Singularity Avoidance of Task-redundant Robots in Pointing Tasks: On Nullspace Projection and Cardan Angles as Orientation Coordinates

Moritz Schappler[a] and Tobias Ortmaier[b]
Institute of Mechatronic Systems, Leibniz University Hannover, An der Universität 1, 30823 Garbsen, Germany

Keywords: Singularity Avoidance, Nullspace Motion, Task Redundancy, Parallel Robots, Euler Angles, Cardan Angles.

Abstract: Robot manipulators are often deployed in tool-symmetric tasks, which only requires defining end effector position and pointing direction. In this case six-axis serial industrial robots and full-mobility (spatial) parallel robots have one degree of task redundancy. Using Cardan angles as orientation coordinates, a unified formulation of the position-level and second-order inverse kinematics problem is set up for both robot types. An efficient scheme for difference-quotient approximation of gradients of performance criteria for projection into the task redundancy's nullspace is presented. The simulation example of a hexapod robot shows that avoiding and exiting parallel robot singularities of type II is possible with the nullspace of all joints. The nullspace controller scheme can be used in offline trajectory optimization and in online motion generation.

1 INTRODUCTION AND STATE OF THE ART

Robot manipulators have been a subject of research for decades. Still more complex parallel kinematic machines (parallel robots), and more complex problems arise, such as mastering singularities and motion in high-dimensional spaces. Due to their complexity, these problems can only be solved with profound algorithms and by the help of information technology.

The optimization of robots performing tasks with axis-symmetric tools like arc welding (Huo and Baron, 2008), drilling (Guo et al., 2015) or milling (Mousavi et al., 2018) has gained increasing attention in research and industry over the last years. Due to their workspace, serial robots are predestinated for welding tasks but also for drilling tasks e.g. in the aircraft industry. For machining tasks parallel robots are favorable due to their higher stiffness than serial kinematic machine tools or serial robot arm manipulators.

Serial industrial robots as well as the hexapod parallel robot often used for machining tasks have six degrees of freedom (DoF) in the operational space and in the (actuation) joint space. They do not have *intrinsic redundancy* (for serial robots) or *kinematic or actuation redundancy* (in the case of parallel robots). A *task (or functional) redundancy* of degree one exists, if the six-DoF robot is used in an axis-symmetric task (requiring only five DoF). For a definition of the redundancy see (Huo and Baron, 2008) or (Léger and Angeles, 2016) for serial robots and (Gosselin and Schreiber, 2018) for parallel robots.

The redundancy allows defining a nullspace and performing a pose optimization using gradient projection into that nullspace. This method is very efficient and already well-established for *kinematic* redundancy (Nakamura et al., 1987; Chiaverini et al., 2008; Lillo et al., 2019). The redundancy of a rotational end effector DoF requires the adaption of the inverse kinematics problem (IKP) to account for the nonlinearity of rotation. Further, the IKP can be distinguished between position level and velocity level or higher differential order. Several geometric approaches for the IKP of *task*-redundant robots have been investigated, such as

- twist decomposition (Huo and Baron, 2008),
- orthogonal decomposition of the task space (Léger and Angeles, 2016; Corinaldi et al., 2016),
- reciprocal Euler angles (Schappler et al., 2019),
- separation of joint coordinates in redundant and non-redundant on position level (Ozgoren, 2013) or on velocity level (Reiter et al., 2018),
- expressing the end effector angular velocity in the local frame and removing the last component (Žlajpah, 2017; Reiter et al., 2018) corresponding to the redundant coordinate.

[a] https://orcid.org/0000-0001-7952-7363
[b] https://orcid.org/0000-0003-1644-3685

Schappler, M. and Ortmaier, T.
Singularity Avoidance of Task-redundant Robots in Pointing Tasks: On Nullspace Projection and Cardan Angles as Orientation Coordinates.
DOI: 10.5220/0010621103380349
In *Proceedings of the 18th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2021)*, pages 338-349
ISBN: 978-989-758-522-7

The alternative to the modified inverse kinematics (IK) formulations above is a full formulation of the IK (ignoring task redundancy). Then, a subsequent optimization of the redundant task space coordinate is necessary, as performed by (Zhu et al., 2013), (Guo et al., 2015) and (Mousavi et al., 2018) for serial robots. Especially for parallel robots this approach was previously pursued combined with

- interval analysis (Merlet et al., 2000),
- an iterative solution using linear or quadratic programming (Oen and Wang, 2007),
- discrete switching patterns of the redundant coordinate in rest positions (Kotlarski et al., 2010),
- iterations of small-angle perturbations of the redundant coordinate (Gao et al., 2019).

In special variations of the milling task using parallel kinematic machines (PKM), more than one rotational coordinate can be treated as redundant. Examples are end milling (Shaw and Chen, 2001) or milling with a spherical cutter (Smirnov et al., 2013).

The overview already shows that gradient-based local optimization approaches are more widely used in serial robotics than in parallel robotics, where global optimizations dominate. A promising strategy is the combination of local and global approaches, e.g. by using differential dynamic programming (Santos and da Silva, 2017).

Only in recent publications, gradient-based optimization with nullspace projection is performed for parallel robots, with focus on kinematic redundancy by (Gosselin and Schreiber, 2016) and (Santos and da Silva, 2017) and on task redundancy by (Agarwal et al., 2016). The main effort is put on the kinematic and actuation redundancy, as can be seen in the extensive reviews on parallel robot redundancy (Luces et al., 2017) and (Gosselin and Schreiber, 2018), where task redundancy is not even mentioned.

Redundancy is exploited to improve performance criteria of the robot manipulators, such as joint limits (Huo and Baron, 2008; Zhu et al., 2013) or milling process stability (Mousavi et al., 2018). Singularity avoidance is implemented by criteria such as

- the distance in the joint space to the configuration that first exceeded a parameter of singularity (Huo and Baron, 2008),
- the squared condition number (Zhu et al., 2013; Léger and Angeles, 2016; Corinaldi et al., 2016) via the Frobenius norm relation (Merlet, 2006),
- the condition number of the PKM forward kinematics Jacobian (Gosselin and Schreiber, 2016),
- the consideration of all singular values of the Jacobian (Santos and da Silva, 2017),
- the homogenized pose error as a measure for accuracy and singularities (Kotlarski et al., 2010),
- the Jacobian's determinant (Agarwal et al., 2016).

Although the physical meaning of the manipulator Jacobian's condition number is questionable, cf. (Kotlarski et al., 2010), using it as a measure for singularity avoidance is well established and can be justified, even without homogenization of units, cf. (Merlet, 2006). Especially for parallel robots singularities can be located anywhere in the workspace and avoiding them in offline trajectory planning and online trajectory execution is paramount, cf. (Luces et al., 2017; Gosselin and Schreiber, 2018).

The work from (Agarwal et al., 2016) presents a promising approach to tackle the problem of singularity avoidance for parallel robots using gradient projection in the nullspace of task redundancy. However, open points remain, regarding the feedback loop design of the nullspace motion and an extension to spatial robots regarding the nonlinearity of rotation. These points will be encountered in this paper by a consideration of Cardan angles as end effector coordinates, as in (Schappler et al., 2019), by considering a second-order nullspace motion, cf. (Reiter et al., 2018) and by incorporating aspects of control design from (De Luca et al., 1992), where this problem was already approached for serial robots. To summarize, the contributions of this paper are

- a formulation of the differential inverse kinematics problem for one-DoF task redundancy using Cardan angles for end effector orientation,
- a numeric scheme for simplification of the computation of gradients of performance criteria in the nullspace projection using difference quotients,
- unifying the scheme for serial and parallel robots,
- and exemplary simulative studies on offline trajectory planning with singularity avoidance.

The remainder of the paper is structured as follows. After a remark on task coordinates in section 2, the kinematics model and nullspace motion is laid out in section 3 for serial robots and in section 4 for parallel robots. Considerations on the control loop design and remarks on convergence are given in section 5 and the scheme is validated in section 6 at the example of pose optimization and trajectory tracking of a six-DoF parallel robot. Section 7 concludes the paper.

2 ON TASK COORDINATES

In the following, the six *operational space coordinates*, i.e. the position and orientation of a robot's end effector in the Cartesian space, are expressed with the vector $\boldsymbol{x} = (r_x, r_y, r_z, \varphi_x, \varphi_y, \varphi_z)^{\mathrm{T}}$. Unlike for the unambiguous position part $\boldsymbol{r}$, a suitable representation of rotation has to be chosen for the orientation part $\boldsymbol{\varphi}$.

For tasks with rotational symmetry given in section 1, a selection of the Euler angle convention is suitable, where the last of the three (intrinsic) elementary rotations is defined around the tool axis. The Cardan angles (or *X-Y'-Z*" Euler angles) are especially practical and intuitive, since the tool axis in robotics is by convention the z_E-axis. The end effector's rotation matrix w.r.t. the base frame is defined by ${}^0\boldsymbol{R}_E(\boldsymbol{x}) = \boldsymbol{R}_x(\varphi_x)\boldsymbol{R}_y(\varphi_y)\boldsymbol{R}_z(\varphi_z)$ using the basic rotation matrices. The angle convention established in tool machines on the contrary follows the extrinsic *X-Y-Z* Euler angles convention for expressing orientation, which corresponds to the intrinsic *Z-Y'-X*" convention. The rotation axes are termed "A, B and C" (corresponding to "X, Y and Z" translation), (Smirnov et al., 2013). The term "Euler angles" is used for generalization, also including Tait-Bryan (and Cardan) angles and is not limited to proper Euler angles. Representing orientation with Euler angles can introduce mathematical singularities corresponding to a gimbal lock. For Cardan angles this only occurs for φ_y=±90°, which is not relevant for most tasks and e.g. technically not possible for most parallel robots and therefore out of this paper's scope.

The *task space coordinates* of pointing tasks are defined as $\boldsymbol{y} = (r_x, r_y, r_z, \varphi_x, \varphi_y)^{\mathrm{T}}$. Due to the rotational symmetry of the robot tool and the chosen representation of rotation, the last operational space coordinate φ_z corresponds to a rotation of the robot end effector around the tool axis and can be selected arbitrarily in the regarded case of task redundancy.

3 SERIAL-LINK ROBOTS

Redundancy resolution for serial robots has been widely researched, cf. (Chiaverini et al., 2008), but the case of task redundancy is only implicitly included in the fundamental works. In the following, section 3.1 clarifies some foundations e.g. from (Chiaverini et al., 2008) regarding task redundancy and section 3.2 introduces new aspects on the computation of the nullspace in task redundancy.

3.1 Inverse Kinematics Model

The forward kinematics problem relates the $n_{\boldsymbol{q}}$ joint space coordinates $\boldsymbol{q}$ with the operational space by

$$\boldsymbol{x} = \left[\boldsymbol{x}_{\mathrm{t}}^{\mathrm{T}}, \boldsymbol{x}_{\mathrm{r}}^{\mathrm{T}}\right]^{\mathrm{T}} = \left[\boldsymbol{r}_E^{\mathrm{T}}(\boldsymbol{q}), \boldsymbol{\varphi}_{XYZ}^{\mathrm{T}}({}^0\boldsymbol{R}_E(\boldsymbol{q}))\right]^{\mathrm{T}}. \quad (1)$$

The non-redundant inverse kinematics problem (IKP) can be expressed by an implicit residual in $\mathbb{R}^6$

$$\boldsymbol{\Phi}(\boldsymbol{q},\boldsymbol{x}) = \begin{bmatrix}\boldsymbol{\Phi}_{\mathrm{t}}\\ \boldsymbol{\Phi}_{\mathrm{r}}\end{bmatrix} = \begin{bmatrix}-\boldsymbol{x}_{\mathrm{t}} + \boldsymbol{r}_E(\boldsymbol{q})\\ \boldsymbol{\theta}\left({}^0\boldsymbol{R}_E^{\mathrm{T}}(\boldsymbol{x}_{\mathrm{r}})\,{}^0\boldsymbol{R}_E(\boldsymbol{q})\right)\end{bmatrix} = \boldsymbol{0} \quad (2)$$

with arbitrary convention $\boldsymbol{\theta}$, which can be solved numerically using the Newton-Raphson algorithm. In the case of task redundancy, a reduced residual

$$\boldsymbol{\Psi}(\boldsymbol{q},\boldsymbol{y}) = \boldsymbol{0} \in \mathbb{R}^5 \quad \text{with rotational part} \quad (3)$$

$$\boldsymbol{\Psi}_{\mathrm{r}} = \begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix}\boldsymbol{\theta}_{ZYX}\left({}^0\boldsymbol{R}_E^{\mathrm{T}}(\boldsymbol{x}_{\mathrm{r}})\,{}^0\boldsymbol{R}_E(\boldsymbol{q})\right) \in \mathbb{R}^2 \quad (4)$$

is needed, which can be set up by using the Euler angles $\boldsymbol{\theta}$ of the rotation matrix between desired pose $\boldsymbol{x}$ and actual pose $\boldsymbol{x}(\boldsymbol{q})$. The angle convention in $\boldsymbol{\Psi}$ has to be reciprocal (i.e. *Z-Y'-X*") to the one used for the end effector pose $\boldsymbol{x}$, as derived in (Schappler et al., 2019). The redundant coordinate φ_z has no effect.

Using a task-redundant robot with $n_{\boldsymbol{q}} \geq 6$ allows to optimize additional objective functions $h(\boldsymbol{q})$ using nullspace projection in the iterative solution by

$$\boldsymbol{q}_{\mathrm{opt}} = \underset{\boldsymbol{q}}{\operatorname{argmin}}(h(\boldsymbol{q})) \quad \text{s.t.} \quad \boldsymbol{\Psi}(\boldsymbol{q},\boldsymbol{y}) = \boldsymbol{0}. \quad (5)$$

The iterative solution in the step k is obtained with the Moore-Penrose pseudo-inverse by

$$\boldsymbol{q}^{k+1} = \boldsymbol{q}^k - \boldsymbol{\Psi}_{\partial\boldsymbol{q}}^{\#}\boldsymbol{\Psi} + (\boldsymbol{I}_{n_{\boldsymbol{q}}} - \boldsymbol{\Psi}_{\partial\boldsymbol{q}}^{\#}\boldsymbol{\Psi}_{\partial\boldsymbol{q}})(-h_{\partial\boldsymbol{q}}^{\mathrm{T}}), \quad (6)$$

where the index $\partial\boldsymbol{q}$ denotes the derivative of the symbol w.r.t. $\boldsymbol{q}$, # the pseudo-inverse, $\boldsymbol{I}_{n_{\boldsymbol{q}}}$ the identity matrix of appropriate dimension $n_{\boldsymbol{q}}$ and the dependency on $\boldsymbol{q}^k$ and $\boldsymbol{y}$ is omitted for the sake of brevity. The notation will be kept throughout the paper.

Using this allows to optimize the robot for one specific pose, e.g. for drilling tasks, where the feed motion is negligible regarding the robot's dimension. To obtain a continuous trajectory, e.g. for milling tasks, the forward differential kinematics

$$\dot{\boldsymbol{x}} = \boldsymbol{J}_{\boldsymbol{x}}\dot{\boldsymbol{q}} \quad \text{and} \quad \ddot{\boldsymbol{x}} = \dot{\boldsymbol{J}}_{\boldsymbol{x}}\dot{\boldsymbol{q}} + \boldsymbol{J}_{\boldsymbol{x}}\ddot{\boldsymbol{q}} \quad (7)$$

are used. The matrix $\boldsymbol{J}_{\boldsymbol{x}}$ is known as the analytic Jacobian. The geometric Jacobian on the contrary relates angular velocities in the base frame instead of time derivatives of the Euler angles to the joint velocities.

The transfer from operational space to task space is performed by removing the last row of (7) with $\boldsymbol{y} = \boldsymbol{P}_{\boldsymbol{y}}\boldsymbol{x}$ and $\boldsymbol{J}_{\boldsymbol{y}} = \boldsymbol{P}_{\boldsymbol{y}}\boldsymbol{J}_{\boldsymbol{x}}$. The matrices $\boldsymbol{J}_{\boldsymbol{y}}$ and $\boldsymbol{\Psi}_{\partial\boldsymbol{q}}$ are not identical due to the nonlinearity of rotation in SE(3). The task space inverse differential kinematics are obtained by the pseudo-inverse solution

$$\ddot{\boldsymbol{q}} = \ddot{\boldsymbol{q}}_{\mathrm{T}} + \ddot{\boldsymbol{q}}_{\mathrm{N}} = \boldsymbol{J}_{\boldsymbol{y}}^{\#}(\ddot{\boldsymbol{y}} - \dot{\boldsymbol{J}}_{\boldsymbol{y}}\dot{\boldsymbol{q}}) + (\boldsymbol{I}_{n_{\boldsymbol{q}}} - \boldsymbol{J}_{\boldsymbol{y}}^{\#}\boldsymbol{J}_{\boldsymbol{y}})\boldsymbol{v}. \quad (8)$$

Again as in (6), nullspace projection of arbitrary vectors $\boldsymbol{v}$ can be performed. The vector may be chosen as $\boldsymbol{v} = -h_{\partial\boldsymbol{q}}^{\mathrm{T}}$ or as discussed in section 5.

3.2 Task Redundancy and Nullspace

Objective functions $h(\boldsymbol{q})$ can be defined analytically or numerically. The former is not possible or feasible for all functions. For the latter a method for the efficient derivation of the gradient $h_{\partial\boldsymbol{q}}$ will be presented and discussed for several cases.

3.2.1 General Case

The general case is not limited to task redundancy and can be used in the position-level inverse kinematics problem (6) for $\|\boldsymbol{\Psi}\| \not\approx 0$. A difference quotient is calculated for a small increment Δq_i of all joints by

$$h_{\partial q_i} \approx (h(\boldsymbol{q}+\Delta\boldsymbol{q}) - h(\boldsymbol{q}))/\Delta q_i, \tag{9}$$

where $\Delta q_j = 0$ with $j \neq i$ in $\Delta\boldsymbol{q}$. This requires $n_{\boldsymbol{q}}+1$ evaluations of the objective function $h(\boldsymbol{q})$ in each iteration of (6) or each discrete trajectory sample of (8). This is e.g. mentioned in (Lillo et al., 2019), but is probably used in many implementations.

3.2.2 Task Redundancy of Degree One

In the case of task redundancy with $n_{\boldsymbol{q}} - \dim(\boldsymbol{y}) = 1$, the nullspace projection can be exploited to simplify the computation. The assumption is that $\|\boldsymbol{\Psi}\| \approx 0$, which is always the case in the differential inverse kinematics (8). In the position-level IKP (6) convergence of the task or an already valid initial value $\boldsymbol{q}^0$ is required. Writing $\boldsymbol{q} := \boldsymbol{q}(\boldsymbol{x})$ leads to

$$\frac{\partial h}{\partial \boldsymbol{x}} = \frac{\partial h}{\partial \boldsymbol{q}}\frac{\partial \boldsymbol{q}}{\partial \boldsymbol{x}} \quad \text{or} \quad \frac{\partial h}{\partial \boldsymbol{q}} = \frac{\partial h}{\partial \boldsymbol{x}}\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{q}}. \tag{10}$$

The latter expression can be divided into the single operational space coordinates via

$$\frac{\partial h}{\partial \boldsymbol{q}} = \frac{\partial h}{\partial r_x}\frac{\partial r_x}{\partial \boldsymbol{q}} + \frac{\partial h}{\partial r_y}\frac{\partial r_y}{\partial \boldsymbol{q}} + \ldots + \frac{\partial h}{\partial \varphi_z}\frac{\partial \varphi_z}{\partial \boldsymbol{q}}. \tag{11}$$

The nullspace projector included in (6) or (8) is

$$\boldsymbol{N} = \boldsymbol{I}_{n_{\boldsymbol{q}}} - \boldsymbol{J}_{\boldsymbol{y}}^{\#}\boldsymbol{J}_{\boldsymbol{y}} = \boldsymbol{I}_{n_{\boldsymbol{q}}} - \boldsymbol{\Psi}_{\partial \boldsymbol{q}}^{\#}\boldsymbol{\Psi}_{\partial \boldsymbol{q}} \;\text{ for }\; \|\boldsymbol{\Psi}\| = 0. \tag{12}$$

Since the nullspace is defined as a rotation within the φ_z coordinate, the projector eliminates the other terms

$$\boldsymbol{N}\left(\frac{\partial h}{\partial r_x}\frac{\partial r_x}{\partial \boldsymbol{q}}\right)^{\mathrm{T}} = \boldsymbol{0}, \;\ldots,\; \boldsymbol{N}\left(\frac{\partial h}{\partial \varphi_y}\frac{\partial \varphi_y}{\partial \boldsymbol{q}}\right)^{\mathrm{T}} = \boldsymbol{0}. \tag{13}$$

Therefore the gradient in the case of one-DoF task redundancy can be simplified to

$$h_{\partial \boldsymbol{q}}^{*} = \frac{\partial h}{\partial \varphi_z}\frac{\partial \varphi_z}{\partial \boldsymbol{q}}, \tag{14}$$

where the last term is the last row of the Jacobian $\boldsymbol{J}_{\boldsymbol{x}}$. The asterisk denotes the simplified calculation of the gradient which follows the relations

$$\boldsymbol{N}(h_{\partial \boldsymbol{q}}^{*})^{\mathrm{T}} = \boldsymbol{N}(h_{\partial \boldsymbol{q}})^{\mathrm{T}} \quad \text{and} \quad h_{\partial \boldsymbol{q}}^{*} \neq h_{\partial \boldsymbol{q}}. \tag{15}$$

Using a small increment of the redundant coordinate $\Delta\boldsymbol{x} = (\boldsymbol{0}^{\mathrm{T}}, \Delta\varphi_z)^{\mathrm{T}}$, the first term can be determined numerically via the difference quotient

$$(\partial h/\partial \varphi_z) \approx (h(\boldsymbol{q} + \boldsymbol{J}_{\boldsymbol{x}}^{-1}\Delta\boldsymbol{x}) - h(\boldsymbol{q}))/\Delta\varphi_z. \tag{16}$$

This only requires two numeric evaluations of $h(\boldsymbol{q})$ which can be highly beneficial for computationally expensive objectives such as collision avoidance or singularity avoidance. The approach of simulating a $\Delta\varphi_z$ can be regarded similar to the principle of small-angle perturbation of (Gao et al., 2019), where a global instead of local optimization is performed.

4 PARALLEL ROBOTS

The inverse kinematics of task redundant parallel robots can be approached similarly as serial robots. A a parallel robot consists of m kinematic chains connecting a base link with a moving platform. Following e.g. (Merlet, 2006), the joint space $\boldsymbol{q}$ now also contains passive joints (here also including the platform coupling joints). Active joint coordinates are included in this formulation and are also denoted by $\boldsymbol{\theta}$. Further, the platform pose can be expressed by the operational space coordinate $\boldsymbol{x}$ which can also be used as a minimal coordinate of the robot.

In the following, section 4.1 again applies approaches from literature, e.g. (Merlet, 2006; Chiaverini et al., 2008) to task redundancy and section 4.2 discusses projection in the nullspace.

4.1 Inverse Kinematics Model

To solve the IKP for parallel robots, similar to (2) and (3) for serial robots, the full and reduced kinematic constraints $\boldsymbol{\Phi}(\boldsymbol{q},\boldsymbol{x}) \in \mathbb{R}^{6m}$ and $\boldsymbol{\Psi}(\boldsymbol{q},\boldsymbol{y}) \in \mathbb{R}^{6m-1}$ are defined. The latter can be obtained in the minimal form only dependent on the task space coordinate $\boldsymbol{y}$ by using reciprocal Euler angles for the orientation constraints of the first leg chain like in (4). The constraints of other (following) legs are defined relative to the first (leading) leg, e.g. as vector loop via the end effector, see (Schappler et al., 2019). While the IKP can be solved with the same Newton-Raphson approach (6) as serial robots, the differential relation

$$\boldsymbol{\Phi}_{\partial \boldsymbol{q}}\dot{\boldsymbol{q}} + \boldsymbol{\Phi}_{\partial \boldsymbol{x}}\dot{\boldsymbol{x}} = \boldsymbol{0} \quad \text{and} \quad \boldsymbol{\Psi}_{\partial \boldsymbol{q}}\dot{\boldsymbol{q}} + \boldsymbol{\Psi}_{\partial \boldsymbol{y}}\dot{\boldsymbol{y}} = \boldsymbol{0} \tag{17}$$

also has to take the platform coordinates $\boldsymbol{x}$ and accordingly $\boldsymbol{y}$ into account. The inverse differential kinematics is split up into the particular solution and the nullspace solution by $\ddot{\boldsymbol{q}} = \ddot{\boldsymbol{q}}_{\mathrm{T}} + \ddot{\boldsymbol{q}}_{\mathrm{N}}$. The task-related differential inverse kinematics on velocity level is

$$\dot{\boldsymbol{q}}_{\mathrm{T}} = -\boldsymbol{\Phi}_{\partial \boldsymbol{q}}^{-1}\boldsymbol{\Phi}_{\partial \boldsymbol{x}}\dot{\boldsymbol{x}} = \tilde{\boldsymbol{J}}_{\boldsymbol{x}}^{-1}\dot{\boldsymbol{x}}. \tag{18}$$

The matrix $\tilde{\boldsymbol{J}}_{\boldsymbol{x}}^{-1}$ is the Jacobian relating all joint velocities and the platform velocity (in operational space coordinates $\boldsymbol{x}$). The acceleration can be obtained by

$$\ddot{\boldsymbol{q}}_{\mathrm{T}} = \tilde{\boldsymbol{J}}_{\boldsymbol{x}}^{-1}\ddot{\boldsymbol{x}} + \dot{\tilde{\boldsymbol{J}}}_{\boldsymbol{x}}^{-1}\dot{\boldsymbol{x}}. \tag{19}$$

In the case of task redundancy, either the solution of (19) with $\dot{\varphi}_z$=$\ddot{\varphi}_z$=0 can be selected or the minimum-norm solution by using the $\boldsymbol{\Psi}$-related constraints with

$$\ddot{\boldsymbol{q}}_{\mathrm{T}} = -\boldsymbol{\Psi}_{\partial \boldsymbol{q}}^{\#} \left(\dot{\boldsymbol{\Psi}}_{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} + \dot{\boldsymbol{\Psi}}_{\partial \boldsymbol{y}} \dot{\boldsymbol{y}} + \boldsymbol{\Psi}_{\partial \boldsymbol{y}} \ddot{\boldsymbol{y}} \right). \quad (20)$$

Another approach is obtaining $\ddot{\boldsymbol{q}}_{\mathrm{T}}$ from time differentiation of $\dot{\boldsymbol{q}}_{\mathrm{T}} = -\boldsymbol{\Psi}_{\partial \boldsymbol{q}}^{\#} \boldsymbol{\Psi}_{\partial \boldsymbol{y}} \dot{\boldsymbol{y}}$, which was shown for serial robots by (Reiter et al., 2018). Similar to (8), the nullspace motion in the full joint space on acceleration level is obtained as

$$\ddot{\boldsymbol{q}}_{\mathrm{N}} = (\boldsymbol{I}_{n_{\boldsymbol{q}}} - \boldsymbol{\Psi}_{\partial \boldsymbol{q}}^{\#} \boldsymbol{\Psi}_{\partial \boldsymbol{q}}) \boldsymbol{v}_{\boldsymbol{q}}. \quad (21)$$

It has to be noted that for the full joint space of parallel robots a relation of the nullspace projector and the Jacobian like (12) for serial robots does not exist directly. Only the inverse matrix $\tilde{\boldsymbol{J}}_{\boldsymbol{x}}^{-1}$ can be defined, but not the non-inverse $\tilde{\boldsymbol{J}}_{\boldsymbol{x}}$, since $n_{\boldsymbol{q}} > \dim(\boldsymbol{x})$.

To be able to use the manipulator Jacobian $\boldsymbol{J}_{\boldsymbol{x}}$ for the nullspace projection, the nullspace motion has to be computed in the actuation space with $n_{\boldsymbol{\theta}}$ coordinates $\boldsymbol{\theta}$. This can be obtained by selection of components of the full joint space expressions with

$$\boldsymbol{\theta} = \boldsymbol{P}_{\boldsymbol{\theta}} \boldsymbol{q} \quad \text{and} \quad \boldsymbol{J}_{\boldsymbol{x}}^{-1} = \boldsymbol{P}_{\boldsymbol{\theta}} \tilde{\boldsymbol{J}}_{\boldsymbol{x}}^{-1}. \quad (22)$$

The manipulator Jacobian can be computed by (numeric) matrix inversion with

$$\boldsymbol{J}_{\boldsymbol{x}} = \left(\boldsymbol{J}_{\boldsymbol{x}}^{-1} \right)^{-1} = \left(\boldsymbol{P}_{\boldsymbol{\theta}} \tilde{\boldsymbol{J}}_{\boldsymbol{x}}^{-1} \right)^{-1}. \quad (23)$$

Only after this step, the reduced Jacobian $\boldsymbol{J}_{\boldsymbol{y}}$ without the row for the redundant task space coordinate φ_z can be obtained analogously to the serial robot case with $\boldsymbol{J}_{\boldsymbol{y}} = \boldsymbol{P}_{\boldsymbol{y}} \boldsymbol{J}_{\boldsymbol{x}}$. This approach does not require a minimal kinematics set, where passive joints are eliminated in the constraints resulting to $\boldsymbol{\Phi}(\boldsymbol{\theta}, \boldsymbol{x})$ instead of $\boldsymbol{\Phi}(\boldsymbol{q}, \boldsymbol{x})$, like e.g. used by (Agarwal et al., 2016).

Actuator speeds $\dot{\boldsymbol{\theta}}$ can be projected in the full joint space by

$$\dot{\boldsymbol{q}} = \tilde{\boldsymbol{J}}_{\boldsymbol{x}}^{-1} \boldsymbol{J}_{\boldsymbol{x}} \dot{\boldsymbol{\theta}}. \quad (24)$$

The nullspace motion can then be computed like in the serial robot case (8) with

$$\ddot{\boldsymbol{\theta}}_{\mathrm{T}} = \boldsymbol{J}_{\boldsymbol{y}}^{\#} (\ddot{\boldsymbol{y}} - \dot{\boldsymbol{J}}_{\boldsymbol{y}} \dot{\boldsymbol{\theta}}) \quad \text{and} \quad \ddot{\boldsymbol{\theta}}_{\mathrm{N}} = (\boldsymbol{I}_{n_{\boldsymbol{\theta}}} - \boldsymbol{J}_{\boldsymbol{y}}^{\#} \boldsymbol{J}_{\boldsymbol{y}}) \boldsymbol{v}_{\boldsymbol{\theta}}. \quad (25)$$

The two nullspace formulations lead to the same motion and combining (24) with (25) and (21) lead to the relation

$$\boldsymbol{N}_{\boldsymbol{q}} = k \tilde{\boldsymbol{J}}_{\boldsymbol{x}}^{-1} \boldsymbol{J}_{\boldsymbol{x}} \boldsymbol{N}_{\boldsymbol{\theta}} \boldsymbol{J}_{\boldsymbol{x}}^{\mathrm{T}} \tilde{\boldsymbol{J}}_{\boldsymbol{x}}^{-\mathrm{T}} \quad (26)$$

with $\boldsymbol{N}_{\boldsymbol{q}} = \boldsymbol{I}_{n_{\boldsymbol{q}}} - \boldsymbol{\Psi}_{\partial \boldsymbol{q}}^{\#} \boldsymbol{\Psi}_{\partial \boldsymbol{q}}$ and $\boldsymbol{N}_{\boldsymbol{\theta}} = \boldsymbol{I}_{n_{\boldsymbol{\theta}}} - \boldsymbol{J}_{\boldsymbol{y}}^{\#} \boldsymbol{J}_{\boldsymbol{y}}$. The relation of the scalar factor $k = k(\boldsymbol{q})$ is yet to be found. The structure of the nullspace of task redundancy is visible in the term $\boldsymbol{J}_{\boldsymbol{x}} \boldsymbol{N}_{\boldsymbol{\theta}} \boldsymbol{J}_{\boldsymbol{x}}^{\mathrm{T}}$, where only the last scalar entry is non-zero, corresponding to a motion in the coordinate φ_z. The serial robot case of (12) is included in (26) by the relation $\tilde{\boldsymbol{J}}_{\boldsymbol{x}}^{-1} \boldsymbol{J}_{\boldsymbol{x}} = \boldsymbol{I}$ due to $n_{\boldsymbol{\theta}} = n_{\boldsymbol{q}}$.

Obtaining the matrix $\boldsymbol{J}_{\boldsymbol{x}}$ requires the inversion of $\boldsymbol{\Phi}_{\partial \boldsymbol{q}}$ in (18). If the matrix is rank deficient, a singularity of type I is present. More critical are singularities of type II, where $\boldsymbol{\Phi}_{\partial \boldsymbol{q}}$ has full rank but $\boldsymbol{J}_{\boldsymbol{x}}^{-1}$ has not. This is caused by the structure of $\boldsymbol{\Phi}_{\partial \boldsymbol{x}}$ and the choice of actuated joints by $\boldsymbol{P}_{\boldsymbol{\theta}}$, cf. (Merlet, 2006).

4.2 Task Redundancy and Nullspace

The main difference between nullspace motion for serial and parallel robots is the existence of the full joint space different from the actuation joint space, i.e. $\boldsymbol{P}_{\boldsymbol{\theta}} \neq \boldsymbol{I}$. As discussed above, the nullspace projector for the full joint space only relies on the geometric matrix of the inverse kinematics $\boldsymbol{\Psi}_{\partial \boldsymbol{q}}$ and not on the Jacobian $\boldsymbol{J}_{\boldsymbol{x}}$. Therefore this coordinate space is especially suitable to generate desired motion although being in or near a singularity of type II. Further, it can be used to incorporate objective functions $h(\boldsymbol{q})$ e.g. for the joint limits of passive joints. Several cases can be distinguished:

4.2.1 General Case (Position-level IKP)

The general case is similar to section 3.2.1 for serial robots. This case is useful especially if $\|\boldsymbol{\Psi}\| \not\approx 0$, like in the position-level IKP. For parallel robots with objectives $h(\boldsymbol{q})$ only dependent on the joint coordinates it is identical to serial robots. For criteria $h(\boldsymbol{q}, \boldsymbol{x})$,

$$h_{\partial q_i} \approx (h(\boldsymbol{q} + \Delta \boldsymbol{q}, \boldsymbol{x}) - h(\boldsymbol{q}, \boldsymbol{x})) / \Delta q_i \quad (27)$$

is used, with $\Delta q_j = 0$ for $j \neq i$. This corresponds to the quotient from (9). No change of $\boldsymbol{x}$ is considered, since by $\|\boldsymbol{\Psi}\| \not\approx 0$ the kinematic loops of the robot are not closed with the platform. In the case of a full-mobility parallel robot, $n_{\boldsymbol{q}} = 36$ and therefore the $n_{\boldsymbol{q}}$+1 evaluations become computationally expensive.

4.2.2 Task Redundancy and Full Joint Space

In the case that the kinematic constraints are met, $\|\boldsymbol{\Psi}\| \approx 0$ holds and the preceding approach is not efficient. Nullspace motion and objective function are defined in the full joint space like $\boldsymbol{v}_{\boldsymbol{q}} = -(\partial h / \partial \boldsymbol{q})^{\mathrm{T}}$. The one-DoF task redundancy can be exploited by consideration of the elimination of components with the nullspace projector as in (13). The nullspace motion in operational space $\Delta \boldsymbol{x}$=$(\boldsymbol{0}^{\mathrm{T}}, \Delta \varphi_z)^{\mathrm{T}}$ results in the full joint space to $\Delta \boldsymbol{q} = \tilde{\boldsymbol{J}}_{\boldsymbol{x}}^{-1} \Delta \boldsymbol{x}$. With this, the numeric approximation of the gradient for task redundancy is

$$h_{\partial q_i}^{*} \approx (h(\boldsymbol{q} + \Delta \boldsymbol{q}, \boldsymbol{x} + \Delta \boldsymbol{x}) - h(\boldsymbol{q}, \boldsymbol{x})) / \Delta q_i. \quad (28)$$

The nullspace motion is obtained by the projector $\boldsymbol{N_q} = \boldsymbol{I}_{n_q} - \boldsymbol{\Psi}^{\#}_{\partial \boldsymbol{q}} \boldsymbol{\Psi}_{\partial \boldsymbol{q}} \in \mathbb{R}^{n_q \times n_q}$ from (21). The formulation requires only two evaluations of $h(\boldsymbol{q},\boldsymbol{x})$.

4.2.3 Task Redundancy and Actuation Space

Computing the nullspace motion in the actuation joint space with $\boldsymbol{\nu}_{\boldsymbol{\theta}} = -(\partial h/\partial \boldsymbol{\theta})^{\mathrm{T}}$ has the advantage of only using the projector $\boldsymbol{N}_{\boldsymbol{\theta}} = \boldsymbol{I}_{n_{\boldsymbol{\theta}}} - \boldsymbol{J}^{\#}_{\boldsymbol{y}} \boldsymbol{J}_{\boldsymbol{y}} \in \mathbb{R}^{n_{\boldsymbol{\theta}} \times n_{\boldsymbol{\theta}}}$ from (25) of much lower dimension than in the previous scheme, since e.g. $n_{\boldsymbol{\theta}}$=6 and $n_{\boldsymbol{q}}$=36 for fully parallel robots. The assumption $\|\boldsymbol{\Psi}\| \approx 0$ still has to hold.

The approach (14) from the serial robot case can be adapted to the parallel robot objective function $h(\boldsymbol{q},\boldsymbol{x})$, similar to the procedure leading to (28). Anticipating the nullspace projection, as a first step, the gradient is projected from the redundant coordinate φ_z into the actuated joint space with

$$h^{*}_{\partial \boldsymbol{\theta}} = \frac{\partial h}{\partial \varphi_z} \frac{\partial \varphi_z}{\partial \boldsymbol{\theta}}. \tag{29}$$

Numeric approximation of the first gradient yields

$$(\partial h/\partial \varphi_z) \approx (h(\boldsymbol{q}+\Delta \boldsymbol{q}, \boldsymbol{x}+\Delta \boldsymbol{x}) - h(\boldsymbol{q},\boldsymbol{x}))/\Delta \varphi_z. \tag{30}$$

The second term of (29) again is the last row of $\boldsymbol{J_x}$ which requires the inversion (23) and therefore a non-singular configuration of the robot. If the trajectory is generated for the full joint space, then the actuator nullspace motion $\ddot{\boldsymbol{\theta}}_{\mathrm{N}} = \boldsymbol{N}_{\boldsymbol{\theta}} \boldsymbol{\nu}_{\boldsymbol{\theta}}$ can be projected in that space by the time derivative of $\dot{\boldsymbol{q}}_{\mathrm{N}} = \tilde{\boldsymbol{J}}^{-1}_{\boldsymbol{x}} \boldsymbol{J}_{\boldsymbol{x}} \dot{\boldsymbol{\theta}}_{\mathrm{N}}$.

If the gradient $\partial h/\partial \boldsymbol{q}$ is available in analytic form, the projection can be performed by

$$\frac{\partial h}{\partial \boldsymbol{\theta}} = \frac{\partial h}{\partial \boldsymbol{q}} \frac{\partial \boldsymbol{q}}{\partial \boldsymbol{\theta}} = \frac{\partial h}{\partial \boldsymbol{q}} \tilde{\boldsymbol{J}}^{-1}_{\boldsymbol{x}} \boldsymbol{J}_{\boldsymbol{x}}. \tag{31}$$

This is helpful if e.g. the function for hyperbolic distance to the joint limits from (Zhu et al., 2013) is used.

5 NULLSPACE MOTION

The previous two sections review different modalities for nullspace motion of serial and parallel robots. The focus is on the necessary gradients of objective functions h in the case of task redundancy. The schemes (8), (21) and (25) are defined on acceleration level. This allows to generate a consistent and continuously differentiable trajectory of $\boldsymbol{q}(t)$, $\dot{\boldsymbol{q}}(t)$ and $\ddot{\boldsymbol{q}}(t)$, which is beneficial for the implementation for tracking controllers with feed-forward control with inverse dynamics, cf. (De Luca et al., 1992) and (Reiter et al., 2018). A general nullspace vector $\boldsymbol{\nu}$ is used in the equations of this paper, which allows incorporating different control architectures. A general architecture

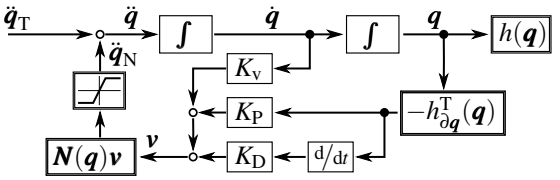


Figure 1: Nullspace motion controller scheme.

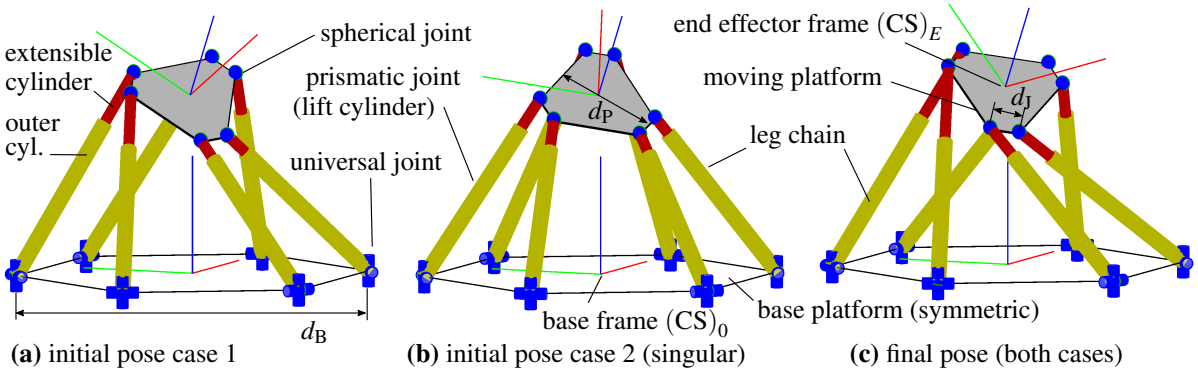
from (De Luca et al., 1992) is depicted in Figure 1. It uses a nullspace feedback loop with a PD controller (K_{P} and K_{D}) and an additional damping term (K_{V}). The architecture can be applied in all cases presented.

As stated in (Reiter et al., 2018), using the nullspace component in the differential kinematics on acceleration level is not equivalent to the derivative of the velocity-level solution. One approach is to perform analytic differentiation of the nullspace scheme, which works for serial robots up to higher orders (Reiter et al., 2018). Since this may be computational expensive for parallel robots, a discrete time implementation for the differentiation is used here, as suggested by (De Luca et al., 1992). Therefore the projection schemes on acceleration level of (8) and (21) do not fall into the non-equivalence category explained by (Reiter et al., 2018), if used as depicted in Figure 1.

Resolution of task redundancy on acceleration level with nullspace projection was introduced for parallel robots by (Agarwal et al., 2016), There only $\boldsymbol{\nu} = -K_{\mathrm{P}} h^{\mathrm{T}}_{\partial \boldsymbol{\theta}}$ is used, which corresponds to $K_{\mathrm{V}} = K_{\mathrm{D}} = 0$ in Figure 1. In that reference, a proof of stability of this control strategy was performed stating that the potential h is never increasing (in the minimization case). Due to the assumption $\dot{\boldsymbol{q}} = \boldsymbol{0}$ which was extended from the initial value to generality, the proof is only valid for $h(t{=}0)$, not for $h(t{>}0)$ and the statement can not be supported by the proof as shown next.

The nonlinear control system of Figure 1 can be described as a multi-input, multi-output system with nonlinear input gain $\boldsymbol{N}(\boldsymbol{q})$ and output function $h^{\mathrm{T}}_{\partial \boldsymbol{q}}(\boldsymbol{q})$ and a double integrator as internal dynamics, neglecting the task tracking input $\ddot{\boldsymbol{q}}_{\mathrm{T}}$ and the limitation of $\ddot{\boldsymbol{q}}_{\mathrm{N}}$. Already in the linear case, controlling a double integrator system only with proportional feedback leads to undamped oscillations. A stabilizing control is possible using a PD controller structure. This consideration leads to a use of all gains, $K_{\mathrm{V}}, K_{\mathrm{D}}, K_{\mathrm{P}} > 0$, as suggested in (De Luca et al., 1992).

The derivative gain K_{D} leads to high values for the nullspace acceleration when not already starting in an optimum of h – exactly like the velocity-based schemes this corresponds to. Therefore, an additional limit is added into the control scheme of Figure 1, with $\ddot{\boldsymbol{q}}_{\mathrm{N,max}} = \ddot{\boldsymbol{q}}_{\max} - \ddot{\boldsymbol{q}}_{\mathrm{T}}$. This presents an alternative to the successive consideration of maximum acceler-



(a) initial pose case 1 φ_z=0°, $\text{cond}(\boldsymbol{J}_{\boldsymbol{x}}^{-1})$=107.5

(b) initial pose case 2 (singular) φ_z=33.8°, $\text{cond}(\boldsymbol{J}_{\boldsymbol{x}}^{-1})=1.8\cdot10^5$

(c) final pose (both cases) φ_z=−25.0°, $\text{cond}(\boldsymbol{J}_{\boldsymbol{x}}^{-1})$=56.1

Figure 2: Task-redundant poses of the parallel robot of Sec. 6.1: (a) non-singular pose, (b) singular pose, (c) final value.

ations in the velocity-based schemes like (Santos and da Silva, 2017). Additional nullspace motion is generated to counteract when approaching limits $\boldsymbol{q}_{\max}$ of joint positions and $\dot{\boldsymbol{q}}_{\max}$ of joint velocities. Lower limits $\boldsymbol{q}_{\min}$ are handled in the same manner.

The amount and extent of objectives that can be included in the task redundancy resolution scheme is limited if only one degree of redundancy exists. Instead of a nested nullspace projection like in (Nakamura et al., 1987), only weighted sums of objectives can be used, which can be sufficient for properly designed objective functions as in (Zhu et al., 2013) or when prioritizing objectives, cf. (Lillo et al., 2019).

6 CASE STUDY

The validation of the nullspace projection of section 4 and the controller scheme of section 5 is performed in the following via simulation. The assumption of a five-DoF task and a six-DoF robot still holds. A parallel robot is used since the benefits and applicability of the proposed methods are better than for serial robots.

6.1 Singularity Avoidance in Nullspace

The nullspace motion is first examined without an additional robot task motion, i.e. $\dot{\boldsymbol{q}}_{\mathrm{T}} = \ddot{\boldsymbol{q}}_{\mathrm{T}} = \boldsymbol{0}$. The presented method can be applied to optimize any scalar performance criterion h in the nullspace of the task redundancy. In this example, the condition number of the robot Jacobian matrix of (23) is selected as $h(\boldsymbol{q},\boldsymbol{x}) = \text{cond}(\boldsymbol{J}_{\boldsymbol{x}}) = \text{cond}(\boldsymbol{J}_{\boldsymbol{x}}^{-1})$. Minimizing this criterion moves the manipulator towards low condition numbers and therefore away from singularities.

A hexapod parallel (Stewart-Gough type) robot is considered in a pose $\boldsymbol{y}$ with a high tilting angle of the platform. Technical details of the robot are given in Table 1. The high tilting angle (expressed by φ_x and φ_y) presents the general spatial case, where the representation of orientation by Cardan angles (*X-Y'-Z"*) is beneficial. The torsion angle φ_z of the platform is set to two different values, depicted in Fig 2 a and 2 b, representing the two cases to be investigated.

6.1.1 Case 1: Non-singular Initial Pose

In the first case the robot starts in the non-singular pose of Fig 2 a with $\varphi_{z,\text{start},1}$=0°. The values of the optimization objective h over the redundant coordinate φ_z are depicted in Figure 3 c over a full rotation of the platform (red dashed line). The overall performance of the robot is highly influenced by singularities, which are located at $\varphi_{z,\text{sing},1}\approx-65°$, $\varphi_{z,\text{sing},2}\approx34°$, $\varphi_{z,\text{sing},3}\approx64°$ and $\varphi_{z,\text{sing},4}\approx136°$. The locations of these singularities $\varphi_{z,\text{sing}}$ were obtained from global evaluation of the redundant coordinate.

The nullspace motion is first generated by the *position-level IK scheme* of (6). The convergence towards the next local minimum of $h(\varphi_{z,\text{final}})$=56.1 at $\varphi_{z,\text{final}}$=−25° can be comprehended from Fig 3 a for $h(t)$, from Fig 3 b for $\varphi_z(t)$ and from Fig 3 c for the characteristic diagram $h(\varphi_z)$ (blue lines). Compared to the initial value of $h(\varphi_{z,\text{start},1})$=107.5 it can be assumed that the overall performance of the robot is improved by the nullspace motion. This could be quantified by other, more physically motivated performance criteria such as accuracy, stiffness or actuator forces, but is out of scope of this publication. The position-level scheme only serves as a reference, since no time relation is included originally and the steps k of (6) are transferred to a time basis t using the joint velocity limits. A C^2-continuous trajectory (with continuously differentiable position and velocity) regarding limits for $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$ can not be assured by this.

Therefore, the nullspace motion is obtained using the *trajectory IK scheme* from (25) and section 4.2.3

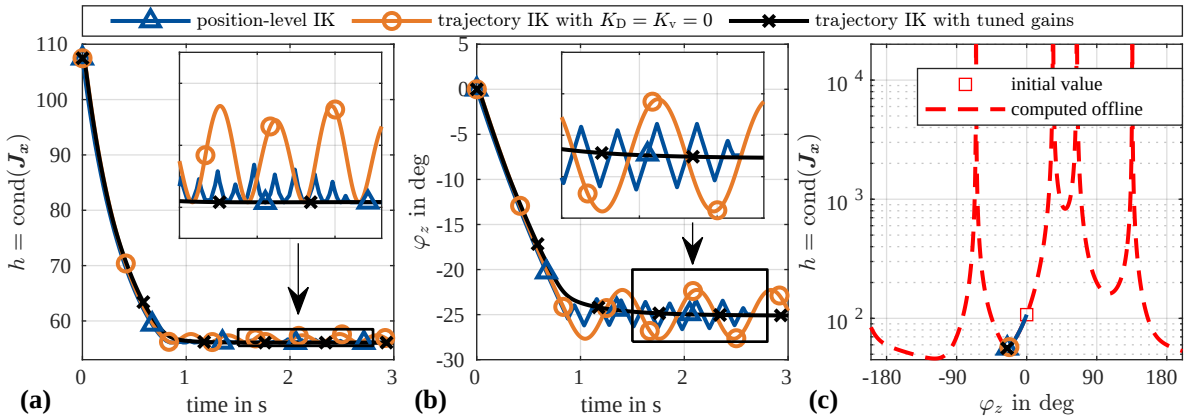


Figure 3: Simulation results of the parallel robot starting in the non-singular pose from Fig 2 a.

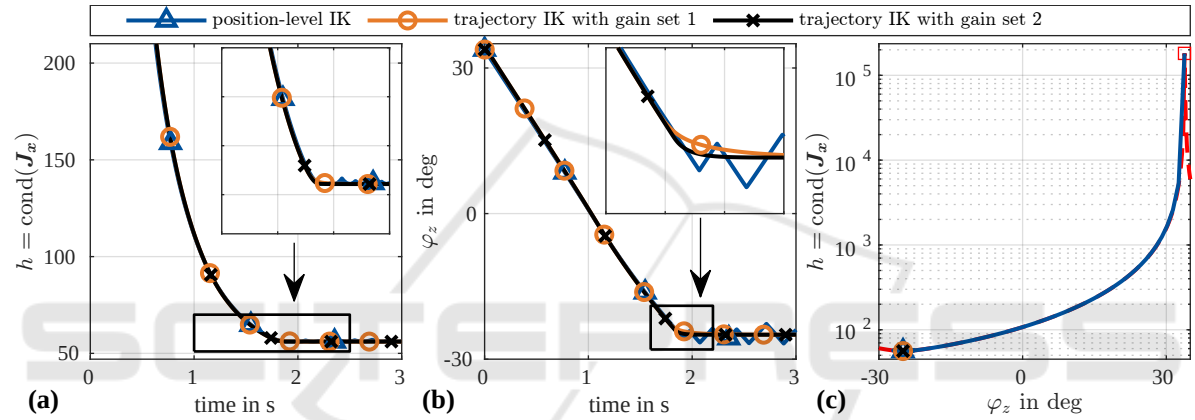


Figure 4: Simulation results of the parallel robot starting in the singular pose from Fig 2 b.

in the *actuator space*. To emphasize the argumentation of section 5, first the feedback gains $K_D = 0$ and $K_v = 0$ are used with $K_P = 1$. A discrete-time implementation with sample time 1 ms was used. The time evolution in Fig 3 a and 3 b shows undamped oscillations (orange lines). Therefore this parameterization does not allow to reach the purpose of moving to the next local optimum of h with little overshoot and fast convergence. To accomplish this control objective, the parameterization K_v=0.5, K_P=1 and K_D=0.5 is selected, which was obtained by manual tuning. The motion controller now reaches the optimal value of $\varphi_{z,\mathrm{final}} = -25°$ within about two seconds (black lines).

6.1.2 Case 2: Singular Initial Pose

The proposed new nullspace projection approach of the full joint coordinate space of (21) and section 4.2.2 has to be used if the initial pose or an intermediate pose represents a singular configuration. This is the case for the second starting pose $\varphi_{z,\mathrm{start},2}$=33.8°, where a numeric value for the condition number of $\mathrm{cond}(\boldsymbol{J}_x^{-1}) = 1.8 \cdot 10^5$ is reached. In this case, the inversion of $\boldsymbol{J}_x^{-1}$ is not possible (or at least does not produce feasible results). Controller schemes relying on $\boldsymbol{J}_x$, e.g. formulated in the operational space, would produce high accelerations or high actuator forces. Therefore a controller scheme avoiding this drawback has to be used as a fallback to physically operate the robot, e.g. a scheme defined only in the actuator space.

The motion controller of section 5 is again used to generate the nullspace motion with singularity avoidance, as in the previous case. This time, the motion is generated in the *full joint space* from (21) and section 4.2.2 with the *trajectory IK scheme*.

The simulation results are given in Fig 4. Again, the position-level IK and two parameterizations of the trajectory IK are shown, while the position-level IK only serves as comparison. The gains for the trajectory IK were tuned manually and using a particle swarm optimization and were set to K_v=0.03, K_D=0.01, and K_P=0.05 in the first set and K_v=0.2, K_D=0.03 and K_P=0.5 in the second set. With both sets again a fast convergence within two seconds can be achieved while maintaining the limits for $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$ and

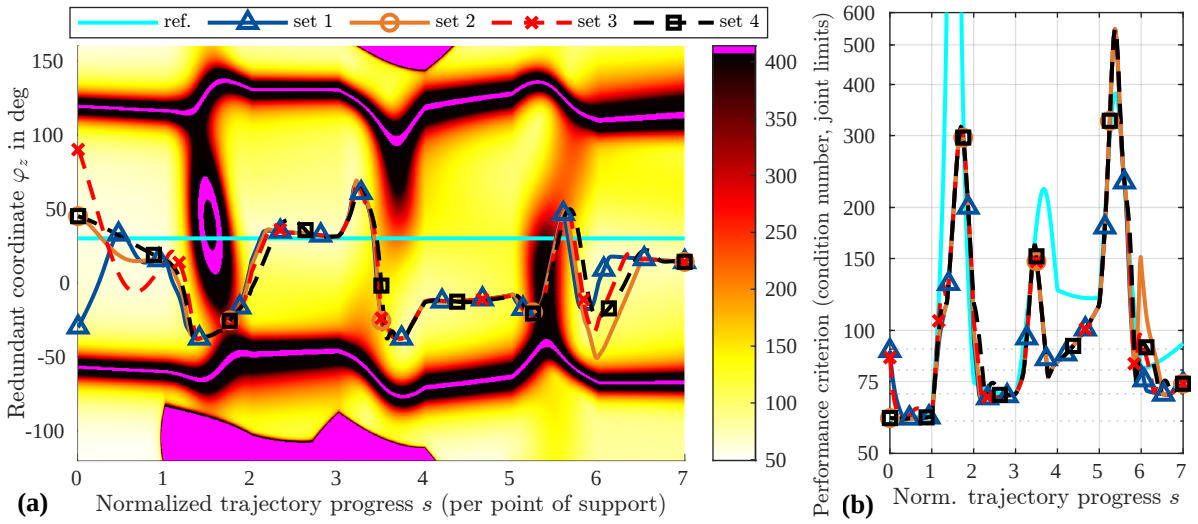


Figure 5: (a) Performance map of the condition number over the trajectory progress and all possibilities for the redundant coordinate. (b) Distribution of the condition number over the trajectory for different settings.

$\ddot{\boldsymbol{q}}$. The extent of oscillation and the settling time is influenced by the selected rather low joint velocity limits (see Table 1). This shows the feasibility of the proposed nullspace scheme in full joint coordinates to be able to leave a type-II singularity. The gain values above are not directly comparable to the ones of section 6.1.1, since there the compensation of the term $k(\boldsymbol{q})$ of (26) is attempted. The values in section 6.1.1 are divided by $k'(\boldsymbol{q},\boldsymbol{x}) = ||\tilde{\boldsymbol{J}}_{\boldsymbol{x}}^{-1}\boldsymbol{J}_{\boldsymbol{x}}||$, which presents a rough estimate. The determination of the exact relation is subject of ongoing work. Also, a general rule to determine the controller gains has to be found.

6.2 Trajectory with Singularity Avoidance

The previous evaluation of section 6.1 shows the general property of the task coordinate's nullspace to allow moving the manipulator away from singularities without an additional task motion. In a practical application like milling a pointing task trajectory $\boldsymbol{y}(t)$ with C^2 differentiability (or higher) is given. Therefore a task trajectory $\ddot{\boldsymbol{q}}_{\mathrm{T}}(t)$ according to (20) is active in the motion generation scheme summarized in Figure 1. Since milling presents one of the major use cases of hexapod parallel robots, an exemplary trajectory of milling a 45° bezel to a rectangular pocket is used for evaluation, as visualized in Figure 6. The red, green and blue axes around the rectangle represent the x-, y- and z-axis of the end effector frame for φ_z=0. Due to the task symmetry, only the pointing direction of the blue z-axes has to be aligned. The tilting angle of 45° poses high requirements on the robot and emphasizes the necessity of an appropriate spatial orientation representation. The trajectory is put together by rest-to-rest motions between eight poses $\boldsymbol{y}_i$ as points of support, summarized in Table 2 and corresponding to $s=0$ to $s=7$ in Figure 6. At each corner of the 250 mm×200 mm-rectangle a change of orientation is necessary to adjust the 45° angle for the next edge, corresponding to $1\leq s\leq 2$, $3\leq s\leq 4$ and $5\leq s\leq 6$ in Figure 6. A normalized trajectory coordinate s instead of the time t is used to represent progress along the path in the figures of this section for the sake of simplicity. Each rest-to-rest motion is assigned a range of one in s. However, the trajectory is generated by a trapezoidal acceleration profile (C^2 time-differentiable S-curve) with $\dot{r}_{\max}$=50 mm s^{-1}, $\dot{\varphi}_{\max}$=10°/s, a discrete sample time of 1 ms and acceleration and jerk times of 10 ms. The duration of the trajectory is 63.1 s.

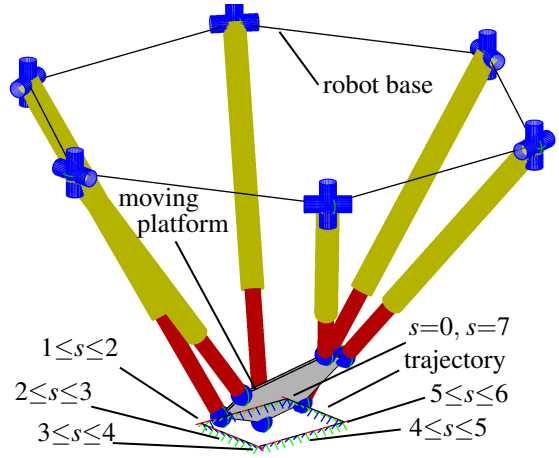


Figure 6: Starting pose of the robot trajectory for φ_z=0.

The global distribution of the condition number $h(\boldsymbol{q},\boldsymbol{x}) = \mathrm{cond}(\boldsymbol{J}_{\boldsymbol{x}})$ is shown as a performance map in

Figure 5 for the complete trajectory, represented by 1062 samples of the path coordinate s on the horizontal axis. The vertical axis is chosen as the redundant coordinate φ_z which was discretized in 361 samples in the range of $\pm 180^\circ$. The condition number is coded by the color of the map. High condition numbers $h > 400$ are saturated in the color code to dark red. Singularities, which are defined as condition numbers $h > 10^4$, and areas of joint limit violation are marked with magenta. Creating the colored performance map in this resolution requires 383 thousand evaluations of the inverse kinematics and the performance criterion. With this map a *global optimization* of the coordinate $\varphi_z(s)$ can be performed, which corresponds to the methods of the state of the art.

Using a *local optimization* on the contrary does not require the expensive computation of the performance map. The nullspace motion of the redundant coordinate φ_z is performed again by the nullspace controller scheme of section 5. Different settings for the nullspace controller are chosen for illustration and the resulting nullspace motion is shown as different lines in the performance map in Figure 5 a and as the evolution of the condition number over the progress of the trajectory in Figure 5 b. The settings are listed in Table 3. Using different starting values and settings emphasizes the robustness of the approach.

It has to be mentioned that the effects of the nullspace gains are limited by the velocity and acceleration limits of the joints (see Table 1). Therefore mainly in the last part of the trajectory a difference between the settings becomes visible. The overall behavior of avoiding areas of singularity is achieved with all settings. The reference is a constant rotation of the platform with φ_z=30° (cyan line in Figure 5). This presents a good solution at the beginning, but leads to a singularity during change of orientation at the second corner for $1 \leq s \leq 2$.

The first nullspace setting (set 1) in Figure 5 corresponds to the first setting in section 6.1.2. The other settings each have different gains. All nullspace settings produce a motion that avoids the first area of singularity at $s \approx 1.5$. Also a second area of high, but non-singular condition numbers at $s \approx 3.7$ is avoided. A third area of high condition numbers at $s \approx 5.4$ can not be avoided, since the shape of the performance map leads to a dead end at the previous section of low condition numbers at $s > 5$. Here the disadvantage of the local optimization becomes apparent and a global optimization or a component of looking ahead (for higher s) would outperform the proposed approach. The qualitative effect of the decreasing gains and increasing damping over the settings numbers can be seen at parts of the trajectory where a reconfiguration of the platform orientation is performed, e.g. at $s \approx 2.1$ and $s \approx 5.9$. For lower gains the optimal platform rotation is achieved later (in the sense of time t and therefore also of trajectory progress s).

6.3 Evaluation of Computation Time

The number of function evaluations required for the determination of the gradients of performance objectives obviously influences the computation time. The inverse kinematics algorithms on position and trajectory level were implemented in MATLAB as compiled stand-alone functions (using the MATLAB code generator and mex files). This combines fast iterations and efficient debugging with the MATLAB graphical user interface on the one hand and fast runtime of `C++` code on the other hand. The computation time was evaluated on a standard desktop computer (Intel Core i5-7500 CPU, 3.40 GHz, Linux operating system, generic kernel, no real-time). The hardware performance corresponds to typical PC-based robot controllers. Measured computation times should be regarded relative to each other and in their order of magnitude and may differ on other computing hardware.

Executing the *position-level IK* in the evaluation of section 6.1 requires 394 steps with the Newton-Raphson algorithm to compute an optimal pose which takes a mean computation time of 226 ms (n=50, σ=3.96 ms). The mean time per iteration is 0.57 ms (σ=0.01 ms). This corresponds to the blue lines in Figure 3. 555 iterations are necessary in section 6.1.2 with a similar mean time per iteration of 0.56 ms (n=50, σ=0.01 ms, blue lines in Figure 4).

If a variation in all joint coordinates is computed separately according to section 4.2.1, one iteration takes 1.78 ms (n=50, σ=0.04 ms). Despite this method being not applicable to the case of section 6.1, this shows the significance of using the gradient computation by the redundant coordinate's difference quotient from section 4.2.2.

The computation of the null space motion in the actuation space according to section 4.2.3 (and 6.1.1) instead of the full joint space of section 4.2.2 (and 6.1.2) does not improve the computation time of the *trajectory IK* as expected in the current implementation. A simulated time of 10 s and therefore 10,000 samples were computed to obtain the black and orange lines in Figures 3 and 4. Timing evaluation was performed with n=50 repetitions. In the actuated joint space (Figure 3) the computation time was 9.3 s (σ=52 ms, orange line, 0.93 ms per sample) and 9.7 s (σ=243 ms, black line). In the full joint space (Figure 4) the computation times were 9.1 s (σ=147 ms, orange line) and 9.2 s (σ=15 ms, black line). Using

the MATLAB profiler on the non-compiled function reveals that computing $\boldsymbol{N_q}$ of section 4.2.2 only takes about 3.5 % of the total time and $\boldsymbol{N_\theta}$ of section 4.2.3 only takes about 0.6 % of the total time. Performing the 6×6 inverse necessary for the latter approach only takes 0.15 %. The term $\boldsymbol{N_q}$ is also used for a nullspace motion upon joint position limit violation and is computed in all cases. This explains the higher computation times in the latter cases. Due to the small differences between the computation times the full coordinate space (section 4.2.2) should always be used within the current implementation of the algorithm.

The effort for computing the nullspace motion in general becomes visible when comparing the non-redundant case in section 6.2 (cyan line in Figure 5) to the redundant cases (other lines). The non-redundant trajectory takes 12.7 s (n=10, σ=23 ms, 0.20 ms per sample) while the redundant case takes 54.7 s (n=10, σ=139 ms, 0.87 ms per sample).

In summary it can be stated that despite the higher computational effort for the nullspace motion algorithms already the current implementation can be used as an online controller, since the runtime stays below 1 ms, which is the typical control loop sample time for such robots. The MATLAB functions can be directly used as blocks in SIMULINK which can be deployed with the EtherLab toolchain (using standard Ethernet hardware of a desktop computer with Linux realtime kernel as target and the EtherCAT bus as sensor/actuator interface). Runtime may further be improved by disabling computations with little effect and by using functions specific to the robot instead of the currently used general approach without limiting assumptions.

7 CONCLUSIONS AND OUTLOOK

A simplified computation of the gradients of performance criteria of robot manipulators is presented, which exploits the properties of task redundancy of degree one. This presents an extension to the application of redundancy resolution frameworks for serial and parallel robot manipulators. The performance criteria have to be evaluated under real-time conditions to allow their online optimization. Therefore, deployment of more sophisticated performance criteria will be possible by reducing the number of function evaluations necessary for gradient calculation.

With the presented nullspace formulation avoiding singularities for parallel robots is possible by performing the calculations in the full joint space. Using this more complex model allows to define a fallback controller for exiting a singularity of type II. This can be used if tracking a trajectory fails due to external disturbance or in case of manually guiding a parallel robot platform in teaching mode. In offline trajectory planning such as for machining tasks the method can be used to locally optimize path segments combined with global optimization of the whole path similar to Figure 5 of the paper.

Compared to the authors' earlier work, the redundancy resolution schemes of (Kotlarski et al., 2010) are extended for an efficient continuous adaption of a redundant degree of freedom instead of the previously favored discrete optimization. The need for computationally expensive global optimization decreases. The theoretical contribution (Schappler et al., 2019) regarding kinematic modeling is brought one step closer towards an application.

Future work comprises finding rules for setting gain parameters of the nullspace scheme, finding the analytic relation (26) between nullspace projectors of parallel robot actuator space and full joint space, and performing a stability analysis using control theory.

Table 1: Characteristic values of the parallel robot of Sec. 6.

Name	Symbol	Value	Unit
base diameter	d_B	1200	mm
platform diam.	d_P	400	mm
(in Sec. 6.2)	d_P	300	mm
joint pair distance	d_J	100	mm
platform position	r_x, r_y, r_z	50, 30, 600	mm
platf. orientation	φ_x, φ_y	$30°, -30°$	deg
act. j. stroke limits	$\theta_{i,\mathrm{min/max}}$	600, 1200	mm
pass j. angle lim.	—	not set	deg
act. j. velo. limits	$\dot{\theta}_{i,\mathrm{max}}$	2	$\mathrm{m\,s^{-1}}$
pass j. velo. lim.	—	45	deg/s
act. j. acc. limits	$\ddot{\theta}_{i,\mathrm{max}}$	20	$\mathrm{m\,s^{-2}}$
pass j. acc. lim.	—	1146	$\mathrm{deg/s^2}$

Table 2: Rest poses $\boldsymbol{y}_i$ of the trajectory in Figures 5 and 6.

s	r_x mm	r_y mm	r_z mm	φ_x deg	φ_y deg
0	-50	40	700	45	0
1	200	40	700	45	0
2	200	40	700	0	-45
3	200	-160	700	0	-45
4	200	-160	700	-45	0
5	-50	-160	700	-45	0
6	-50	-160	700	0	45
7	-50	40	700	0	45

Table 3: Settings for the nullspace motion of the trajectory in Figure 5 and Figure 6.

Param.	ref.	set 1	set 2	set 3	set 4
$\varphi_{z,0}$	30°	−30°	45°	90°	45°
K_V	—	0.03	0.1	0.25	0.5
K_P	—	0.05	0.01	0.005	0.0025
K_D	—	0.01	0.002	0.001	0.0005

ACKNOWLEDGEMENTS

The authors acknowledge the support by the Deutsche Forschungsgemeinschaft (DFG) under grant number 341489206. MATLAB Code to reproduce the results is available at GitHub under free license at https://github.com/SchapplM/robotics-paper_icinco2021.

REFERENCES

Agarwal, A., Nasa, C., and Bandyopadhyay, S. (2016). Dynamic singularity avoidance for parallel manipulators using a task-priority based control scheme. *Mechanism and Machine Theory*, 96:107–126.

Chiaverini, S., Oriolo, G., and Walker, I. D. (2008). Kinematically redundant manipulators. In *Springer Handbook of Robotics*, pages 245–268. Springer.

Corinaldi, D., Angeles, J., and Callegari, M. (2016). Posture optimization of a functionally redundant parallel robot. In *Advances in Robot Kinematics 2016*, pages 101–108. Springer.

De Luca, A., Oriolo, G., and Siciliano, B. (1992). Robot redundancy resolution at the acceleration level. *Laboratory Robotics and Automation*, 4:97–97.

Gao, Y., Chen, K., Gao, H., Xiao, P., and Wang, L. (2019). Small-angle perturbation method for moving platform orientation to avoid singularity of asymmetrical 3-RRR planner parallel manipulator. *Journal of The Brazilian Society of Mechanical Sciences and Engineering*, 41:1–18.

Gosselin, C. and Schreiber, L.-T. (2016). Kinematically redundant spatial parallel mechanisms for singularity avoidance and large orientational workspace. *IEEE Transactions on Robotics*, 32(2):286–300.

Gosselin, C. and Schreiber, L.-T. (2018). Redundancy in Parallel Mechanisms: A Review. *Applied Mechanics Reviews*, 70(1).

Guo, Y., Dong, H., and Ke, Y. (2015). Stiffness-oriented posture optimization in robotic machining applications. *Robotics and Computer-Integrated Manufacturing*, 27(2):367–376.

Huo, L. and Baron, L. (2008). The joint-limits and singularity avoidance in robotic welding. *Industrial Robot: An International Journal*, 35(5):456–464.

Kotlarski, J., Do Thanh, T., Heimann, B., and Ortmaier, T. (2010). Optimization strategies for additional actuators of kinematically redundant parallel kinematic machines. In *2010 IEEE International Conference on Robotics and Automation (ICRA)*, pages 656–661.

Žlajpah, L. (2017). On orientation control of functional redundant robots. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2475–2482.

Léger, J. and Angeles, J. (2016). Off-line programming of six-axis robots for optimum five-dimensional tasks. *Mechanism and Machine Theory*, 100:155–169.

Lillo, P. D., Chiaverini, S., and Antonelli, G. (2019). Handling robot constraints within a set-based multi-task priority inverse kinematics framework. In *2019 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7477–7483.

Luces, M., Mills, J. K., and Benhabib, B. (2017). A review of redundant parallel kinematic mechanisms. *Journal of Intelligent & Robotic Systems*, 86:175–198.

Merlet, J.-P. (2006). Jacobian, manipulability, condition number, and accuracy of parallel robots. *Journal of Mechanical Design*, 128(1):199–206.

Merlet, J.-P., Perng, M.-W., and Daney, D. (2000). Optimal trajectory planning of a 5-axis machine-tool based on a 6-axis parallel manipulator. In *Advances in Robot Kinematics*, pages 315–322. Springer.

Mousavi, S., Gagnol, V., Bouzgarrou, B. C., and Ray, P. (2018). Stability optimization in robotic milling through the control of functional redundancies. *Robotics and Computer-Integrated Manufacturing*, 50:181–192.

Nakamura, Y., Hanafusa, H., and Yoshikawa, T. (1987). Task-priority based redundancy control of robot manipulators. *The International Journal of Robotics Research*, 6(2):3–15.

Oen, K.-T. and Wang, L.-C. T. (2007). Optimal dynamic trajectory planning for linearly actuated platform type parallel manipulators having task space redundant degree of freedom. *Mechanism and Machine Theory*, 42(6):727–750.

Ozgoren, M. K. (2013). Optimal inverse kinematic solutions for redundant manipulators by using analytical methods to minimize position and velocity measures. *Journal of Mechanisms and Robotics*, 5(3).

Reiter, A., Müller, A., and Gattringer, H. (2018). On higher order inverse kinematics methods in time-optimal trajectory planning for kinematically redundant manipulators. *IEEE Transactions on Industrial Informatics*, 14(4):1681–1690.

Santos, J. C. and da Silva, M. M. (2017). Redundancy resolution of kinematically redundant parallel manipulators via differential dynamic programing. *Journal of Mechanisms and Robotics*, 9(4).

Schappler, M., Tappe, S., and Ortmaier, T. (2019). Modeling parallel robot kinematics for 3T2R and 3T3R tasks using reciprocal sets of Euler angles. *MDPI Robotics*, 8(3). Open access journal. Online: https://doi.org/10.3390/robotics8030068.

Shaw, D. and Chen, Y.-S. (2001). Cutting path generation of the stewart-platform-based milling machine using an end-mill. *International Journal of Production Research*, 39(7):1367–1383.

Smirnov, V., Plyusnin, V., and Mirzaeva, G. (2013). Energy efficient trajectories of industrial machine tools with parallel kinematics. In *2013 IEEE International Conference on Industrial Technology (ICIT)*, pages 1267–1272.

Zhu, W., Qu, W., Cao, L., Yang, D., and Ke, Y. (2013). An off-line programming system for robotic drilling in aerospace manufacturing. *The International Journal of Advanced Manufacturing Technology*, 68(9-12):2535–2545.