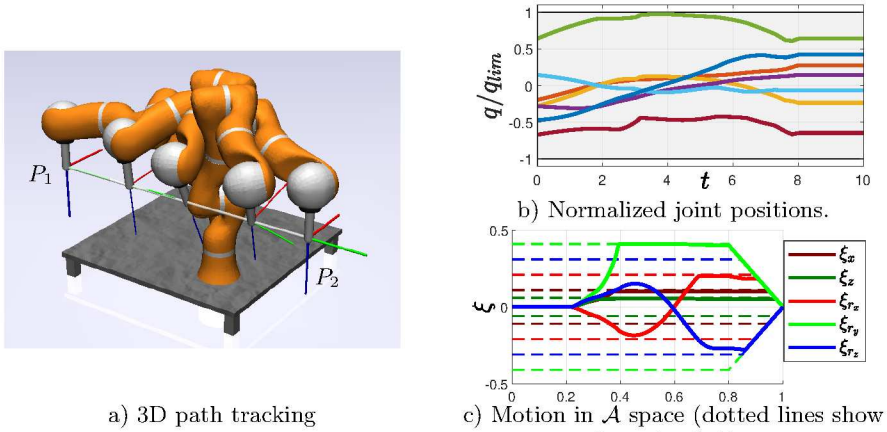


Fig. 2. KUKA LWR robot moving along straight line exploiting bounded functional redundancy (rotation around y-axis); $\mathbf{K}_p = 100$, $a = 0.9$.

Finally, let assume that path tracking allows deviations in orientation and positions, except in the direction toward the goal point P_2. In this case, the subspace $\mathcal{A}$ has been spanned over $\boldsymbol{\xi} = [x, z, r_x, r_y, r_z]^T$ and the bounds have been $\overline{\boldsymbol{\xi}} = -\underline{\boldsymbol{\xi}} = \min(2\|\boldsymbol{p}(s) - \boldsymbol{p}(1)\|, [0.1\text{m}, 0.05\text{m}, 0.2\text{rad}, 0.4\text{rad}, 0.6\text{rad}]^T)$. The resulting motion is shown in Fig. 3. We can see that the robot uses motion in $\mathcal{A}$ to avoid joint limits. The deviations from the path during the motion have been in the prescribed bounds and the final pose in P_2 has been reached as required.

6 Conclusion

In this paper introduced a novel methodology which allows exploiting a bounded free-motion around the prescribed path for redundancy resolution. When the task does not need a controlled motion in a certain spatial direction, then this DOF can be excluded from the task space and hence, it is included in the null-space. However, when a motion in certain spatial directions has to be controlled then this DOF is part of the task space and can not be used for secondary tasks using common redundancy resolution methods. The proposed strategy combines controllers in two task spaces: in main task space $\mathcal{T}$, where DOFs with allowed deviations are included, and in the second task space $\mathcal{T}_\mathcal{A}$, where these DOFs are excluded. The motion generated by the controller in $\mathcal{T}_\mathcal{A}$ is mapped into $\mathcal{T}$ and by properly selecting the saturation parameters of this controller, the resulting motion does not violate the constraints in $\mathcal{T}$. This is also illustrated by simulation examples.



a) 3D path tracking

b) Normalized joint positions.

c) Motion in $\mathcal{A}$ space (dotted lines show bounds of $\mathcal{A}$).

Fig. 3. KUKA LWR robot moving along straight line exploiting bounded functional redundancy (rotation around all axes); $\mathbf{K}_p = 100$, $a = 0.9$.

References

1. Alatartsev, S., Belov, A., Nykolaychuk, M., Ortmeier, F.: Robot Trajectory Optimization for the Relaxed End-effector Path. In: Proceedings of the 11th International Conference on Informatics in Control, Automation and Robotics. pp. 385–390. No. September, Vienna, Austria (2014)
2. Baron, L.: A Joint-Limits Avoidance Strategy for Arc-Welding Robots. In: Int. Conf. on Integrated Design and Manufacturing in Mech. Eng. Montreal, Canada (2000)
3. Berenson, D., Srinivasa, S., Kuffner, J.: Task Space Regions: A framework for pose-constrained manipulation planning. International Journal of Robotics Research 30(12), 1435–1460 (2011)
4. From, P.J., Gravdahl, J.T.: A real-time algorithm for determining the optimal paint gun orientation in spray paint applications. IEEE Transactions on Automation Science and Engineering 7(4), 803–816 (2010)
5. Huo, L., Baron, L.: Kinematic inversion of functionally-redundant serial manipulators: Application to arc-welding. Transactions of the Canadian Society for Mechanical Engineering 29(4), 679–690 (2005)
6. Leger, J., Angeles, J.: A Redundancy-resolution Algorithm for Five-degree-of-freedom Tasks via Sequential Quadratic Programming. In: TrC-IFToMM Symposium on Theory of Machines and Mechanisms,. Izmir (2015)
7. Sanderud, A.R., Thomessen, T., Hashimoto, H., Osumi, H., Niitsuma, M.: An approach to path planning and real-time redundancy control for human-robot collaboration. IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM pp. 1018–1023 (2014)
8. Sciavicco, L., Siciliano, B.: Modelling and Control of Robot Manipulators. Advanced textbooks in control and signal processing, Springer, London, 2nd edn. (2000)
9. Zanchettin, A.M., Rocco, P.: On the use of functional redundancy in industrial robotic manipulators for optimal spray painting. In: IFAC Proceedings Volumes (IFAC-PapersOnline). vol. 18, pp. 11495–11500 (2011)