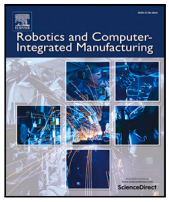
Robotics and Computer–Integrated Manufacturing 82 (2023) 102516

Contents lists available at ScienceDirect

Robotics and Computer-Integrated Manufacturing

journal homepage: www.elsevier.com/locate/rcim



Full length article

Optimization-based path planning framework for industrial manufacturing processes with complex continuous paths☆

Thomas Weingartshofer [a,*], Bernhard Bischof [b], Martin Meiringer [a], Christian Hartl-Nesic [a], Andreas Kugi [a,b]

[a] Automation and Control Institute, TU Wien, 1040 Vienna, Austria
[b] Center for Vision, Automation & Control, AIT Austrian Institute of Technology GmbH, 1210 Vienna, Austria

ARTICLE INFO

Keywords:
Motion planning of continuous paths
Industrial robots
Manufacturing process
Manufacturing tolerances
Motion constraints
Optimization-based approach

ABSTRACT

The complexity of robotic path planning problems in industrial manufacturing increases significantly with the current trends of product individualization and flexible production systems. In many industrial processes, a robotic tool has to follow a desired manufacturing path most accurately, while certain deviations, also called process tolerances and process windows, are allowed. In this work, a path planning framework is proposed, which systematically incorporates all process degrees of freedom (DoF), tolerances and redundant DoF of the considered manufacturing process as well as collision avoidance. Based on the specified process DoF and tolerances, the objective function and the hard and soft constraints of the underlying optimization problem can be easily parametrized to find the optimal joint-space path. By providing the analytical gradients of the objective function and the constraints and utilizing modern multi-core CPUs, the computation performance can be significantly improved. The proposed path planning framework is demonstrated for an experimental drawing process and a simulated spraying process. The planner is able to solve complex planning tasks of continuous manufacturing paths by systematically exploiting the process DoF and tolerances while allowing to move through singular configurations, which leads to solutions that cannot be found by state-of-the-art concepts.

1. Introduction

Current trends in industry show that the number of employed industrial robots increases significantly [1], the product diversity advances up to full individualization [2] and the demand for flexible production systems raises tremendously [3]. In automated manufacturing processes, the motions of industrial robots are mostly planned using CAD-based offline path planning approaches [4], which have to keep pace with the recent developments in terms of flexibility, complexity and computational performance. Frequently implemented automated industrial manufacturing processes include welding, spray painting, milling, sanding, polishing and chamfering. Executing a manufacturing process often requires a robotic tool to follow a given manufacturing path most accurately. If the requirements of the manufacturing process exceed the capabilities of the robotic system or the planning algorithm, the manufacturing paths have to be adapted [5], the robot has to be positioned differently relative to the workpiece [6], a different robot has to be used, or the robotic tool has to be adapted [7], which are laborious procedures in general. Alternatively, incorporating the specific properties of the executed process into the path planning has the potential to significantly increase the flexibility of a given robot configuration.

The Cartesian degrees of freedom (DoF) of the tool, which are needed to accomplish the manufacturing task, see [8], are called *process DoF* in the following. Most processes demand all six DoF from the robot, e.g. cutting and sewing, while some processes only require five DoF. The remaining DoF is referred to as a *redundant process DoF* in this work. For example, a rotating tool like a drill or a polishing disk is considered as redundant process DoF. Furthermore, some processes allow for deviations from the manufacturing path to some extent. In the following, the term *process window* refers to allowed tool deviations from the manufacturing path, which do not degrade the process quality, see, e.g., [8]. On the other hand, deviations which degrade the process quality to a tolerable extent, but should be avoided if possible, will be denoted *process tolerances*. The process tolerances may be constrained additionally by defining a *tolerance band*. In the remainder of this

☆ The authors acknowledge TU Wien Bibliothek, Austria for financial support through its Open Access Funding Programme.
* Corresponding author.
E-mail address: weingartshofer@acin.tuwien.ac.at (T. Weingartshofer).

https://doi.org/10.1016/j.rcim.2022.102516
Received 26 July 2022; Received in revised form 5 December 2022; Accepted 18 December 2022

work, the generic term *process properties* encompasses process tolerances, process windows, constraints and redundant process DoF. A manufacturing process is specified by a set of process properties in one or multiple process DoF, i.e. allowing position deviations in specific Cartesian directions or orientation deviations w.r.t. specific axes. For example, in a polishing process, the orientation may deviate slightly from the surface normal, but the tool position should exactly follow the desired path. Additionally, the rotating disk of the polishing tool represents a redundant process DoF. In a spraying process, the spray nozzle has to perfectly align with the surface normal vector and the manufacturing path, but the distance of the spray nozzle to the surface may vary within a certain tolerance band.

In the literature, several path planning algorithms tailored to specific manufacturing processes have been published, which incorporate only the application-specific process properties, e.g. welding [9,10], surface inspection [11], sanding [12] and chamfering [13]. In this work, a general deterministic path planning framework is proposed, which systematically considers all process properties. Additionally, the planning algorithm takes into account collision avoidance and is capable of planning through singular joint configurations. By incorporating process tolerances, windows and redundant process DoF, the path planning framework is able to compute continuous robot motions for manufacturing paths which exceed the kinematic capabilities of the robot. These advanced planning abilities enable the algorithm to handle industrial path planning problems of high complexity and with significant flexibility demands.

2. Literature review and contribution

The state of the art of path planning algorithms for industrial robots are summarized and discussed in this section. An emphasis is laid on path planning methods which consider tolerances, process windows, constraints and redundant DoF.

Note that this review focuses on path planning in joint space for a given Cartesian tool path. This tool path may be designed manually or be the result of an automatic path generation algorithm based on the workpiece geometry, e.g. [14]. In many existing works, algorithms tailored to an industrial process are proposed, e.g. spray painting [15,16] or polishing [17]. Path planning algorithms are distinguished by the underlying method, i.e. sampling based or optimization based, which are discussed in the following. Additionally, the pathwise inverse kinematics problem is addressed as a special form of the inverse kinematics problem. At the end of this section, the discussed methods are compared and the main contribution of this work is presented.

2.1. Sampling-based path planners

A detailed survey of sampling-based path planners which systematically incorporate constraints is given in [18]. The most prominent library of sampling-based path planners is the *Open Motion Planning Library (OMPL)* [19], in which several sampling-based approaches are included, such as *Probabilistic Roadmap Methods (PRM)* and *Rapidly-exploring Random Trees (RRT)*. As the main focus of those approaches is solving point-to-point motion planning problems, the intermediate Cartesian path cannot be provided beforehand. Furthermore, sampling-based path planners mostly require smoothing as a post-processing step to remove redundant and jerky motions, see, e.g., [20]. Additionally, including multiple constraints, redundant DoF and tolerances in the computation of a joint-space path increases the problem size drastically, which becomes infeasible to solve.

The popular *Descartes* algorithm [21] is a semi-constrained offline path planner which considers tolerances. Therein, for every Cartesian path point, multiple inverse kinematic solutions are computed using inverse kinematics solvers such as the *Kinematics and Dynamics Library (KDL)* [22] or *Inverse Kinematics Fast (IKFast)* [23]. Process tolerances are considered by sampling the allowed tolerance band with an equidistant grid, for which all corresponding inverse kinematic solutions are computed. Hence, the number of possible inverse kinematic solutions for each path point increases, in particular if tolerances are allowed for multiple DoF. This leads to high memory usage and high computation time while searching the best joint-space path in a graph search with the Dijkstra algorithm. In [24], an RRT-based planner is used to compute collision-free paths without an explicit goal configuration. Instead, a goal region is described by workspace goal criteria and a path with lowest tolerance utilization is computed. In general, the consideration of multiple constraints and costs in sampling-based methods is challenging, especially if optimal paths have to be found, see [25]. In [26], a sampling-based planner is introduced to solve point-to-point problems. The planner incorporates kinematic and dynamic constraints in a framework to find optimal paths.

In many works, path planners tailored to a special industrial process are designed, e.g. for welding applications in [10]. Therein, the redundant process DoF of the tool is discretized and a collision-free path is generated by a modified RRT* algorithm. Another example is presented in [9], where the planning for a complex welding path for a 6-DoF robot is performed in two steps. First, the joint configurations of the start and end poses of the end-effector are determined. A pose comprises the Cartesian position and orientation coordinates. Second, the continuous robot motion between those configurations is computed. In this process, the rotation around the welding torch is a redundant process DoF. This DoF is sampled with an equidistant grid and the corresponding joint configurations are computed. If no solution of the inverse kinematics problem is found, a different inclination angle of the torch is used. Furthermore, the joint movement defines costs to be minimized by the beam search algorithm while near-singular configurations of the robot are avoided. The above path planners [9,10] sample the redundant DoF and therefore need to reduce the search space by sampling in order to solve the planning problem.

2.2. Optimization-based path planners

In optimization-based path planners, an optimization scheme is employed to plan feasible and locally or globally optimal joint-space paths. Constraints can be incorporated systematically. A sequential convex optimization algorithm called *Trajectory Optimization for Motion Planning (Trajopt)* is presented in [27]. The *Trajopt* algorithm guesses one or multiple initial solutions for a trajectory and then optimizes the joint-space path length while considering kinematic constraints. Process tolerances are not taken into account by this algorithm. The *covariant Hamiltonian optimization for motion planning (CHOMP)* algorithm in [28] is used to find smooth collision-free trajectories and the *stochastic trajectory optimization for motion planning (STOMP)* algorithm in [29] is a gradient-free optimization with the possibility to include constraints. The above path planners compute an intermediate path between two points and therefore are suited for solving point-to-point planning problems only.

Optimization-based approaches are capable of considering additional criteria for path planning, i.e. energy consumption [30,31], time [32–34], time and jerk [35,36] and a combination of multiple criteria [37]. The dynamic model of the robot is taken into account in, e.g., [34,38]. Additionally, the path optimization can also be executed in a post-processing step. For example, in [39] the joint-space path length is minimized from an initial guess.

A robotic layup task is presented in [8] for a multi-robot scenario. The specific process window for this task is utilized to generate robot trajectories. In particular, the redundant process DoF of the roller tool is discretized and an optimization-based approach is used to solve the trajectory planning problem while considering process constraints and avoiding singular configurations.

2.3. Inverse kinematics and pathwise inverse kinematics

A general review of methods to compute the inverse kinematics is given in [40]. In [41], a continuous multivariate inverse function is described, which is used to find the global inverse kinematics of redundant manipulators. The global inverse kinematics computes a unique joint configuration for a given end-effector pose also for redundant manipulators. The *TRAC-Inverse Kinematics (TRAC-IK)* solver, proposed in [42], considers tolerances on each Cartesian dimension and improves the calculation times compared to *KDL* [22]. Another widely used method to solve the inverse kinematics is to formulate an optimization problem, see, e.g., [43,44]. In [45], a genetic algorithm to solve inverse kinematics problems is proposed, where different objectives of the manipulator are weighted in an optimization problem as soft constraints.

Pathwise inverse kinematics refers to applying an inverse kinematics scheme to a Cartesian end-effector path in order to find one or more corresponding joint-space paths. Solving the pathwise inverse kinematics problem by computing the inverse kinematics for each path point independently, in general leads to discontinuous joint-space paths. An example for an optimization-based scheme is presented in [46]. The *trajectory optimization of a redundant manipulator (TORM)* in [46] computes trajectories for given end-effector paths. A two-stage gradient descent optimization is used to minimize the position and orientation error of the joint-space path w.r.t. the desired end-effector poses. Thereby, new trajectories are generated repeatedly to avoid local minima, however, the process properties of the underlying manufacturing process cannot be considered systematically. Another work which is concerned with solving pathwise inverse kinematics problems is given in [47], where an anytime graph-based path planner minimizes a geometric task-space criterion, the so-called Fréchet distance. This Fréchet distance can be interpreted as utilized process tolerance. In [48], the inverse kinematics problem is solved with geometric task-prioritization, if the desired motion cannot be executed by the robot. In the path planner proposed in [48], the tolerances of the desired geometric task are implemented as soft constraints. Hence, no guarantees in terms of maximum deviations can be given.

The online path planner *Relaxed Inverse Kinematics (RelaxedIK)* [49] computes continuous joint-space paths as a sequence of optimization problems, where the solution of the previous path point is used as initial guess for the subsequent optimization problem. The position and orientation deviations from the given path are weighted in the objective function as soft constraints. Based on *RelaxedIK*, the path planner proposed in [50] focuses on offline path planning and considers process tolerances. Starting at multiple joint configurations for the first path point, continuous joint-space paths are computed, which account for the tolerances of the end-effector pose. Although self collisions are detected, no general collision avoidance for obstacles in the workspace is considered. In both works [49,50], singular configurations are avoided, but it cannot be guaranteed that the resulting path adheres to all tolerance bands and process windows.

2.4. Comparison, contribution and outline

Sampling-based path planners are widely used for path planning of industrial processes in spatially constrained environments. By considering (redundant) process DoF, the search space increases exponentially and cannot be covered sufficiently and smoothly using sampling. Pathwise inverse kinematics solvers are based on the inverse kinematics solution of robots, for which every (redundant) process DoF has to be discretized to find continuous robot movements. In contrast, in optimization-based path planners these additional DoF can be considered as continuous optimization variables and additional constraints (e.g. process windows and bands) and criteria (e.g. time or energy optimality) can be incorporated.

In this work, an optimization-based path planner with a general framework to systematically incorporate process tolerances, process windows, constraints and redundant process DoF is proposed. In the following, industrial processes are considered which are described by a geometric manufacturing path on the workpiece and a set of process properties. The incorporation of these process properties into the path planning algorithm allows to deviate from the given exact Cartesian manufacturing path in the allowed process DoF. This is suitable for many applications including, e.g., spraying, grinding and welding. By considering the process properties and thus by exploiting the process windows and tolerance bands, the robot is able to realize a much larger number of feasible manufacturing paths. In a second step, the found joint-space solutions are evaluated and the one with the best manufacturing quality is chosen.

The main contribution of the proposed work is the systematic integration of process properties in the optimization-based path planning algorithm as a general framework of equality and inequality constraints, which is not available in state-of-the-art path planners. In this way, desired manufacturing process properties in the task space can be guaranteed to hold while other DoF are allowed to vary within given process windows or tolerance bands. Due to the optimization-based approach, no analytical inverse kinematics formulation is required, which is especially useful for kinematically redundant manipulators. Furthermore, analytical gradients of the objective function and all equality and inequality constraints are used to significantly improve the calculation time and convergence speed of the path planner. The proposed path planner also incorporates collision avoidance methods, is able to plan through kinematic robot singularities and uses parallelization on multiple CPU cores.

Remark 1. It is worth mentioning that the proposed path planner is also advantageous for non-redundant robots performing non-redundant processes. Even for non-redundant robots, like industrial robots with 6 DoF and a typical kinematics, e.g., Kuka Cybertech KR8 R1620 [51], the inverse kinematics is not unique and up to 16 different solutions are obtained for a given pose. Further non-uniqueness has to be considered, if one or more axes have a large or infinite turning range, i.e. robots with axis ranges of more than $\pm 360°$. All possible solutions for the inverse kinematics can be handled with the proposed optimization-based path planner.

This paper is organized as follows. In Section 3, the notation and mathematical model of a general robot is introduced. Then, in Section 4, manufacturing paths and process properties of a manufacturing process are mathematically defined. The proposed path planning framework is described in detail in Section 5. In Section 6, the path planner is applied to two different manufacturing processes. The work is concluded in Section 7.

3. Mathematical model

In this section, the mathematical definitions of general robot kinematics are given. The forward kinematics of the end-effector frame $\mathcal{E}$ w.r.t. the robot base frame $\mathcal{B}$ of a robot with n DoF is given by, see, e.g., [52]

$$\begin{bmatrix} \mathbf{p}_{\mathcal{B}}^{\mathcal{E}} \\ \mathbf{o}_{\mathcal{B}}^{\mathcal{E}} \end{bmatrix} = \mathbf{h}(\mathbf{q}) \quad , \tag{1}$$

where the joint configuration $\mathbf{q}^{\mathrm{T}} = \begin{bmatrix} q_1 \cdots q_n \end{bmatrix}$ contains the generalized coordinates of the robot, i.e. positions of prismatic joints and angles of revolute joints. The vector on the left-hand side of (1) represents a pose consisting of the Cartesian position vector $\mathbf{p}^{\mathrm{T}} = \begin{bmatrix} x & y & z \end{bmatrix}$ and the orientation expressed as unit quaternion $\mathbf{o}^{\mathrm{T}} = \begin{bmatrix} \eta & \boldsymbol{\epsilon}^{\mathrm{T}} \end{bmatrix}$, with the

scalar part η and the vector part $\boldsymbol{\epsilon}$. Similarly, the pose (1) can also be written as homogeneous transformation

$$\mathbf{H}_{\mathcal{B}}^{\mathcal{E}} = \begin{bmatrix} \mathbf{R}_{\mathcal{B}}^{\mathcal{E}} & \mathbf{p}_{\mathcal{B}}^{\mathcal{E}} \\ \mathbf{0} & 1 \end{bmatrix} \quad , \tag{2}$$

where $\mathbf{R}_{\mathcal{B}}^{\mathcal{E}}$ denotes the rotation matrix associated with the quaternion $\mathbf{o}_{\mathcal{B}}^{\mathcal{E}}$ in (1). Note that the notation $(\cdot)_{\mathcal{A}}^{\mathcal{B}}$ refers to mathematical objects describing the geometric relation of the frame $\mathcal{B}$ w.r.t. $\mathcal{A}$, expressed in $\mathcal{A}$. The instantaneous velocity related to (1) of the end-effector frame $\mathcal{E}$ w.r.t. the robot base frame $\mathcal{B}$ is given by, see [52]

$$\begin{bmatrix} \dot{\mathbf{p}}_{\mathcal{B}}^{\mathcal{E}} \\ \boldsymbol{\omega}_{\mathcal{B}}^{\mathcal{E}} \end{bmatrix} = \begin{bmatrix} \mathbf{J}_{\mathcal{B},v}^{\mathcal{E}}(\mathbf{q}) \\ \mathbf{J}_{\mathcal{B},\omega}^{\mathcal{E}}(\mathbf{q}) \end{bmatrix} \dot{\mathbf{q}} \quad , \tag{3}$$

with the geometric Jacobian $\mathbf{J}_{\mathcal{B},v}^{\mathcal{E}}$ related to the linear velocity $\dot{\mathbf{p}}_{\mathcal{B}}^{\mathcal{E}}$, the geometric Jacobian $\mathbf{J}_{\mathcal{B},\omega}^{\mathcal{E}}$ related to the angular velocity $\boldsymbol{\omega}_{\mathcal{B}}^{\mathcal{E}}$ and the joint velocity $\dot{\mathbf{q}}$.

4. Manufacturing process

In this work, a manufacturing process is specified by the manufacturing paths on the workpiece together with the process properties which incorporate process DoF, redundant DoF, constraints, process tolerances and process windows. These concepts are defined mathematically in this section. Subsequently, the general collision detection method *Voronoi-clip (V-Clip)* [53] is briefly introduced and collision models in the robot environment are defined.

4.1. Manufacturing path

A manufacturing path is given as a sequence of poses $\mathcal{H}_{\mathcal{P}}^{\mathcal{M}} = \{\mathbf{H}_{\mathcal{P},i}^{\mathcal{M}}, i = 1,\dots,m\}$, where each pose $\mathbf{H}_{\mathcal{P},i}^{\mathcal{M}}, i = 1,\dots,m$, describes the manufacturing frame $\mathcal{M}$ w.r.t. the workpiece frame $\mathcal{P}$, i.e. the desired tool pose during process execution. Hence, the manufacturing process is executed by moving the tool, described by the tool center point (TCP) $\mathcal{T}$, along the manufacturing path $\mathcal{H}_{\mathcal{P}}^{\mathcal{M}}$ within the allowed tolerance bands and process windows and considering redundant process DoF. The manufacturing paths are assumed to be given and may be generated using CAD or automatic path generation algorithms based on the workpiece geometry. Note that the manufacturing path is assumed to be sufficiently smooth to generate continuous joint-space paths. Additionally, the spatial resolution is chosen based on the desired accuracy of the manufacturing result.

4.2. Manufacturing tool

Let $\mathbf{H}_{\mathcal{P}}^{\mathcal{T}}$ denote the pose of the TCP frame $\mathcal{T}$ w.r.t. the workpiece frame $\mathcal{P}$. Depending on the manufacturing process, the tool may be mounted on the end-effector of the robot while the workpiece is stationary or the tool may be stationary in the world frame $\mathcal{W}$ while the workpiece is mounted on the end-effector. Hence, depending on the mounting of the tool, the transformation $\mathbf{H}_{\mathcal{P}}^{\mathcal{T}}$ is specified differently, as discussed in the following, see Fig. 1.

4.2.1. Tool on end-effector

When the tool is mounted on the end-effector, the pose of the TCP frame $\mathcal{T}$ w.r.t. the end-effector frame $\mathcal{E}$ is given by the homogeneous transformation $\mathbf{H}_{\mathcal{E}}^{\mathcal{T}}$. Additionally, the stationary poses of the robot base $\mathcal{B}$ and the workpiece $\mathcal{P}$ w.r.t. the world frame $\mathcal{W}$ are specified by $\mathbf{H}_{\mathcal{W}}^{\mathcal{B}}$ and $\mathbf{H}_{\mathcal{W}}^{\mathcal{P}}$, respectively. Consequently, the pose $\mathbf{H}_{\mathcal{P}}^{\mathcal{T}}$ of the manufacturing system is computed in the form

$$\mathbf{H}_{\mathcal{P}}^{\mathcal{T}}(\mathbf{q}) = (\mathbf{H}_{\mathcal{W}}^{\mathcal{P}})^{-1}\mathbf{H}_{\mathcal{W}}^{\mathcal{B}}\mathbf{H}_{\mathcal{B}}^{\mathcal{E}}(\mathbf{q})\mathbf{H}_{\mathcal{E}}^{\mathcal{T}} \quad , \tag{4}$$

which may be understood as an augmented forward kinematics of the tool $\mathcal{T}$ w.r.t. the workpiece $\mathcal{P}$, see Fig. 1a.

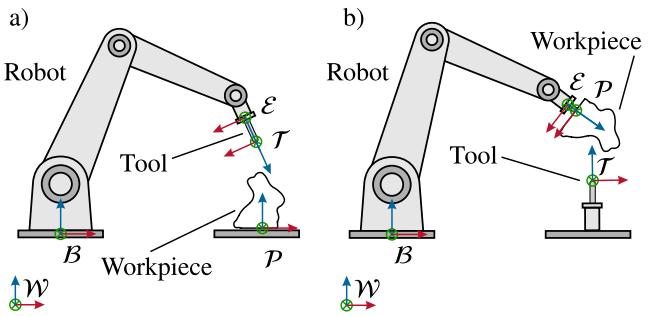


Fig. 1. Kinematics of the robot: (a) Tool on end-effector, (b) Tool stationary.

4.2.2. Stationary tool

If the tool is stationary, the workpiece is mounted on the end-effector flange. Thereby, the pose $\mathbf{H}_{\mathcal{P}}^{\mathcal{T}}$ of the manufacturing system is

$$\mathbf{H}_{\mathcal{P}}^{\mathcal{T}}(\mathbf{q}) = \mathbf{H}_{\mathcal{P}}^{\mathcal{E}}(\mathbf{H}_{\mathcal{B}}^{\mathcal{E}}(\mathbf{q}))^{-1}\mathbf{H}_{\mathcal{B}}^{\mathcal{W}}\mathbf{H}_{\mathcal{W}}^{\mathcal{T}} \quad , \tag{5}$$

utilizing the known geometric relations between the workpiece frame $\mathcal{P}$, the end-effector frame $\mathcal{E}$, the TCP frame $\mathcal{T}$, the world frame $\mathcal{W}$ and the robot base frame $\mathcal{B}$, described by $\mathbf{H}_{\mathcal{P}}^{\mathcal{E}}$, $\mathbf{H}_{\mathcal{W}}^{\mathcal{T}}$ and $\mathbf{H}_{\mathcal{B}}^{\mathcal{W}}$, see Fig. 1b.

4.3. Process DoF and process properties

Many industrial processes allow for process tolerances or process windows in one or more process DoF during process execution. Hence, the tool frame $\mathcal{T}$ may deviate from the manufacturing frame $\mathcal{M}$ to a certain predefined extent without deteriorating the quality of the process. For example in a robotic ultrasonic cutting process, the position of the knife must follow the given manufacturing path exactly while the knife tilt may vary within a certain process window, defined by an orientation cone, see Fig. 2a. In contrast, in a spray painting process, the distance to the surface of the spray object should stay within a given tolerance band while the lateral position coordinates must match the manufacturing frame $\mathcal{M}$ exactly. Furthermore, if a rotationally symmetric spray jet is used, the manufacturing process becomes invariant to the rotation of the spray jet around the surface normal vector. Thus, in this case a redundant DoF is present in the process, see Fig. 2b. The proposed path planner systematically considers all process tolerances and process DoF. Taking this information into account, feasible and optimal robot motions can be computed, where otherwise no solution would be found. The process windows or tolerance bands of the process DoF are specified by the minimum and maximum allowed displacements, see Fig. 3a

$$\mathbf{d}_{\min} = \begin{bmatrix} d_{x,\min} \\ d_{y,\min} \\ d_{z,\min} \end{bmatrix}, \qquad \mathbf{d}_{\max} = \begin{bmatrix} d_{x,\max} \\ d_{y,\max} \\ d_{z,\max} \end{bmatrix} \tag{6}$$

and the minimum and maximum rotations in terms of roll–pitch–yaw angles, see Fig. 3b

$$\boldsymbol{\phi}_{\min} = \begin{bmatrix} \phi_{x,\min} \\ \phi_{y,\min} \\ \phi_{z,\min} \end{bmatrix}, \qquad \boldsymbol{\phi}_{\max} = \begin{bmatrix} \phi_{x,\max} \\ \phi_{y,\max} \\ \phi_{z,\max} \end{bmatrix} \tag{7}$$

of the tool frame $\mathcal{T}$ w.r.t. the manufacturing frame $\mathcal{M}$. Note that the simple (linear) box constraints (6) and (7) may also be replaced by nonlinear constraints, e.g. circular or spherical distance constraints.

4.4. Collision avoidance

The proposed path planning framework incorporates the general collision detection framework *V-Clip* [53], which is summarized in this section. This framework is used to find the pair of globally closest features of convex polyhedrons, of which the signed distance and also

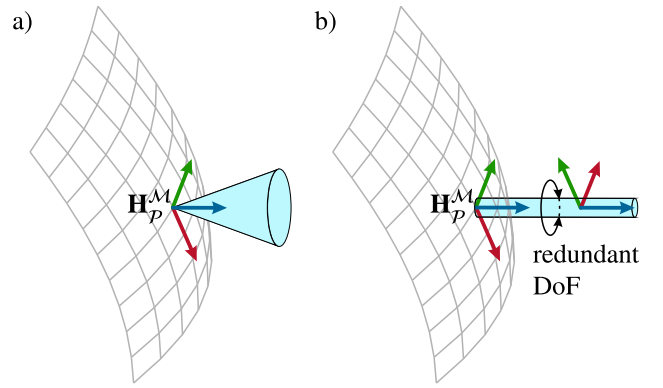


Fig. 2. Tolerances for manufacturing processes (highlighted in light blue): (a) cutting process, (b) spray painting process.

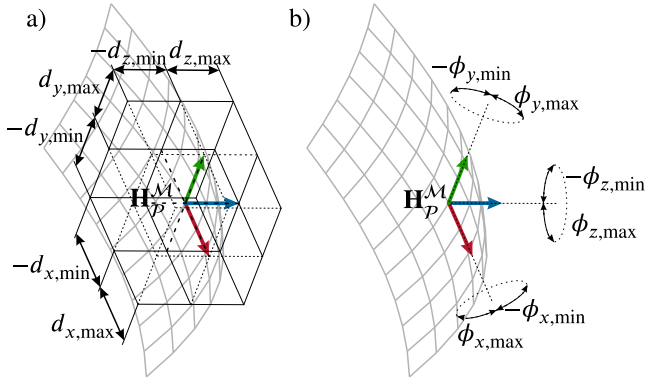


Fig. 3. Tolerance bands and process windows: (a) displacements, (b) orientation deviations of the tool frame $\mathcal{T}$ w.r.t. the manufacturing frame $\mathcal{M}$.

the analytical gradient is computed, see, e.g., [54]. The signed distance and its gradient are utilized in the optimization algorithm in the next section.

Each collision object in the environment is described as one polyhedron, consisting of vertices, edges and faces. A feature is represented by a single vertex, two vertices defining an edge or three vertices defining a face. The *V-Clip* algorithm iteratively examines the neighboring features until the pair of closest features is determined. Assuming that the robot movement between two consecutive poses of the manufacturing path $\mathcal{H}_{\mathcal{P}}^{\mathcal{M}}$ is small, the pair of closest features rarely changes. Hence, the previously found pair is chosen as initial guess for the subsequent path planning step. In this way, the pair of closest features is mostly found in the first iteration of the algorithm.

With the *V-Clip* framework, arbitrary objects can be included in the collision avoidance, e.g. parts of the robot itself, the tool, the workpiece and the environment. The collision checks are then tailored to the manufacturing process, the robot and its environment. If collisions between multiple objects have to be examined, the *V-Clip* algorithm is applied to each pair of collision objects separately. Therefore, the number of collision checks can be easily reduced to lower the computation time by examining only pairs of objects which are critical for the considered manufacturing process. An example of a collision model for a drawing process is shown in Fig. 4. In the depicted manufacturing process, the robot's wrist and the tool mounted on the end-effector are checked for collisions with the table and the box on the table.

5. Optimization-based path planning

The proposed optimization-based path planning algorithm starts by finding a discrete subset of all feasible robot starting configurations for

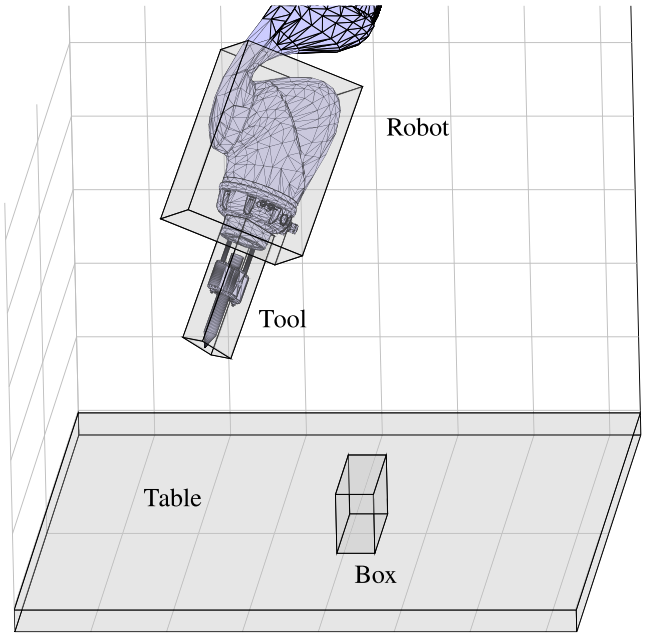


Fig. 4. Convex polyhedrons as collision objects are checked to avoid collisions from the robot and the tool with the table and the box on the table.

the first manufacturing frame $\mathbf{H}_{\mathcal{P},1}^{\mathcal{M}}$, considering the available process DoF, redundant DoF, process tolerances, windows and constraints. Starting from the obtained initial configurations, the corresponding joint-space paths are computed by sequentially solving an optimization problem, taking into account the process properties and collision avoidance. Finally, the best joint-space path is selected and a time parametrization is determined to obtain a dynamically feasible robot trajectory. Note that the proposed path planner is deterministic, as no random samples are required to solve the path planning problem.

In this section, the components of the path planner are explained in more detail. First, the starting configurations are discussed. Second, the optimization problem and the sequential solution procedure are introduced. Third, a variety of cost function terms and constraint functions are introduced, which can be combined and parametrized to precisely describe the considered manufacturing process. Fourth, the selection process for the best joint-space path is detailed and finally, the time parametrization of this path is explained.

5.1. Starting configurations

By considering the process and the redundant DoF, as well as the process tolerances and windows, a set of feasible initial tool poses are derived from the first manufacturing frame $\mathbf{H}_{\mathcal{P},1}^{\mathcal{M}}$. The process windows and tolerance bands defined in (6), (7) are sampled with an equidistant grid. The vectors

$$\mathbf{d}_{\mathbf{t}} = \frac{1}{d_{\text{grid}}} \left(\mathbf{d}_{\max} - \mathbf{d}_{\min}\right) \circ \mathbf{t} + \mathbf{d}_{\min} \quad , \tag{8}$$

$$\boldsymbol{\phi}_{\mathbf{r}} = \frac{1}{\phi_{\text{grid}}} \left(\boldsymbol{\phi}_{\max} - \boldsymbol{\phi}_{\min}\right) \circ \mathbf{r} + \boldsymbol{\phi}_{\min} \quad , \tag{9}$$

denote the individual grid points and $\circ$ is the element-wise product, also known as Hadamard product [55]. Thereby, $d_{\text{grid}}+1$ and $\phi_{\text{grid}}+1$ are the number of grid points and the vectors $\mathbf{t}^{\mathrm{T}} = \begin{bmatrix} t_x & t_y & t_z \end{bmatrix}$ and $\mathbf{r}^{\mathrm{T}} = \begin{bmatrix} r_x & r_y & r_z \end{bmatrix}$ are composed of the indices $t_x, t_y, t_z = 0, 1, \ldots, d_{\text{grid}}$ and $r_x, r_y, r_z = 0, 1, \ldots, \phi_{\text{grid}}$. Note that the indices $\mathbf{t}$ and $\mathbf{r}$ in (8) and (9) are vector-valued. Using every combination of the indices in $\mathbf{t}$ and $\mathbf{r}$,

i.e. every sampled grid point, a set of initial frames $\mathcal{H}$ is derived from the first manufacturing frame $\mathbf{H}_{\mathcal{P},1}^{\mathcal{M}}$ in the form

$$\mathcal{H} = \left\{ \mathbf{H}_{\mathcal{P},1}^{\mathcal{M}} \begin{bmatrix} \mathbf{R}_{\Delta}(\boldsymbol{\phi}_{\mathbf{r}}) & \mathbf{d}_{\mathbf{t}} \\ \mathbf{0} & 1 \end{bmatrix} \middle| \begin{matrix} t_x, t_y, t_z = 0, 1, \dots, d_{\text{grid}} \\ r_x, r_y, r_z = 0, 1, \dots, \phi_{\text{grid}} \end{matrix} \right\} \tag{10}$$

with the rotation matrix for the roll–pitch–yaw angles $\boldsymbol{\phi}_{\mathbf{r}}^{\mathrm{T}} = \begin{bmatrix} \phi_{x,r_x} & \phi_{y,r_y} & \phi_{z,r_z} \end{bmatrix}$

$$\mathbf{R}_{\Delta}(\boldsymbol{\phi}) = \begin{bmatrix} c_x c_y & c_x s_y s_z - s_x c_z & c_x s_y c_z + s_x s_z \\ s_x c_y & s_x s_y s_z + c_x c_z & s_x s_y c_z - c_x s_z \\ -s_y & c_y s_z & c_y c_z \end{bmatrix}, \tag{11}$$

where s_x and c_x denote abbreviations of the trigonometric functions $\sin(\phi_x)$ and $\cos(\phi_x)$, respectively, see [52].

Next, the discrete subset of all feasible joint-space configurations $\mathcal{Q}_f$ for the set of initial frames $\mathcal{H}$ in (10) is computed. This yields a total number of e_f joint-space solutions, which are sequentially numbered as $\mathcal{Q}_f = \{\mathbf{q}_{f,1}, \mathbf{q}_{f,2}, \dots, \mathbf{q}_{f,e_f}\}$. This set may be computed via an analytical inverse kinematics (if present) or using numerical inverse kinematics solvers, e.g. *TRAC-IK* [42]. Note that the redundant DoF of kinematically redundant robots have to be sampled as well. To reduce the number of initial joint configurations, the joint-space solutions contained in $\mathcal{Q}_f$ are filtered using a minimum-distance criterion. This yields the reduced set of joint-space solutions $\mathcal{Q}_g$ as

$$\begin{aligned} \mathcal{Q}_g &= \{\mathbf{q}_{g,1}, \mathbf{q}_{g,2}, \dots, \mathbf{q}_{g,e_g}\} \\ &= \left\{ \mathbf{q}_{f,i}, \mathbf{q}_{f,j} \in \mathcal{Q}_f \middle| q_{\text{dist}} < \left\| \mathbf{q}_{f,i} - \mathbf{q}_{f,j} \right\|_{\infty}, i, j = 1, \dots, e_f \right\}, \end{aligned} \tag{12}$$

where q_{dist} determines the minimum distance between any two solutions in $\mathcal{Q}_g$. In (12), $\|\cdot\|_{\infty}$ denotes the maximum norm of a vector and e_g is the total number of solutions in $\mathcal{Q}_g$. Note that with $\mathbf{q}_{f,i}$ and $\mathbf{q}_{f,j}$ satisfying the condition in (12), the joint configuration with the smallest joint angle of axis 1 is used in $\mathcal{Q}_g$ due to the implementation of the used MATLAB function *uniquetol*.

Note that sampling the process windows and computing the inverse kinematics solutions only need to be performed with a low grid resolution and only for the first manufacturing frame $\mathbf{H}_{\mathcal{P},1}^{\mathcal{M}}$, which yields different joint-space solutions $\mathcal{Q}_g$ for the subsequent path planning problem. In contrast, sampling-based path planning algorithms require a large number of samples for each pose of the manufacturing path, see, e.g., the *Descartes* algorithm [21], which becomes infeasible for higher numbers of DoF.

5.2. Optimization problem

Starting from each joint-space solution $\mathcal{Q}_g$ from Section 5.1, a series of optimization problems is solved sequentially, see Fig. 5. Each series of optimization problems is given by

$$\mathbf{q}_{u,i}^{*} = \underset{\mathbf{q}_{u,i} \in \mathbb{R}^n}{\arg\min}\ f(\mathbf{q}_{u,i}, \mathbf{H}_{\mathcal{P},i}^{\mathcal{M}}), \quad i = 1, \dots, m \tag{13a}$$

$$\text{s.t. } \mathbf{q}_{\min} \le \mathbf{q}_{u,i} \le \mathbf{q}_{\max} \tag{13b}$$

$$\mathbf{c}_{\text{eq}}(\mathbf{q}_{u,i}, \mathbf{H}_{\mathcal{P},i}^{\mathcal{M}}) = \mathbf{0} \tag{13c}$$

$$\mathbf{c}_{\text{ineq}}(\mathbf{q}_{u,i}, \mathbf{H}_{\mathcal{P},i}^{\mathcal{M}}) \le \mathbf{0}, \tag{13d}$$

with the initial guess for the i-th optimization problem

$$\mathbf{q}_{u,i,0} = \begin{cases} \mathbf{q}_{g,u} & i = 1 \\ \mathbf{q}_{u,i-1}^{*} & i > 1 \end{cases}, \tag{14}$$

where $u = 1, \dots, e_g$ selects the joint-space solution from $\mathcal{Q}_g$. The optimal joint configuration $\mathbf{q}_{u,i}^{*}$ for the specific path pose $\mathbf{H}_{\mathcal{P},i}^{\mathcal{M}}$ is found by minimizing the objective function f, which is detailed in the next section. Therefore, the interior-point method from the MATLAB solver

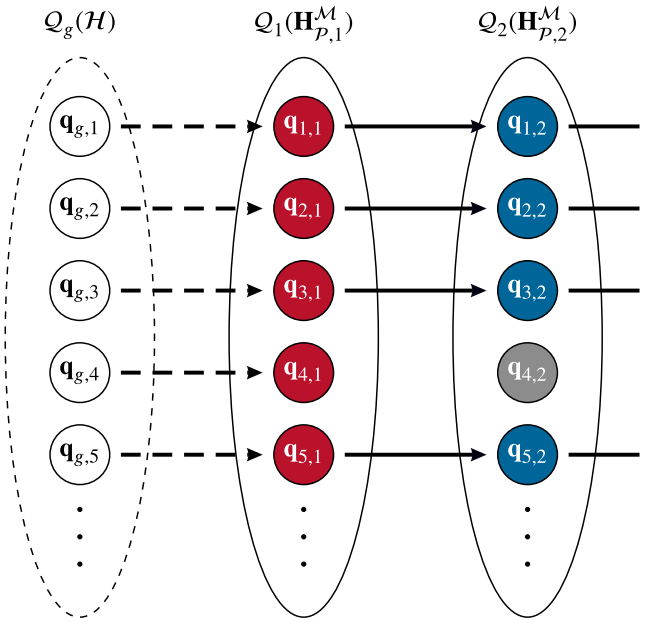


Fig. 5. Series of optimization problems starting from the initial guesses in $\mathcal{Q}_g$ obtained from $\mathcal{H}$ in (10) to find the optimal joint-space path. In this example, the initial guess $\mathbf{q}_{g,4}$ does not yield a complete continuous joint-space path.

fmincon is used, see, e.g., [56], which solves the corresponding problem, cf. (13)

$$\begin{bmatrix} \mathbf{q}_{u,i}^{*} & \boldsymbol{\sigma}^{*} \end{bmatrix} = \underset{\substack{\mathbf{q}_{u,i} \in \mathbb{R}^n \\ \boldsymbol{\sigma} \in \mathbb{R}^{\kappa}}}{\arg\min}\ f(\mathbf{q}_{u,i}, \mathbf{H}_{\mathcal{P},i}^{\mathcal{M}}) - \mu \sum_{i=1}^{\kappa} \ln(\sigma_i) \tag{15a}$$

$$\text{s.t. } \boldsymbol{\sigma} \ge 0 \tag{15b}$$

$$\mathbf{c}_{\text{eq}}(\mathbf{q}_{u,i}, \mathbf{H}_{\mathcal{P},i}^{\mathcal{M}}) = \mathbf{0} \tag{15c}$$

$$\begin{bmatrix} \mathbf{c}_{\text{ineq}}(\mathbf{q}_{u,i}, \mathbf{H}_{\mathcal{P},i}^{\mathcal{M}}) \\ \mathbf{q}_{\min} - \mathbf{q}_{u,i} \\ \mathbf{q}_{u,i} - \mathbf{q}_{\max} \end{bmatrix} + \boldsymbol{\sigma} = \mathbf{0}, \tag{15d}$$

with the slack variable $\boldsymbol{\sigma}$, the dimension of the inequality constraints vector $\kappa = \dim\left(\begin{bmatrix} \mathbf{c}_{\text{ineq}}^{\mathrm{T}} & \mathbf{q}_{\min}^{\mathrm{T}} & \mathbf{q}_{\max}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}\right)$ and $\mu > 0$, see [57,58]. The last term in (15a) are barrier functions to approximate the inequality constraints.

The process and redundant DoF and the process tolerances and windows of the specific manufacturing process are formulated as soft constraints in f and hard constraints using the equality constraints $\mathbf{c}_{\text{eq}}$ and inequality constraints $\mathbf{c}_{\text{ineq}}$. A number of different objective functions and constraints are provided and explained in the next section. Additionally, the upper and lower joint limits $\mathbf{q}_{\max}$ and $\mathbf{q}_{\min}$, respectively, are considered as inequality constraints in (13b). The initial guess $\mathbf{q}_{u,i,0}$ for each optimization problem in the sequence, see (14), is chosen as the solution $\mathbf{q}_{u,i-1}^{*}$ from the previous optimization. For the first manufacturing frame $\mathbf{H}_{\mathcal{P},1}^{\mathcal{M}}$, the joint-space solutions $\mathcal{Q}_g$ serve as initial guess for the optimization. To speed up the solution of the optimization problem, the analytical gradient of the objective function and constraints are utilized. Note that the joint-space solutions in $\mathcal{Q}_g$ are independent of each other. Hence, multiple joint-space paths can be computed in parallel on a multi-core CPU, which significantly improves the computational performance. Furthermore, the constraints and objective function can be adapted for each path pose $\mathbf{H}_{\mathcal{P},i}^{\mathcal{M}}, i = 1, \dots, m$, since the optimization problem is solved in series.

Remark 2. The solution space may further be increased by considering cross connections between individual joint-space paths, e.g., from $q_{1,1}$

to $q_{2,2}$ in Fig. 5, which are not taken into account in this work. This could be implemented using a graph representation of the solution space. Consequently, the individual solution paths depend from each other, which makes a parallel computation impossible. Including cross connections while preserving, at least to a certain extent, parallelism is an interesting future research direction. Note that a similar path planning approach based on the analytical inverse kinematics is published in [7].

5.3. Objective functions and constraints

With the proposed optimization-based path planning approach, the objective functions and constraints are tailored to precisely describe the considered manufacturing process and taking into account the process properties. This way, individual Cartesian coordinates may be strictly constrained while allowing tolerances in other coordinates.

In general, the objective function consists of p cost terms $f_j, j = 1, \dots, p$ in the form

$$f(\mathbf{q}_{u,i}, \mathbf{H}_{\mathcal{P},i}^{\mathcal{M}}) = \sum_{j=1}^{p} f_j(\mathbf{q}_{u,i}, \mathbf{H}_{\mathcal{P},i}^{\mathcal{M}}) \tag{16}$$

and the equality and inequality constraints are also composed of different components

$$\mathbf{c}_{\text{eq}} = \begin{bmatrix} \mathbf{c}_{\text{eq},1} \\ \mathbf{c}_{\text{eq},2} \\ \vdots \end{bmatrix} \quad , \qquad \mathbf{c}_{\text{ineq}} = \begin{bmatrix} \mathbf{c}_{\text{ineq},1} \\ \mathbf{c}_{\text{ineq},2} \\ \vdots \end{bmatrix} \quad . \tag{17}$$

Since the individual terms in (16)–(17) strongly depend on the manufacturing process under consideration, the indices will be omitted in the following to improve the clarity of presentation.

5.3.1. Position deviation

The deviation $\Delta\mathbf{p}$ of the actual tool position $\mathbf{p}_{\mathcal{P}}^{\mathcal{T}}$ from the desired position $\mathbf{p}_{\mathcal{P}}^{\mathcal{M}}$, described in the manufacturing frame $\mathcal{M}$, is computed as

$$\Delta\mathbf{p} = \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = \left(\mathbf{R}_{\mathcal{P}}^{\mathcal{M}}\right)^{\mathrm{T}} \left(\mathbf{p}_{\mathcal{P}}^{\mathcal{M}} - \mathbf{p}_{\mathcal{P}}^{\mathcal{T}}(\mathbf{q})\right) \quad , \tag{18}$$

using the transformations from Section 4.2.

Depending on the considered manufacturing process, the position deviation may appear as soft constraint in the objective function to implement process tolerances

$$f = \frac{1}{2}\Delta\mathbf{p}^{\mathrm{T}}\mathbf{A}_1\Delta\mathbf{p} \quad , \tag{19}$$

with the positive semi-definite weighting matrix $\mathbf{A}_1$, as hard constraint in the equality constraint to implement a strictly constrained process DoF

$$\mathbf{c}_{\text{eq}} = \Delta\mathbf{p} \quad , \tag{20}$$

or as inequality constraint to implement process windows

$$\mathbf{c}_{\text{ineq}} = \begin{bmatrix} \mathbf{d}_{\min} - \Delta\mathbf{p} \\ \Delta\mathbf{p} - \mathbf{d}_{\max} \end{bmatrix} \quad . \tag{21}$$

The position deviation may also appear in both the objective function (19) and the inequality constraint (21), which yields the implementation of tolerance bands. Note that instead of the complete vector $\Delta\mathbf{p}$, its components Δx, Δy and Δz may be used individually as soft, hard or inequality constraint. Fixing some entries using hard constraints while considering the remaining entries as soft constraints is possible, depending on the requirements of the manufacturing process.

The analytical gradient of (20) and (21) follow as

$$\frac{\partial \mathbf{c}_{\text{eq}}}{\partial \mathbf{q}} = \frac{\partial \Delta\mathbf{p}}{\partial \mathbf{q}} = -\left(\mathbf{R}_{\mathcal{P}}^{\mathcal{M}}\right)^{\mathrm{T}} \mathbf{J}_{\mathcal{P},v}^{\mathcal{T}}(\mathbf{q}) \quad , \tag{22}$$

$$\frac{\partial \mathbf{c}_{\text{ineq}}}{\partial \mathbf{q}} = \begin{bmatrix} -\frac{\partial \Delta\mathbf{p}}{\partial \mathbf{q}} \\ \frac{\partial \Delta\mathbf{p}}{\partial \mathbf{q}} \end{bmatrix} \quad , \tag{23}$$

with the Jacobian $\mathbf{J}_{\mathcal{P},v}^{\mathcal{T}}(\mathbf{q})$ related to the linear velocity of the tool frame $\mathcal{T}$ w.r.t. the workpiece frame $\mathcal{P}$. The gradient of (19) reads as

$$\frac{\partial f}{\partial \mathbf{q}} = \Delta\mathbf{p}^{\mathrm{T}}\mathbf{A}_1 \frac{\partial \Delta\mathbf{p}}{\partial \mathbf{q}} \quad . \tag{24}$$

5.3.2. Orientation deviation

The orientation deviation $\Delta\mathbf{R} = \mathbf{R}_{\mathcal{P}}^{\mathcal{M}}(\mathbf{R}_{\mathcal{P}}^{\mathcal{T}}(\mathbf{q}))^{\mathrm{T}}$, described as unit quaternion $\Delta\mathbf{o}$, reads as

$$\begin{aligned} \Delta\mathbf{o} &= \begin{bmatrix} \Delta\eta \\ \Delta\boldsymbol{\epsilon} \end{bmatrix} = \mathbf{o}_{\mathcal{P}}^{\mathcal{M}} \otimes \left(\mathbf{o}_{\mathcal{P}}^{\mathcal{T}}\right)^{-1} = \begin{bmatrix} \eta_{\mathcal{P}}^{\mathcal{M}} \\ \boldsymbol{\epsilon}_{\mathcal{P}}^{\mathcal{M}} \end{bmatrix} \otimes \begin{bmatrix} \eta_{\mathcal{P}}^{\mathcal{T}} \\ -\boldsymbol{\epsilon}_{\mathcal{P}}^{\mathcal{T}} \end{bmatrix} \\ &= \begin{bmatrix} \eta_{\mathcal{P}}^{\mathcal{M}}\eta_{\mathcal{P}}^{\mathcal{T}} + \left(\boldsymbol{\epsilon}_{\mathcal{P}}^{\mathcal{M}}\right)^{\mathrm{T}}\boldsymbol{\epsilon}_{\mathcal{P}}^{\mathcal{T}} \\ \eta_{\mathcal{P}}^{\mathcal{T}}\boldsymbol{\epsilon}_{\mathcal{P}}^{\mathcal{M}} - \eta_{\mathcal{P}}^{\mathcal{M}}\boldsymbol{\epsilon}_{\mathcal{P}}^{\mathcal{T}} - \boldsymbol{\epsilon}_{\mathcal{P}}^{\mathcal{M}} \times \boldsymbol{\epsilon}_{\mathcal{P}}^{\mathcal{T}} \end{bmatrix} \quad , \end{aligned} \tag{25}$$

where the operator $\otimes$ denotes the quaternion product. If the orientation of the actual tool frame $\mathcal{T}$ exactly matches the orientation of the desired manufacturing frame $\mathcal{M}$, the quaternion $\Delta\mathbf{o}$ in (25) becomes $\Delta\mathbf{o}^{\mathrm{T}} = \begin{bmatrix} 1 & \mathbf{0} \end{bmatrix}$, see [52]. Thus, the orientation deviation is constructed as soft constraint with the positive scalar weight a as

$$f = \frac{a}{2}(1 - \Delta\eta)^2 \quad . \tag{26}$$

In (26), all orientation deviations are penalized equally. If the manufacturing process allows for different orientation tolerances w.r.t. each axis, the soft constraint

$$f = \frac{1}{2}\Delta\boldsymbol{\epsilon}^{\mathrm{T}}\mathbf{A}_2\Delta\boldsymbol{\epsilon} \quad , \tag{27}$$

with the positive semi-definite weighting matrix $\mathbf{A}_2$ is used. In Appendix A.1 the cost terms in (26) and (27) for the orientation deviation (25) are motivated and validated using mathematical and geometric arguments.

The soft constraints (26) and (27) may also be implemented as hard constraints in the form

$$c_{\text{eq}} = 1 - \Delta\eta \tag{28}$$

or

$$\mathbf{c}_{\text{eq}} = \Delta\boldsymbol{\epsilon} \quad , \tag{29}$$

respectively. When using these hard constraints, rotations around certain axes can be restricted. Moreover, the inequality constraints

$$\mathbf{c}_{\text{ineq}} = \begin{bmatrix} \boldsymbol{\epsilon}_{\min} - \Delta\boldsymbol{\epsilon} \\ \Delta\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\max} \end{bmatrix} \quad , \tag{30}$$

can be employed, where $\boldsymbol{\epsilon}_{\min}$ and $\boldsymbol{\epsilon}_{\max}$ are computed using (25) from the minimum and maximum orientation deviation defined in (7). Analogous to the position deviation in Section 5.3.1, only individual components of the orientation error $\Delta\boldsymbol{\epsilon}$ may be included in (27), (29) and (30) by assembling the cost functions and constraints accordingly.

The gradients of the objective function terms (26) and (27) are computed as

$$\frac{\partial f}{\partial \mathbf{q}} = -a(1 - \Delta\eta)\frac{\partial \Delta\eta}{\partial \mathbf{q}} \tag{31}$$

and

$$\frac{\partial f}{\partial \mathbf{q}} = \Delta\boldsymbol{\epsilon}^{\mathrm{T}}\mathbf{A}_2 \frac{\partial \Delta\boldsymbol{\epsilon}}{\partial \mathbf{q}} \quad , \tag{32}$$

respectively, using the gradients $\frac{\partial \Delta\eta}{\partial \mathbf{q}}$ and $\frac{\partial \Delta\boldsymbol{\epsilon}}{\partial \mathbf{q}}$ found in Appendix A.2. The gradient of the constraint (28) reads as

$$\frac{\partial c_{\text{eq}}}{\partial \mathbf{q}} = -\frac{\partial \Delta\eta}{\partial \mathbf{q}} \tag{33}$$

and the gradients of (29) and (30) yield

$$\frac{\partial \mathbf{c}_{\text{eq}}}{\partial \mathbf{q}} = \frac{\partial \Delta\boldsymbol{\varepsilon}}{\partial \mathbf{q}} \quad , \tag{34}$$

$$\frac{\partial \mathbf{c}_{\text{ineq}}}{\partial \mathbf{q}} = \begin{bmatrix} -\frac{\partial \Delta\varepsilon}{\partial \mathbf{q}} \\ \frac{\partial \Delta\varepsilon}{\partial \mathbf{q}} \end{bmatrix} \quad . \tag{35}$$

5.3.3. Collision avoidance

The *V-Clip* algorithm [53] determines the pair of globally closest features of two convex polyhedrons, for which the signed distance l is computed. The signed distance l is positive, if two obstacles are at the distance l from each other and a negative distance l describes obstacles in collision. For every collision check $k = 1, \dots, b$ between two objects, the signed distances $l_k(\mathbf{q})$ of all feature pairs are utilized in the soft constraint as

$$f = \frac{1}{2} \begin{bmatrix} \max(0, l_{1,\min} - l_1) \\ \max(0, l_{2,\min} - l_2) \\ \vdots \\ \max(0, l_{b,\min} - l_b) \end{bmatrix}^{\mathrm{T}} \mathbf{B} \begin{bmatrix} \max(0, l_{1,\min} - l_1) \\ \max(0, l_{2,\min} - l_2) \\ \vdots \\ \max(0, l_{b,\min} - l_b) \end{bmatrix} \quad , \tag{36}$$

with the positive semi-definite weighting matrix $\mathbf{B}$ and the respective minimum distances $l_{k,\min}$. The signed distances l_k also be used in inequality constraints as

$$\mathbf{c}_{\text{ineq}} = \begin{bmatrix} -l_1 \\ -l_2 \\ \vdots \\ -l_b \end{bmatrix} \tag{37}$$

to ensure that no collision occurs in the planned joint-space path. Note that using both constraints (36) and (37) together improves the convergence of the optimization problem (13) significantly.

The analytical gradient of (36) reads as

$$\frac{\partial f}{\partial \mathbf{q}} = -\begin{bmatrix} \max(0, l_{1,\min} - l_1) \\ \max(0, l_{2,\min} - l_2) \\ \vdots \\ \max(0, l_{b,\min} - l_b) \end{bmatrix}^{\mathrm{T}} \mathbf{B} \begin{bmatrix} \frac{\partial l_1}{\partial \mathbf{q}} \\ \frac{\partial l_2}{\partial \mathbf{q}} \\ \vdots \\ \frac{\partial l_b}{\partial \mathbf{q}} \end{bmatrix} \quad . \tag{38}$$

In (38), the gradients of the signed distances l_k are computed in the form

$$\frac{\partial l_k}{\partial \mathbf{q}} = \begin{bmatrix} \left(\frac{\partial l_k}{\partial \mathbf{v}_1}\right) & \left(\frac{\partial l_k}{\partial \mathbf{v}_2}\right) & \cdots & \left(\frac{\partial l_k}{\partial \mathbf{v}_w}\right) \end{bmatrix} \begin{bmatrix} \mathbf{J}_{\mathcal{W},v}^{\mathcal{V}_1}(\mathbf{q}) \\ \mathbf{J}_{\mathcal{W},v}^{\mathcal{V}_2}(\mathbf{q}) \\ \vdots \\ \mathbf{J}_{\mathcal{W},v}^{\mathcal{V}_w}(\mathbf{q}) \end{bmatrix} \quad , \tag{39}$$

where the vectors $\mathbf{v}_h, h = 1, \dots, w$, denote the positions of the vertices $\mathcal{V}_h, h = 1, \dots, w$ of the polyhedron and $\mathbf{J}_{\mathcal{W},v}^{\mathcal{V}_h}(\mathbf{q}), h = 1, \dots, w$, are the corresponding Jacobians related to the linear velocity. Note that only the vertices defining the closest feature pair are required to compute the gradient and therefore most of the entries in the left vector in (39) are zero. The gradient of the inequality constraint (37) directly follows from (39). Although the analytical gradient (39) is discontinuous if the globally closest features change, this does not affect the performance of the optimization algorithm (13) in practice.

5.3.4. Joint limits and path continuity

In order to obtain physically feasible robot motions, a joint-space path must be sufficiently smooth and adhere to the joint limits. Hence, to derive continuous joint-space paths, the objective function term

$$f = \frac{1}{2} \left(\mathbf{q} - \mathbf{q}_{u,i,0}\right)^{\mathrm{T}} \mathbf{C} \left(\mathbf{q} - \mathbf{q}_{u,i,0}\right) \quad , \tag{40}$$

with the positive semi-definite weighting matrix $\mathbf{C}$, is used to penalize large joint movements. In (40), the joint configuration $\mathbf{q}_{u,i,0}$ is the initial guess for the optimization of the i-th manufacturing frame $\mathbf{H}_{\mathcal{P},i}^{\mathcal{M}}$, see (14). The corresponding analytical gradient reads as

$$\frac{\partial f}{\partial \mathbf{q}} = \left(\mathbf{q} - \mathbf{q}_{u,i,0}\right)^{\mathrm{T}} \mathbf{C} \quad . \tag{41}$$

Note that robot movements through kinematic singular points of the robot are explicitly allowed with the proposed path planning framework and are executable with standard controllers of an industrial robot using joint-space control.

To compute feasible robot motions, the joint angles of the robot are constrained in (13b) within their axes limits. To improve convergence, joint angles are penalized as soft constraints with the objective function term

$$f = \frac{1}{2} \tilde{\mathbf{q}}^{\mathrm{T}} \mathbf{W} \tilde{\mathbf{q}} \, , \quad \tilde{\mathbf{q}} = \mathbf{q} - \frac{1}{2} \left(\mathbf{q}_{\max} + \mathbf{q}_{\min}\right) \tag{42}$$

and the corresponding gradient

$$\frac{\partial f}{\partial \mathbf{q}} = \tilde{\mathbf{q}}^{\mathrm{T}} \mathbf{W} \quad . \tag{43}$$

The diagonal matrix $\mathbf{W}$ is chosen as

$$\mathbf{W} = \frac{w}{2} \operatorname{diag} \left(\mathbf{q}_{\max} - \mathbf{q}_{\min}\right)^{-2} \quad , \tag{44}$$

where $\mathbf{q}_{\min}^{\mathrm{T}} = \begin{bmatrix} q_{1,\min} \cdots q_{n,\min} \end{bmatrix}$ and $\mathbf{q}_{\max}^{\mathrm{T}} = \begin{bmatrix} q_{1,\max} \cdots q_{n,\max} \end{bmatrix}$ denote the mechanical joint limits of the robot and $w > 0$ is a weighting parameter.

5.4. Optimal joint-space path

Multiple joint-space paths are generated by solving the sequence of optimization problems (13) and starting from the e_g different joint-space solutions contained in $\mathcal{Q}_g$. If a subproblem of a sequence does not yield a feasible solution, the corresponding joint-space path is discarded, see Fig. 5. To find the optimal joint-space path, the costs of the objective function terms for the manufacturing path $\mathcal{H}_{\mathcal{P}}^{\mathcal{M}}$ are added up with

$$f^u = \sum_{i=1}^{m} \sum_{j \in C} f_j(\mathbf{q}_{u,i}, \mathbf{H}_{\mathcal{P},i}^{\mathcal{M}}) \, , \quad u = 1, \dots, e_g \tag{45}$$

for each feasible joint-space path. By comparing the individual costs f^u of each feasible joint-space path $u = 1, \dots, e_g$, the optimal joint-space path $\mathcal{Q}^* = \{\mathbf{q}_1^*, \dots, \mathbf{q}_m^*\}$ is found for the desired manufacturing path $\mathcal{H}_{\mathcal{P}}^{\mathcal{M}}$. Note that only a particular subset $C \subset \{1, \dots, p\}$ of objective function terms may be used in (45) instead of all terms $f_j, j = 1, \dots, p$ from (16). For example, the optimization problem (16) might consider all process properties of the manufacturing process, collision avoidance and the joint limits to yield feasible solutions for the desired manufacturing path $\mathcal{H}_{\mathcal{P}}^{\mathcal{M}}$. Then, the evaluation of the optimal joint-space path in (45) might use a reduced subset C which only contains the process properties and, hence, only considers the achieved manufacturing result. Consequently, the best joint-space path $\mathcal{Q}^*$ is optimal regarding the manufacturing quality and does not take the robot movement into account.

Note that the weighting matrices $\mathbf{A}_1$, $\mathbf{A}_2$, $\mathbf{B}$ and $\mathbf{C}$ and the scalar weights w and a introduced in the previous section have a strong impact on the convergence behavior and the shape of the resulting joint-space solution, in particular if the considered process exhibits a large number of (redundant) process DoF, tolerances and windows.

5.5. Trajectory generation

The result of the planning algorithm is the optimal joint-space path $\mathcal{Q}^*$ corresponding to the desired manufacturing path $\mathcal{H}_{\mathcal{P}}^{\mathcal{M}}$. As a final step, the joint-space path $\mathcal{Q}^*$ is time parametrized to compute a piecewise trajectory $\mathbf{q}^*(t)$ with the sample points $(t_i, \mathbf{q}_i^*), i = 1, \dots, m$. The time stamps t_i are derived from the Cartesian distance between two consecutive sample points $\mathbf{q}_{i-1}^*$ and $\mathbf{q}_i^*$ with

$$t_i = t_{i-1} + \frac{\left\| \mathbf{p}_{\mathcal{P}}^{\mathcal{T}}\left(\mathbf{q}_i^*\right) - \mathbf{p}_{\mathcal{P}}^{\mathcal{T}}\left(\mathbf{q}_{i-1}^*\right) \right\|_2}{\dot{p}_{d,i}} \, , \quad i = 1, \dots, m \quad . \tag{46}$$

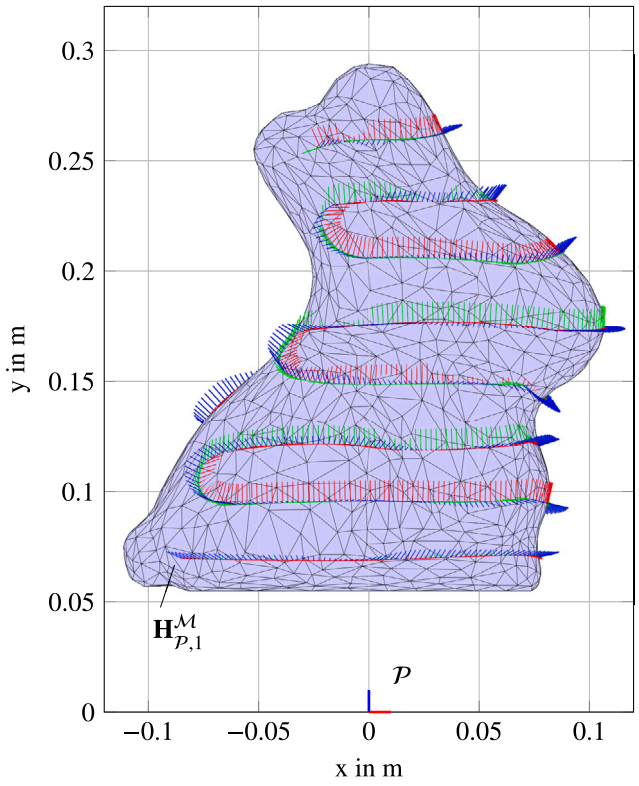


Fig. 6. Manufacturing path $\mathcal{H}^{\mathcal{M}}_{\mathcal{P}}$ on the rabbit-shaped workpiece in the workpiece frame $\mathcal{P}$.

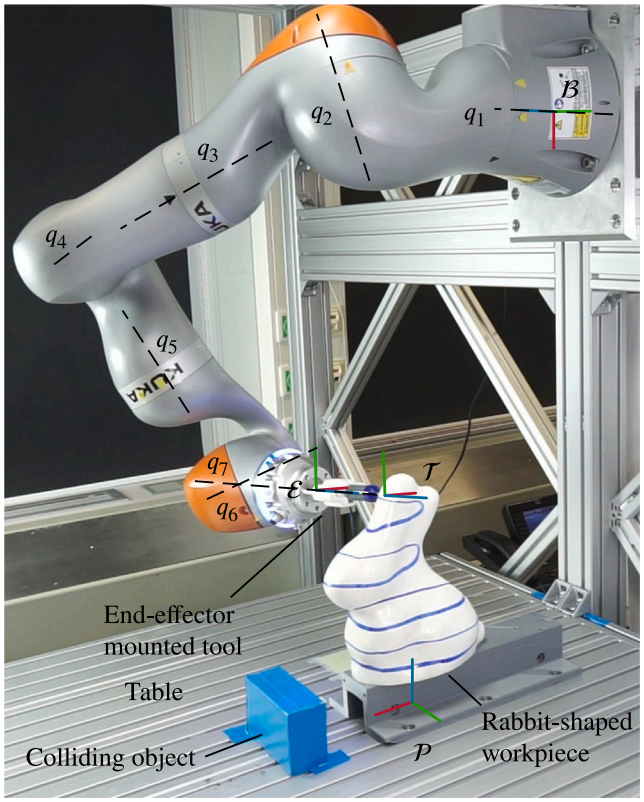


Fig. 7. Experimental setup for the drawing process after completing the process. The coordinate systems and robot joints are annotated.

The desired path velocity is specified by $\dot{p}_{d,i} > 0, i = 1, \ldots, m$, according to the manufacturing process and $\|\cdot\|_2$ denotes the Euclidean norm.

The optimal smooth joint-space trajectory $\mathbf{q}^*(t)$ is generated by computing piecewise cubic Hermite interpolating polynomials [59] for the sample points $(t_i, \mathbf{q}^*_i), i = 1, \ldots, m$. The obtained trajectory $\mathbf{q}^*(t)$ is then executed on the robot to perform the manufacturing process considering the specified process properties.

6. Experimental results

In this section, the proposed path planning framework is applied to two example applications with significantly different process properties. To this end, only the weighting matrices of the objective function terms have to be adapted and the equality and inequality constraints have to be adapted to precisely describe the respective manufacturing process, see Section 5.

The first application is a drawing process, in which a marker with rectangular nib is utilized to draw thin and thick lines on the surface of the workpiece. The thickness of the lines depends on the orientation of the marker w.r.t. the desired drawing path. In the second application, a spraying process is demonstrated. The rotationally symmetric spray nozzle is considered as redundant DoF in this process. The rabbit-shaped workpiece and the manufacturing path $\mathcal{H}^{\mathcal{M}}_{\mathcal{P}}$ employed in both applications are shown in Fig. 6. The total length of the meander-shaped path is approximately 2.5 m with 1024 given path poses. Note that the z-axes of the manufacturing frames (blue) are normal to the workpiece surface and that the manufacturing frames rotate w.r.t. the surface normal vector along the manufacturing path.

6.1. Drawing process

In this process, a line specified by the desired manufacturing path has to be drawn on a 3D-printed rabbit with a marker mounted on the end-effector of an industrial robot. This experiment demonstrates an industrial process with an end-effector mounted tool, see Section 4.2.1, and a complex continuous manufacturing path. This process is representative for similar industrial processes like, e.g. welding, grinding or cutting. The drawing process was executed and validated in a laboratory environment, see Fig. 7. Planning long continuous robot motions with multiple constraints is challenging due to the restrictive mechanical axes limits and the limited workspace of the robot.

6.1.1. Drawing process properties

The experimental setup of the drawing process is shown in Fig. 7. The robot KUKA LBR iiwa 14 R820 is employed to draw a continuous line specified by the manufacturing path on the surface of a stationary rabbit-shaped workpiece with a marker with rectangular nib mounted on the end-effector. To account for kinematic inaccuracies of the real robot, the drawing tool is equipped with a passive compliance mechanism comprising two linear springs. The rectangular nib allows to draw different line thicknesses by rotating the marker around its normal vector. To demonstrate the capabilities of the path planning framework including collision avoidance, the blue collision object in Fig. 7 is placed in front of the workpiece. Due to this blue object the robot cannot reach certain sections of the manufacturing path $\mathcal{H}^{\mathcal{M}}_{\mathcal{P}}$ exactly without collision, see ①–② in Fig. 8.

To perform the drawing process on the surface of the workpiece the position of the nib has to exactly follow the positions of the manufacturing path $\mathcal{H}^{\mathcal{M}}_{\mathcal{P}}$ in Fig. 6. In contrast, orientation deviations of the marker from the surface normal in a certain range only marginally degrade the quality of the drawn line and are therefore specified as cone-shaped process tolerances, see Fig. 2a. Since the nib of the marker is rectangular, a deviation from the desired rotation around the surface normal changes the line thickness, which should be avoided. Therefore, the tolerance band for the rotation around the z-axis is chosen more

Table 1
Objective function terms, constraints and weights used for the drawing process.

Variables	Equations	Weights
Position deviation		
$f(\Delta\mathbf{p})$	(19)	$\mathbf{A}_1 = \mathbf{0}$
$\mathbf{c}_{\text{eq}}(\Delta\mathbf{p})$	(20)	Enabled
$\mathbf{c}_{\text{ineq}}(\Delta\mathbf{p})$	(21)	Disabled
Orientation deviation		
$f(\Delta\eta)$	(26)	$a = 80$
$f(\Delta\boldsymbol{\epsilon})$	(27)	$\mathbf{A}_2 = \text{diag}\,(1, 1, 30)$
$\mathbf{c}_{\text{eq}}(\Delta\eta)$	(28)	Disabled
$\mathbf{c}_{\text{eq}}(\Delta\boldsymbol{\epsilon})$	(29)	Disabled
$\mathbf{c}_{\text{ineq}}(\Delta\boldsymbol{\epsilon})$	(30)	Enabled
Collision avoidance		
$f(l_k)$	(36)	$\mathbf{B} = 10^3\,\text{diag}\,(1, 1, 4)$
$\mathbf{c}_{\text{ineq}}(l_k)$	(37)	Enabled
Joint limits and path continuity		
$f(\mathbf{q})$	(40)	$\mathbf{C} = \text{diag}\,(10, 10, 5, 5, 5, 5)$
$f(\mathbf{q})$	(42), (44)	$w = 0.05$
Robot Kuka LBR iiwa 14 R820 [60]		
$\mathbf{q}_{\max}^{\mathrm{T}} = -\mathbf{q}_{\min}^{\mathrm{T}} = [170°\ 120°\ 170°\ 120°\ 170°\ 120°\ 175°]$		

restrictively. These process properties for the position and orientation are represented by tolerance bands in the form (see Fig. 3)

$$\mathbf{d}_{\max} = -\mathbf{d}_{\min} = \begin{bmatrix} 0\,\text{m} \\ 0\,\text{m} \\ 0\,\text{m} \end{bmatrix}, \quad \boldsymbol{\phi}_{\max} = -\boldsymbol{\phi}_{\min} = \begin{bmatrix} 40° \\ 40° \\ 20° \end{bmatrix} . \tag{47}$$

By parametrizing the objective function and selecting the corresponding equality and inequality constraints, the above process properties are systematically considered in the proposed optimization-based path planner. Since no position deviation is allowed, the equality constraint of the position deviation (20) is used and the corresponding objective term (19) and inequality constraint (21) are omitted. To allow for orientation deviations according to (47), the equality constraints of the marker orientation (28) and (29) are disabled. The deviation from the desired orientation of the manufacturing path $\mathcal{H}_{\mathcal{P}}^{\mathcal{M}}$ is penalized with the objective term (26) to minimize the utilized tolerances. Additionally, with the soft constraint (27), the deviations of the different orientation coordinates are weighted according to the allowed tolerances (47). Also the inequality constraint (30) is enabled to guarantee that the tolerance bands from (47) are observed. Moreover, it is checked that there are no collisions of the marker and the last robot link with the table and the blue collision object, see Fig. 4. Therefore, the corresponding objective function (36) together with the inequality constraint (37) are used to ensure a collision-free robot movement. Finally, to obtain continuous joint movements, the objective terms (40) and (42) are employed, with the robot joint limits $\mathbf{q}_{\min}$ and $\mathbf{q}_{\max}$. Note that the inverse kinematics is implicitly solved by numerical optimization.

The weights and parameters of the individual terms of the objective function and constraints are chosen empirically and are summarized in Table 1. The matrix $\mathbf{A}_1$ weights the position deviation (19), the scalar a and the matrix $\mathbf{A}_2$ the orientation deviation, see (26) and (27), the matrix $\mathbf{B}$ penalizes joint configurations near obstacles see (36), the matrix $\mathbf{C}$ weights joint movements according to (40), and the weight w in (42), (44) penalizes joint angles near their axis limits. In general, diagonal matrices are advantageous, as the number of parameters is greatly reduced and couplings between the individual DoF are avoided. In general, larger matrix entries lead to smaller errors in the respective DoF. Depending on which term in the objective function is crucial for the specific process, the individual weights are chosen larger or smaller.

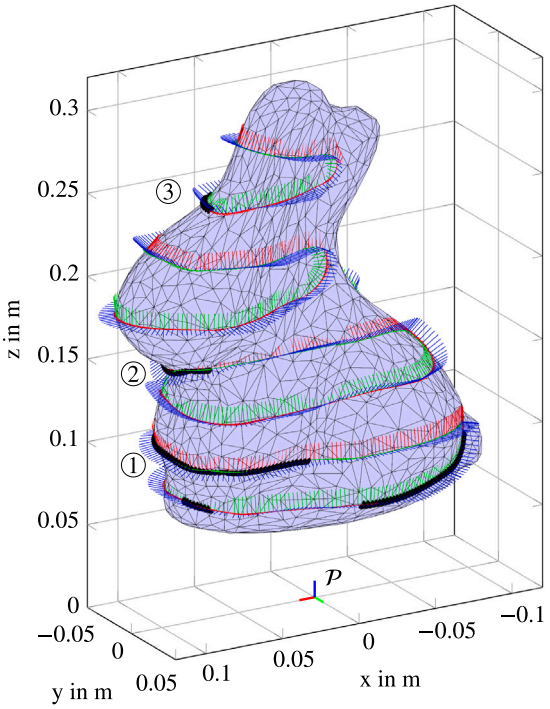


Fig. 8. Side view of the manufacturing path $\mathcal{H}_{\mathcal{P}}^{\mathcal{M}}$ on the rabbit-shaped workpiece in the workpiece frame $\mathcal{P}$. The manufacturing poses marked with black dots can only be reached by exploiting the process properties.

6.1.2. Experimental results of the drawing process

The position of the workpiece frame $\mathcal{P}$ relative to the robot base frame $\mathcal{B}$ is computed with an optimization-based algorithm [7] to reach as many manufacturing path poses in $\mathcal{H}_{\mathcal{P}}^{\mathcal{M}}$ as possible with the tool. Nevertheless, more than 15% of the manufacturing poses in $\mathcal{H}_{\mathcal{P}}^{\mathcal{M}}$ from Fig. 6 are only reachable by exploiting the process properties. These points cannot be reached exactly in position and orientation due to mechanical axes limits, the limited workspace of the robot and colliding objects, i.e. the blue box and the table. These poses are shown in Fig. 8 as black dots.

Using the set of parameters given in Table 1 and described in Section 6.1.1, all feasible joint-space solutions $\mathcal{Q}_g$ for the first frame $\mathbf{H}_{\mathcal{P},1}^{\mathcal{M}}$, see Fig. 6, of the manufacturing path $\mathcal{H}_{\mathcal{P}}^{\mathcal{M}}$ are determined. Subsequently, the series of optimization problems (13) is solved for every starting configuration $\mathcal{Q}_g$, yielding the optimal joint-space path $\mathcal{Q}^*$ by evaluating (45). The same objective function terms as in the optimization (13) are employed, which allows to reuse the already calculated objective function terms for the evaluation, see Table 1. By applying the time parametrization (46), the optimal joint-space trajectory $\mathbf{q}^*(t)$ is obtained and depicted in Fig. 9. The trajectories of the individual joints $q_h^*(t), h = 1, \ldots, n$ are normalized to the respective axes limits $\mathbf{q}_{\min}$ and $\mathbf{q}_{\max}$. Overall, the available axes ranges of the robot joints have to be utilized to a large extent in order to perform the drawing process, which emphasizes the complexity of the task. In particular, the trajectories of the axes 4, 6 and 7 occasionally reach their mechanical axes limits. Additionally, the robot moves three times through a singular configuration, i.e. axis 2 passes through zero, see Fig. 9. The mean calculation time of a single optimization problem from the series (13) including collision checks is approximately 70 ms on an Intel Core i7-8700K in a single-core implementation. If all CPU cores are utilized, the mean calculation time reduces to around 18 ms, since multiple optimization problems are solved in parallel. The total calculation time of the path planning problem with $e_g = 6$ starting configurations is approximately 90 s. Note that the total calculation time depends on the number of path poses m in the given manufacturing path, the number

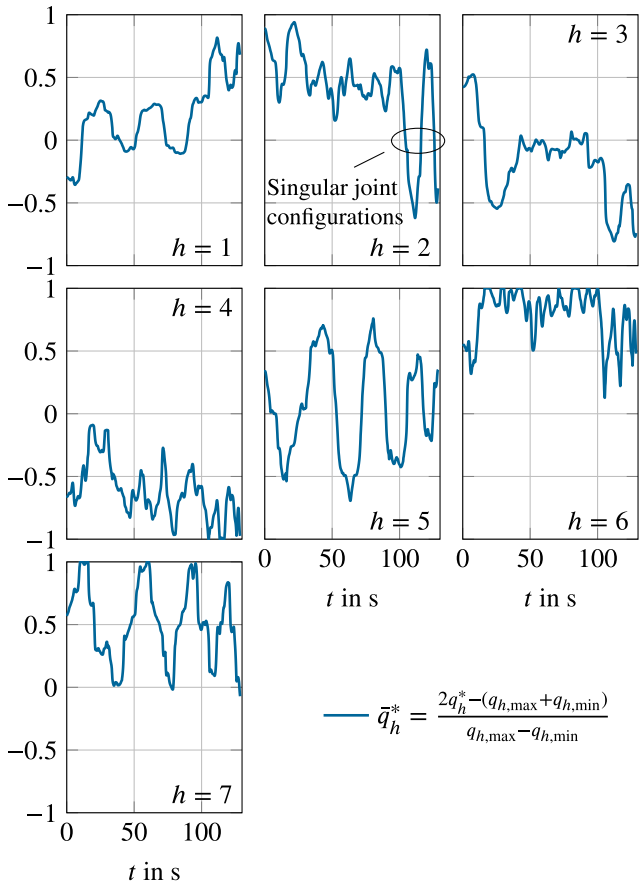


Fig. 9. Optimal joint-space trajectory $\mathbf{q}^*(t)$ for the drawing process. The trajectories of the individual joints are normalized to their respective axes limits $\mathbf{q}_{\min}$ and $\mathbf{q}_{\max}$.

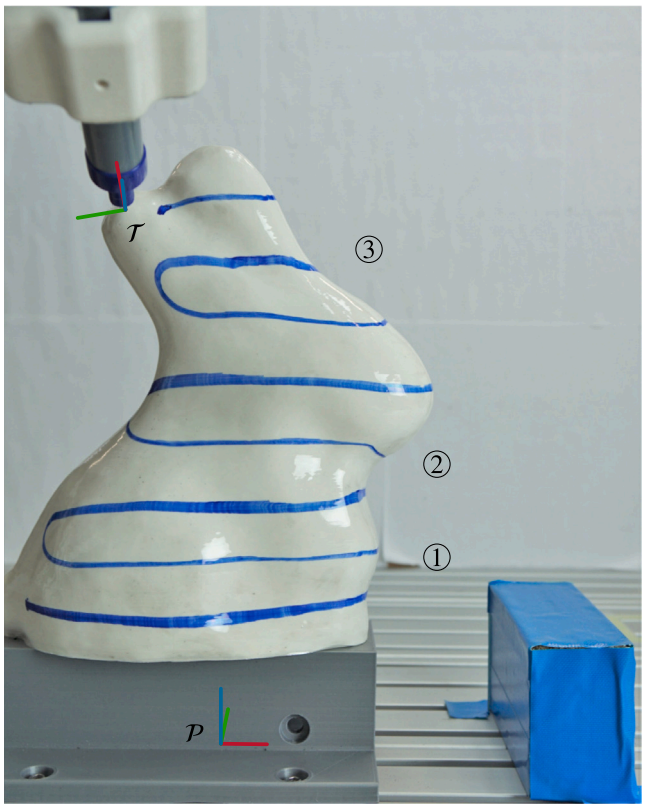


Fig. 10. Result of the drawing process using the manufacturing path $\mathcal{H}_{\mathcal{P}}^{\mathcal{M}}$ shown in Fig. 6.

of different starting configurations e_g, the CPU and the number of available CPU cores. Without collision checks, see Section 5.3.3, the mean optimization time further reduces by a factor of 3. Note that the path planning problem for the drawing process in this scenario is highly constrained and many poses along the manufacturing path $\mathcal{H}_{\mathcal{P}}^{\mathcal{M}}$ are only reachable by exploiting process tolerances. Therefore, most joint-space solutions in $\mathcal{Q}_g$ do not lead to a complete and feasible joint-space path.

A video of the experimental result of the drawing process is shown in www.acin.tuwien.ac.at/4adf/. The result of the drawing process in Fig. 10 shows a good agreement with the desired manufacturing path $\mathcal{H}_{\mathcal{P}}^{\mathcal{M}}$ in Fig. 6. The sections of the path with the two different desired line thicknesses can be easily distinguished. Note that the experiment is executed without absolute calibration and without feedback of the actual Cartesian end-effector position.

As shown in Fig. 8, a significant portion of the manufacturing path poses from $\mathcal{H}_{\mathcal{P}}^{\mathcal{M}}$ is not exactly reachable with the tool mounted on the end-effector. Hence, process tolerances have to be utilized to solve the path planning problem and compute the optimal trajectory $\mathbf{q}^*(t)$. In Fig. 11, the orientation deviations of the TCP frame poses $\mathbf{H}_{\mathcal{P}}^{\mathcal{T}}(\mathbf{q}^*)$ from the desired manufacturing path $\mathcal{H}_{\mathcal{P}}^{\mathcal{M}}$ are presented. The maximum values of the utilized process tolerances are below 30° for ϕ_x and ϕ_y and below 10° for ϕ_z, which results from the higher weighting of ϕ_z in (27), see Table 1. The process tolerances are within the allowed tolerance bands defined in (47).

The first path pose of the manufacturing path $\mathbf{H}_{\mathcal{P},1}^{\mathcal{T}}$ is located near the bottom of the workpiece, see Fig. 6. Moving with the marker from the first path pose $\mathbf{H}_{\mathcal{P},1}^{\mathcal{T}}$ towards ① in Fig. 8, the proposed path planner uses tolerances to avoid collisions with the blue collision object and the table. Also tolerances from the highly penalized rotation around the surface normal vector are necessary due to collision avoidance and the joint limit of axis 7, see Fig. 9. High usage of orientation tolerances is also necessary at $t = 71$ s located at ②. At ②, the robot motion cannot be solved without the additional freedom provided by the tolerances due to the blue collision object. Note that the surface normals at ② directly point towards the blue collision object, see Fig. 6. At $t = 105$ s located at ③, again high orientation tolerances are used. These orientation tolerances result from moving multiple times through a singular configuration, which is required to solve the path planning problem in one continuous robot motion, see also Fig. 9. Because the Cartesian position of the TCP frame $\mathbf{H}_{\mathcal{P}}^{\mathcal{T}}$ w.r.t. the manufacturing frame $\mathcal{H}_{\mathcal{P}}^{\mathcal{M}}$ is implemented as equality constraint, see Table 1, the resulting deviations of the position coordinates converge below the numerical tolerance of the used optimization solver. The presented experiment shows an accurate execution of the drawing process with the demanded process quality based on the specified process properties. Only with the help of the available tolerance bands, the continuous joint-space trajectory $\mathbf{q}^*(t)$ could be computed, which demonstrates the feasibility of the proposed optimization-based path planning framework.

6.2. Spraying process

In this section, the proposed optimization-based path planning framework is applied to a spraying process. In this process, the workpiece is mounted on the end-effector of a robot and a stationary spray nozzle is used as the tool which has to be moved along the manufacturing path to perform the process. The spray jet is considered rotationally symmetric, i.e. a rotation of the spray jet around the spray direction does not affect the process quality and is therefore considered as redundant process DoF. This property also appears in other industrial processes like, e.g. robotic milling, drilling or laser cutting. Although this spraying process differs significantly from the

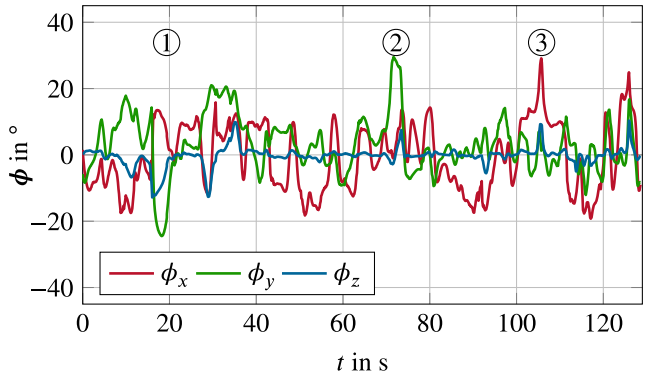


Fig. 11. Orientation tolerances of the TCP frame poses $\mathbf{H}_p^{\mathcal{T}}(\mathbf{q}^*)$ w.r.t. the desired manufacturing path $\mathcal{H}_p^{\mathcal{M}}$ at the drawing process.

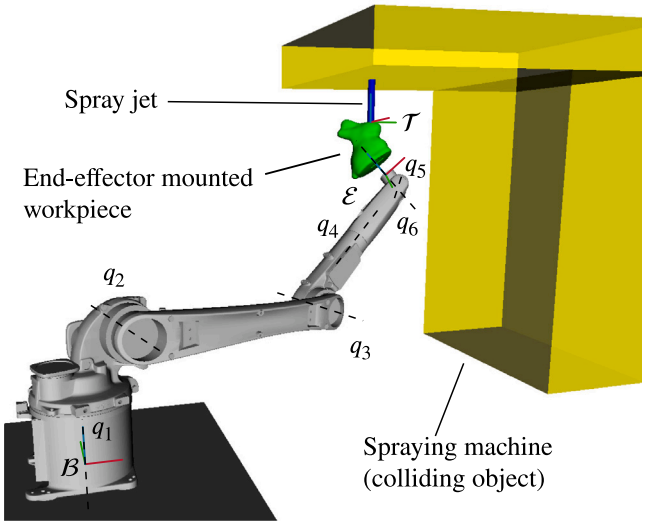


Fig. 12. Setup for the spraying process in simulation. The coordinate systems and robot joints are annotated.

drawing process in Section 6.1, the proposed path planner can be easily adapted to the new process properties. Additionally, long and complex spray paths are challenging path planning problems where the available redundancy has to be exploited again to successfully compute continuous joint-space paths for the process execution.

6.2.1. Spraying process properties

The simulation environment Simulink 3D Animation of the spraying process is shown in Fig. 12 with the robot Kuka Cybertech KR8 R1620. The rabbit-shaped workpiece mounted on the end-effector of the robot is shown in green and the spray machine is depicted as yellow object. The rotationally symmetric spray jet is blue and is considered as redundant DoF in this process. To use collision avoidance from Section 4.4 in the path planning, the spraying machine is embedded in two box objects and collisions with the workpiece are checked. Collisions between the robot and the spraying machine do not occur.

The desired manufacturing path $\mathcal{H}_p^{\mathcal{M}}$ from Fig. 6 has to be followed with the spray jet to perform the spray process on the rabbit-shaped workpiece. Hence, the lateral position (x- and y-direction) of the spray jet has to follow the manufacturing path $\mathcal{H}_p^{\mathcal{M}}$ exactly, while small deviations along the surface normal vector (z-direction) are allowed. The latter only slightly degrades the process quality and is therefore considered as process tolerance. Due to the rotationally symmetric spray nozzle, this process has a redundant process DoF of the orientation around the surface normal vector. In contrast, no rotational deviations around the x- and y-axis are permitted to maintain a circular spray deposit. The process DoF of the spraying process are visualized in Fig. 2b. These process properties are represented by the tolerance bands given by

$$\mathbf{d}_{\max} = \begin{bmatrix} 0\,\mathrm{m} \\ 0\,\mathrm{m} \\ 0.045\,\mathrm{m} \end{bmatrix}, \quad \mathbf{d}_{\min} = \begin{bmatrix} 0\,\mathrm{m} \\ 0\,\mathrm{m} \\ -0.055\,\mathrm{m} \end{bmatrix}, \tag{48a}$$

$$\boldsymbol{\phi}_{\max} = -\boldsymbol{\phi}_{\min} = \begin{bmatrix} 0° \\ 0° \\ 360° \end{bmatrix}. \tag{48b}$$

Based on (48), the optimization problem (13) is tailored to the spraying process by parametrizing the objective function terms and adding the appropriate constraints. The equality constraint of the x- and y-position deviation in (20) is enabled to precisely follow the given manufacturing path $\mathcal{H}_p^{\mathcal{M}}$. Hence, the corresponding position inequality constraints in (21) are disabled and the respective weights of the objective function term (19) are zero. To use the process tolerances along the z-direction of the manufacturing path $\mathcal{H}_p^{\mathcal{M}}$, the equality constraint of the z-position deviation (20) is disabled. Instead, the z-position deviation is used in the objective function term (19) and the corresponding inequality constraint (21) ensures compliance with the defined tolerance band (48). Because no orientation deviation is allowed around the x- and y-axis w.r.t. the manufacturing path $\mathcal{H}_p^{\mathcal{M}}$, the corresponding equality constraints in (28) are enabled and the inequality constraints (30) and objective function terms (26) and (27) are omitted. The redundant process DoF of the spray process, i.e. the orientation around the z-axis, is implemented by setting the respective weight in (27) to zero and disabling the corresponding equality constraints (28) and inequality constraints (30). Thereby, orientation deviations around the z-axis are neither penalized in the optimization problem (13) nor constrained in any way. Collision avoidance is considered between the rabbit-shaped workpiece and the spraying machine with the objective function term (36) and the inequality constraint (37). The objective function terms (40) and (42) are used to obtain continuous joint movements for the robot. The objective function terms, equality and inequality constraints for the spraying process are summarized in Table 2. The individual weights are empirically chosen as diagonal matrices according to the specific process and the robot used.

6.2.2. Simulation results of the spraying process

Similar to the drawing process, an optimization-based algorithm [7] is used to compute the optimal relative position of the robot base frame $\mathcal{B}$ w.r.t. to the stationary TCP frame $\mathcal{T}$ of the spray nozzle. The path planner is parametrized for the spraying process with the objective functions and constraints from Table 2. The series of optimization problems is solved based on the joint-space solutions $\mathcal{Q}_g$ for the first manufacturing path pose $\mathbf{H}_{p,1}^{\mathcal{M}}$, see Fig. 6, which yields the optimal joint-space path $\mathcal{Q}^*$. Note that for this process all considered objective function terms in Table 2 are used to evaluate the optimal joint-space path $\mathcal{Q}^*$ with (45). To obtain a uniform spray coating on the workpiece, the joint-space trajectory $\mathbf{q}^*(t)$ is computed with (46) with a constant path velocity. The joint trajectory $\mathbf{q}^*(t)$ is shown in Fig. 13, where the individual joints $q_h^*(t), h = 1, \ldots, n$ are normalized to the respective axes limits $\mathbf{q}_{\min}$ and $\mathbf{q}_{\max}$. The robot does not reach the mechanical axes limits during the motion and continuously follows the manufacturing path $\mathcal{H}_p^{\mathcal{M}}$. This is possible since the redundant DoF is considered and the robot is able to move through a singular configuration at $t = 63\,\mathrm{s}$, where the joint angle of axis 5 reaches zero. For this application, the mean optimization time on an Intel Core i7-8700K is approximately 36 ms in a single-core implementation and around 9 ms using all cores. The total calculation time is around 60 s with $e_g = 6$ starting configurations. These optimization times are lower compared to the drawing process, which originates from the robot's lower axis count and, hence, the

Table 2
Objective function terms, constraints and weights used for the spraying process.

Variables	Equations	Weights
Position deviation		
$f(\Delta\mathbf{p})$	(19)	$\mathbf{A}_1 = \text{diag}\,(0, 0, 90)$
$c_{\text{eq},x}(\Delta x)$	(20)	Enabled
$c_{\text{eq},y}(\Delta y)$	(20)	Enabled
$c_{\text{eq},z}(\Delta z)$	(20)	Disabled
$c_{\text{ineq},x}(\Delta x)$	(21)	Disabled
$c_{\text{ineq},y}(\Delta y)$	(21)	Disabled
$c_{\text{ineq},z}(\Delta z)$	(21)	Enabled
Orientation deviation		
$f(\Delta\eta)$	(26)	$a = 0$
$f(\Delta\boldsymbol{\varepsilon})$	(27)	$\mathbf{A}_2 = \text{diag}\,(0, 0, 0)$
$\mathbf{c}_{\text{eq}}(\Delta\eta)$	(28)	Disabled
$c_{\text{eq},x}(\Delta\varepsilon_1)$	(28)	Enabled
$c_{\text{eq},y}(\Delta\varepsilon_2)$	(28)	Enabled
$c_{\text{eq},z}(\Delta\varepsilon_3)$	(28)	Disabled
$\mathbf{c}_{\text{ineq}}(\Delta\boldsymbol{\varepsilon})$	(30)	Disabled
Collision avoidance		
$f(l_k)$	(36)	$\mathbf{B} = 10^4\text{diag}\,(1, 1)$
$\mathbf{c}_{\text{ineq}}(l_k)$	(37)	Enabled
Joint limits and path continuity		
$f(\mathbf{q})$	(40)	$\mathbf{C} = \text{diag}\,(5, 5, 5, 5, 5, 5)$
$f(\mathbf{q})$	(42), (44)	$w = 0.05$
Robot Kuka Cybertech KR8 R1620 [51]		
$\mathbf{q}_{\min}^{\mathrm{T}} = -[170°\ 185°\ 137°\ 185°\ 120°\ 350°]$		
$\mathbf{q}_{\max}^{\mathrm{T}} = [170°\ 65°\ 163°\ 185°\ 120°\ 350°]$		

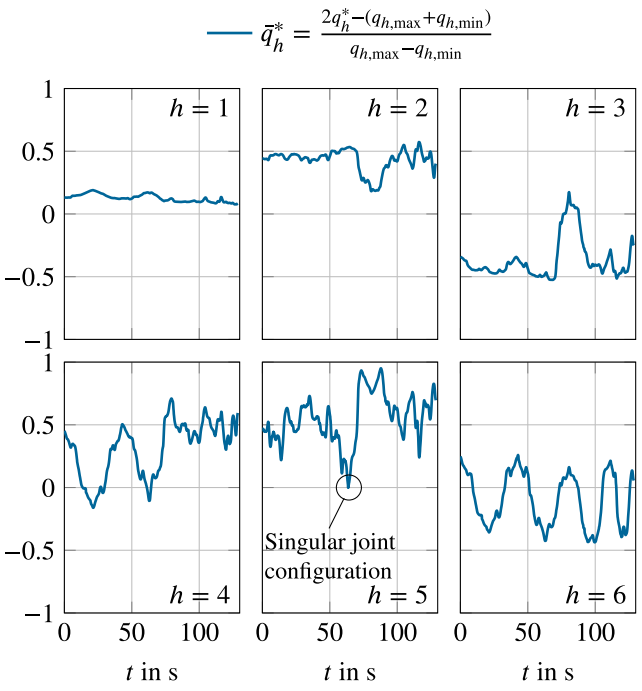


Fig. 13. Optimal joint-space trajectory $\mathbf{q}^*(t)$ for the spraying process. The trajectories of the individual joints are normalized to their respective axes limits $\mathbf{q}_{\min}$ and $\mathbf{q}_{\max}$.

reduction of the optimization variables, see (13). Furthermore, only two collision checks at each optimization are required for the two boxes surrounding the spraying machine. Without collision avoidance checks, an additional 1.5 times reduction of the computing time is possible. A video of the simulation result of the spraying process is shown in www.acin.tuwien.ac.at/4adf/.

The path planner significantly exploits the redundant DoF of the spraying process, i.e. the rotation around the z-axis, and the tolerance band of the z-position to compute a feasible and continuous robot trajectory. In Fig. 14, the position and orientation deviations from the TCP frame poses $\mathbf{H}_p^{\mathcal{T}}(\mathbf{q}^*)$ w.r.t. the desired manufacturing path $\mathcal{H}_p^{\mathcal{M}}$ are presented. Deviations of the x- and y-position are constrained and converge below the numerical tolerance of the used optimization solver. The tolerance band of the z-position is utilized only to a small extent, cf. (48). The orientation deviation of the TCP frame poses $\mathbf{H}_p^{\mathcal{T}}(\mathbf{q}^*)$ w.r.t. the manufacturing path $\mathcal{H}_p^{\mathcal{M}}$ around the x- and y-axis are also constrained and are consistently below the optimizer tolerance. In contrast, rotations around the z-axis of the spray direction, i.e. the redundant DoF, are used extensively to generate a continuous joint-space path $\mathcal{Q}^*$. This experiment shows that the proposed path planner is capable of solving complex path planning problems for various manufacturing processes with long paths by only adapting the objective function terms and constraints based on the required process properties.

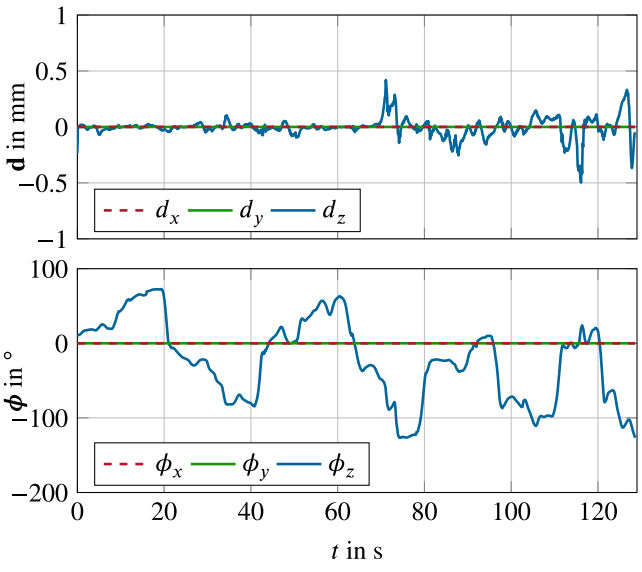


Fig. 14. Position and orientation tolerances of the TCP frame poses $\mathbf{H}_p^{\mathcal{T}}(\mathbf{q}^*(t))$ w.r.t. the desired manufacturing path $\mathcal{H}_p^{\mathcal{M}}$ for the spraying process.

7. Conclusions

In this paper, a novel optimization-based path planning framework for robots is proposed which systematically accounts for all process properties (process tolerances, process windows, constraints, redundant degrees of freedom (DoF)) of the specific manufacturing process and includes a collision avoidance scheme. This allows to find solutions of path planning problems for complex manufacturing paths which cannot be solved with state-of-the-art concepts. By parametrizing the objective function terms and selecting the appropriate equality and inequality constraints, the underlying optimization problem can be easily tailored to the specific needs of the considered manufacturing process. In a first step, the path planner computes multiple joint configurations for the robot based on the pose of the first point on the manufacturing path. Starting from these joint-space solutions, series of optimization problems are solved, which are all independent of each other. This makes it possible to perform a parallel execution of the path planning algorithm on a multi-core CPU. This measure significantly reduces the computing time. Moreover, analytical gradients of the objective

function and of the constraints are provided to further improve the convergence and curb the computing time of the optimization problem. The proposed path planning framework also allows to move the robot through kinematic singularities, which is often avoided in state-of-the-art algorithms that rely on task-space controllers. To demonstrate the feasibility of the proposed approach, a drawing and a spraying process are considered. For the drawing process, the robot is equipped with an end-effector which holds the marker to draw a predefined line on the surface of a workpiece. The marker has a rectangular nib, with which two different line thicknesses can be realized in the process by rotating the marker around the surface normal direction by 90°. However, the orientation of the end-effector is allowed to deviate from the surface normal direction within a specified tolerance band without deteriorating the final drawing quality. For many other industrial processes, like welding or cutting, similar process properties can be defined accordingly. The drawing process was implemented experimentally on a KUKA LBR iiwa 14 R820. Particularly interesting is the fact that 15% of the poses along the drawing path cannot be reached for the nominal path, but the drawing task could be successfully performed with high quality and the desired line thickness in the corresponding segments by exploiting the process tolerances. For the second application, a joint-space path for a simulated spraying process was computed. In this process, the rotation of the spray jet around the surface normal vector is considered a redundant DoF and the distance of the spray nozzle to the surface can vary within a predefined tolerance band. Again, by taking advantage of the redundant DoF, a feasible complex robot path could be planned with the proposed framework. These two applications demonstrate the versatility of the proposed collision-free robotic path planning framework, which can be easily adjusted to different manufacturing processes and is able to systematically account for process tolerances, process windows, and available DoF.

Future works will be concerned with further improving the computation time for the path planning of long manufacturing paths with high resolution and multiple collision objects by using compiled algorithms, different optimization solvers, e.g., [61], and more advanced collision avoidance concepts, e.g., [62]. Additionally, by incorporating the dynamic model of the manipulator into the path planning framework, time-optimal or energy-efficient joint-space trajectories could be obtained. Instead of solving the individual optimization problems sequentially, using graph-based or predictive methods could further enhance the flexibility and the resulting optimal joint-space paths.

CRediT authorship contribution statement

Thomas Weingartshofer: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data curation, Writing – original draft, Writing – review & editing, Visualization. **Bernhard Bischof:** Conceptualization, Methodology, Software. **Martin Meiringer:** Software. **Christian Hartl-Nesic:** Conceptualization, Methodology, Writing – review & editing, Supervision, Project administration. **Andreas Kugi:** Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Appendix

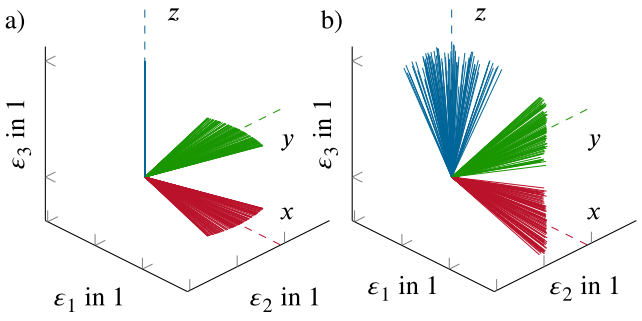


Fig. 15. Orientation deviations $\Delta\mathbf{o}$ w.r.t. the dashed coordinate system in the origin with rotations around (a) the z-axis, (b) the x- and y-axis.

A.1. Cost terms for orientation deviation

The relation between unit quaternions and the angle-axis representation with the vector $\mathbf{r}$ and the rotation θ around $\mathbf{r}$ is given by, see [52]

$$\eta = \cos\left(\frac{\theta}{2}\right) \quad \text{and} \tag{49a}$$

$$\boldsymbol{\epsilon} = \mathbf{r}\sin\left(\frac{\theta}{2}\right) \quad . \tag{49b}$$

In the following two examples with rotations around one axis in (50) and around two axes in (51) are shown, where the specific range of rotation is chosen to clarify the visual presentation in Fig. 15. In Fig. 15a, pure rotations around the z-axis w.r.t. the dashed coordinate system are illustrated, which correspond to unit quaternions $\Delta\mathbf{o}$ in the form

$$\Delta\mathbf{o} = \frac{\Delta\mathbf{o}'}{\|\Delta\mathbf{o}'\|_2}, \quad \Delta\mathbf{o}' = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \varepsilon_3 \end{bmatrix}, \quad -0.15 \leq \varepsilon_3 \leq 0.15\,. \tag{50}$$

In (50), the first two entries of the vector part of the quaternion error $\Delta\boldsymbol{\epsilon}$ are zero while the last entry is a small random number, which corresponds to $\mathbf{r}^{\mathrm{T}} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$. Hence, unit quaternions in the form (50) represent pure rotations around the z-axis.

In contrast, quaternion errors in the form

$$\Delta\mathbf{o} = \frac{\Delta\mathbf{o}'}{\|\Delta\mathbf{o}'\|_2}, \quad \Delta\mathbf{o}' = \begin{bmatrix} 1 \\ \varepsilon_1 \\ \varepsilon_2 \\ 0 \end{bmatrix}, \begin{matrix} -0.15 \leq \varepsilon_1 \leq 0.15 \\ -0.15 \leq \varepsilon_2 \leq 0.15 \end{matrix}, \tag{51}$$

have negligible rotations around the z-axis while the rotations around the x- and y-axis are significant, see Fig. 15b. This example shows that penalizing the individual components of $\Delta\boldsymbol{\epsilon}$ in (27) using $\mathbf{A}_2$, minimizes the rotation around the respective axis in $\mathbf{r}$.

A.2. Gradients of unit quaternions

The time derivative of a unit quaternion is given by, see [52]

$$\dot{\mathbf{o}} = \begin{bmatrix} \dot{\eta} \\ \dot{\boldsymbol{\epsilon}} \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 0 \\ \boldsymbol{\omega} \end{bmatrix} \otimes \begin{bmatrix} \eta \\ \boldsymbol{\epsilon} \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 0 & -\boldsymbol{\omega}^{\mathrm{T}} \\ \boldsymbol{\omega} & \mathbf{S}(\boldsymbol{\omega}) \end{bmatrix}\begin{bmatrix} \eta \\ \boldsymbol{\epsilon} \end{bmatrix}, \tag{52}$$

where $\boldsymbol{\omega}^{\mathrm{T}} = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}$ is the angular velocity and $\mathbf{S}(\boldsymbol{\omega})$ denotes the skew matrix defined as, see [52]

$$\mathbf{S}(\boldsymbol{\omega}) = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \quad . \tag{53}$$

Hence, the time derivative of the quaternion error (25) yields

$$\frac{\mathrm{d}\Delta\mathbf{o}}{\mathrm{d}t} = \begin{bmatrix} \frac{\mathrm{d}\Delta\eta}{\mathrm{d}t} \\ \frac{\mathrm{d}\Delta\boldsymbol{\epsilon}}{\mathrm{d}t} \end{bmatrix} = \frac{\partial\Delta\mathbf{o}}{\partial\mathbf{q}}\dot{\mathbf{q}} = \begin{bmatrix} \frac{\partial\Delta\eta}{\partial\mathbf{q}} \\ \frac{\partial\Delta\boldsymbol{\epsilon}}{\partial\mathbf{q}} \end{bmatrix}\dot{\mathbf{q}} = \frac{1}{2}\begin{bmatrix} -\left(\boldsymbol{\omega}_p^{\mathcal{M}} - \boldsymbol{\omega}_p^{\mathcal{T}}\right)^{\mathrm{T}}\Delta\boldsymbol{\epsilon} \\ \left(\boldsymbol{\omega}_p^{\mathcal{M}} - \boldsymbol{\omega}_p^{\mathcal{T}}\right)\Delta\eta + \left(\boldsymbol{\omega}_p^{\mathcal{M}} + \boldsymbol{\omega}_p^{\mathcal{T}}\right)\times\Delta\boldsymbol{\epsilon} \end{bmatrix}. \tag{54}$$

Eliminating $\dot{\mathbf{q}}$ from (54) leads to

$$\frac{\partial\Delta\mathbf{o}}{\partial\mathbf{q}} = \begin{bmatrix} \frac{\partial\Delta\eta}{\partial\mathbf{q}} \\ \frac{\partial\Delta\boldsymbol{\epsilon}}{\partial\mathbf{q}} \end{bmatrix} = \frac{1}{2}\begin{bmatrix} \Delta\boldsymbol{\epsilon}^{\mathrm{T}}\mathbf{J}_{\mathcal{P},\omega}^{\mathcal{T}}(\mathbf{q}) \\ -\Delta\eta\mathbf{J}_{\mathcal{P},\omega}^{\mathcal{T}}(\mathbf{q}) - \mathbf{S}(\Delta\boldsymbol{\epsilon})\mathbf{J}_{\mathcal{P},\omega}^{\mathcal{T}}(\mathbf{q}) \end{bmatrix}. \tag{55}$$

Thereby, the geometric Jacobian $\mathbf{J}_{\mathcal{P},\omega}^{\mathcal{T}}(\mathbf{q})$ is related to the angular velocity of the contact frame $\mathcal{T}$, see (3), and due to a stationary manufacturing path $\mathcal{H}_p^{\mathcal{M}}$ the corresponding Jacobian $\mathbf{J}_{\mathcal{P},\omega}^{\mathcal{M}}$ vanishes.

References

[1] International Federation of Robotics, World robotics 2020 report, 2020, URL http://reparti.free.fr/robotics2000.pdf, Accessed on 1st July 2022.

[2] M.T. Fralix, From mass production to mass customization, J. Text. Apparel Technol. Manage. 1 (2) (2001) 1–7, URL https://textiles.ncsu.edu/tatm/wp-content/uploads/sites/4/2017/11/fralix_full.pdf, Accessed on 1st July 2022.

[3] W. Terkaj, T. Tolio, A. Valente, A Review on Manufacturing Flexibility, in: Design of Flexible Production Systems, Springer, Berlin Heidelberg, ISBN: 978-3-540-85413-5, 2009, pp. 41–61.

[4] Z. Pan, J. Polden, N. Larkin, S.V. Duin, J. Norrish, Recent progress on programming methods for industrial robots, Robot. Comput.-Integr. Manuf. 28 (2) (2012) 87–94, http://dx.doi.org/10.1016/j.rcim.2011.08.004.

[5] S. Lengagne, N. Ramdani, P. Fraisse, Planning and fast replanning safe motions for humanoid robots, Trans. Robot. 27 (6) (2011) 1095–1106, http://dx.doi.org/10.1109/tro.2011.2162998.

[6] N.C.N. Doan, W. Lin, Optimal robot placement with consideration of redundancy problem for wrist-partitioned 6R articulated robots, Robot. Comput.-Integr. Manuf. 48 (2017) 233–242, http://dx.doi.org/10.1016/j.rcim.2017.04.007.

[7] T. Weingartshofer, C. Hartl-Nesic, A. Kugi, Optimal TCP and robot base placement for a set of complex continuous paths, in: International Conference on Robotics and Automation, IEEE, Xi'an, 2021, pp. 9659–9665, http://dx.doi.org/10.1109/icra48506.2021.9561900.

[8] R.K. Malhan, A.V. Shembekar, A.M. Kabir, P.M. Bhatt, B. Shah, S. Zanio, S. Nutt, S.K. Gupta, Automated planning for robotic layup of composite prepreg, Robot. Comput.-Integr. Manuf. 67 (2021) 102020, http://dx.doi.org/10.1016/j.rcim.2020.102020.

[9] H. Fang, S. Ong, A. Nee, Robot path planning optimization for welding complex joints, Int. J. Adv. Manuf. Technol. 90 (9) (2016) 3829–3839, http://dx.doi.org/10.1007/s00170-016-9684-z.

[10] W. Gao, Q. Tang, J. Yao, Y. Yang, Automatic motion planning for complex welding problems by considering angular redundancy, Robot. Comput.-Integr. Manuf. 62 (2020) 101862, http://dx.doi.org/10.1016/j.rcim.2019.101862.

[11] Q. Wu, J. Lu, W. Zou, D. Xu, Path planning for surface inspection on a robot-based scanning system, in: International Conference on Mechatronics and Automation, IEEE, Beijing, 2015, pp. 2284–2289, http://dx.doi.org/10.1109/icma.2015.7237842.

[12] F. Nagata, Y. Kusumoto, Y. Fujimoto, K. Watanabe, Robotic sanding system for new designed furniture with free-formed surface, Robot. Comput.-Integr. Manuf. 23 (4) (2007) 371–379, http://dx.doi.org/10.1016/j.rcim.2006.04.004.

[13] N. Asakawa, K. Toda, Y. Takeuchi, Automation of chamfering by an industrial robot for the case of hole on free-curved surface, Robot. Comput.-Integr. Manuf. 18 (5-6) (2002) 379–385, http://dx.doi.org/10.1016/s0736-5845(02)00006-6.

[14] W. Sheng, H. Chen, N. Xi, Y. Chen, Tool path planning for compound surfaces in spray forming processes, Trans. Autom. Sci. Eng. 2 (3) (2005) 240–249, http://dx.doi.org/10.1109/tase.2005.847739.

[15] H. Chen, T. Fuhlbrigge, X. Li, Automated industrial robot path planning for spray painting process: A review, in: International Conference on Automation Science and Engineering, IEEE, Arlington, 2008, pp. 522–527, http://dx.doi.org/10.1109/coase.2008.4626515.

[16] Q. Yu, G. Wang, K. Chen, A robotic spraying path generation algorithm for free-form surface based on constant coating overlapping width, in: International Conference on Cyber Technology in Automation, Control, and Intelligent Systems, IEEE, Shenyang, 2015, pp. 1045–1049, http://dx.doi.org/10.1109/cyber.2015.7288089.

[17] A. Kharidege, D.T. Ting, Z. Yajun, A practical approach for automated polishing system of free-form surface path generation based on industrial arm robot, Int. J. Adv. Manuf. Technol. 93 (9) (2017) 3921–3934, http://dx.doi.org/10.1007/s00170-017-0726-y.

[18] Z. Kingston, M. Moll, L.E. Kavraki, Sampling-based methods for motion planning with constraints, Annu. Rev. Control Robot. Auton. Syst. 1 (2018) 159–185, http://dx.doi.org/10.1146/annurev-control-060117-105226.

[19] I.A. Sucan, M. Moll, L.E. Kavraki, The open motion planning library, Robot. Autom. Mag. 19 (4) (2012) 72–82, http://dx.doi.org/10.1109/mra.2012.2205651.

[20] K. Hauser, V. Ng-Thow-Hing, Fast smoothing of manipulator trajectories using optimal bounded-acceleration shortcuts, in: International Conference on Robotics and Automation, IEEE, Anchorage, 2010, pp. 2493–2498, http://dx.doi.org/10.1109/robot.2010.5509683.

[21] J.D. Maeyer, B. Moyaers, E. Demeester, Cartesian path planning for arc welding robots: Evaluation of the descartes algorithm, in: International Conference on Emerging Technologies and Factory Automation, IEEE, Limassol, 2017, pp. 1–8, http://dx.doi.org/10.1109/etfa.2017.8247616.

[22] R. Smits, KDL: Kinematics and Dynamics Library, Orocos, 2022, URL http://www.orocos.org/kdl, Accessed on 1st July 2022.

[23] R. Diankov, Automated Construction of Robotic Manipulation Programs (Ph.D. thesis), Carnegie Mellon University, The Robotics Institute, Pittsburgh, 2010, URL http://www.programmingvision.com/rosen_diankov_thesis.pdf, Accessed on 1st July 2022.

[24] D. Bertram, J. Kuffner, R. Dillmann, T. Asfour, An integrated approach to inverse kinematics and path planning for redundant manipulators, in: International Conference on Robotics and Automation, IEEE, Orlando, 2006, pp. 1874–1879, http://dx.doi.org/10.1109/robot.2006.1641979.

[25] M. Elbanhawi, M. Simic, Sampling-based robot motion planning: A review, IEEE Access 2 (2014) 56–77, http://dx.doi.org/10.1109/access.2014.2302442.

[26] K. Hauser, Y. Zhou, Asymptotically optimal planning by feasible kinodynamic planning in a state–cost space, Trans. Robot. 32 (6) (2016) 1431–1443, http://dx.doi.org/10.1109/tro.2016.2602363.

[27] J. Schulman, J. Ho, A. Lee, I. Awwal, H. Bradlow, P. Abbeel, Finding locally optimal, collision-free trajectories with sequential convex optimization, in: Robotics: Science and Systems, Vol. IX, RSS, Berlin, 2013, pp. 1–10, http://dx.doi.org/10.15607/rss.2013.ix.031.

[28] N. Ratliff, M. Zucker, J.A. Bagnell, S. Srinivasa, CHOMP: Gradient optimization techniques for efficient motion planning, in: International Conference on Robotics and Automation, IEEE, Kobe, 2009, pp. 489–494, http://dx.doi.org/10.1109/robot.2009.5152817.

[29] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, S. Schaal, STOMP: Stochastic trajectory optimization for motion planning, in: International Conference on Robotics and Automation, IEEE, Shanghai, 2011, pp. 4569–4574, http://dx.doi.org/10.1109/icra.2011.5980280.

[30] N.C.N. Doan, P.Y. Tao, W. Lin, Optimal redundancy resolution for robotic arc welding using modified particle swarm optimization, in: International Conference on Advanced Intelligent Mechatronics, IEEE, Banff, 2016, pp. 554–559, http://dx.doi.org/10.1109/aim.2016.7576826.

[31] M. Gadaleta, M. Pellicciari, G. Berselli, Optimization of the energy consumption of industrial robots for automatic code generation, Robot. Comput.-Integr. Manuf. 57 (2019) 452–464, http://dx.doi.org/10.1016/j.rcim.2018.12.020.

[32] C.G.L. Bianco, A. Piazzi, A genetic/interval approach to optimal trajectory planning of industrial robots under torque constraints, in: European Control Conference, IEEE, Karlsruhe, 1999, pp. 942–947, http://dx.doi.org/10.23919/ecc.1999.7099428.

[33] I. Gentilini, K. Nagamatsu, K. Shimada, Cycle time based multi-goal path optimization for redundant robotic systems, in: International Conference on Intelligent Robots and Systems, IEEE, Tokyo, 2013, pp. 1786–1792, http://dx.doi.org/10.1109/iros.2013.6696591.

[34] J. Kim, E.A. Croft, Online near time-optimal trajectory planning for industrial robots, Robot. Comput.-Integr. Manuf. 58 (2019) 158–171, http://dx.doi.org/10.1016/j.rcim.2019.02.009.

[35] A. Gasparetto, V. Zanotto, Optimal trajectory planning for industrial robots, Adv. Eng. Softw. 41 (4) (2010) 548–556, http://dx.doi.org/10.1016/j.advengsoft.2009.11.001.

[36] J. Huang, P. Hu, K. Wu, M. Zeng, Optimal time-jerk trajectory planning for industrial robots, Mech. Mach. Theory 121 (2018) 530–544, http://dx.doi.org/10.1016/j.mechmachtheory.2017.11.006.

[37] F. Rubio, C. Llopis-Albert, F. Valero, J.L. Suñer, Industrial robot efficient trajectory generation without collision through the evolution of the optimal trajectory, Robot. Auton. Syst. 86 (2016) 106–112, http://dx.doi.org/10.1016/j.robot.2016.09.008.

[38] T. Chettibi, H. Lehtihet, M. Haddad, S. Hanchi, Minimum cost trajectory planning for industrial robots, Eur. J. Mech. A Solids 23 (4) (2004) 703–715, http://dx.doi.org/10.1016/j.euromechsol.2004.02.006.

[39] J. Polden, Z. Pan, N. Larkin, S. van Duin, Adaptive partial shortcuts: Path optimization for industrial robotics, J. Intell. Robot. Syst. 86 (1) (2016) 35–47, http://dx.doi.org/10.1007/s10846-016-0437-x.

[40] A. Aristidou, J. Lasenby, Y. Chrysanthou, A. Shamir, Inverse kinematics techniques in computer graphics: A survey, Comput. Graph. Forum 37 (6) (2017) 35–58, http://dx.doi.org/10.1111/cgf.13310.

[41] K. Hauser, Continuous Pseudoinversion of a Multivariate Function: Application to Global Redundancy Resolution, in: Algorithmic Foundations of Robotics XII. Springer Proceedings in Advanced Robotics, Springer, Cham, ISBN: 978-3-030-43089-4, 2020, pp. 496–511.

[42] P. Beeson, B. Ames, TRAC-IK: An open-source library for improved solving of generic inverse kinematics, in: International Conference on Humanoid Robots (Humanoids), IEEE, Seoul, 2015, pp. 928–935, http://dx.doi.org/10.1109/humanoids.2015.7363472.

[43] K. Hauser, Learning the problem-optimum map: Analysis and application to global optimization in robotics, Trans. Robot. 33 (1) (2017) 141–152, http://dx.doi.org/10.1109/tro.2016.2623345.

[44] L.-C. Wang, C. Chen, A combined optimization method for solving the inverse kinematics problems of mechanical manipulators, IEEE Trans. Robot. Autom. 7 (4) (1991) 489–499, http://dx.doi.org/10.1109/70.86079.

[45] P. Ruppel, N. Hendrich, S. Starke, J. Zhang, Cost functions to specify full-body motion and multi-goal manipulation tasks, in: International Conference on Robotics and Automation, IEEE, Brisbane, 2018, pp. 3152–3159, http://dx.doi.org/10.1109/icra.2018.8460799.

[46] M. Kang, H. Shin, D. Kim, S.-E. Yoon, TORM: Fast and accurate trajectory optimization of redundant manipulator given an end-effector path, in: International Conference on Intelligent Robots and Systems, IEEE, Las Vegas, 2020, pp. 9417–9424, http://dx.doi.org/10.1109/iros45743.2020.9341358.

[47] R. Holladay, O. Salzman, S. Srinivasa, Minimizing task-space Fréchet error via efficient incremental graph search, Robot. Autom. Lett. 4 (2) (2019) 1999–2006, http://dx.doi.org/10.1109/lra.2019.2899668.

[48] M. Faroni, M. Beschi, N. Pedrocchi, A. Visioli, Predictive inverse kinematics for redundant manipulators with task scaling and kinematic constraints, Trans. Robot. 35 (1) (2019) 278–285, http://dx.doi.org/10.1109/tro.2018.2871439.

[49] D. Rakita, B. Mutlu, M. Gleicher, Relaxedik: Real-time synthesis of accurate and feasible robot arm motion, in: Robotics: Science and Systems, Vol. XIV, RSS, Pittsburgh, 2018, pp. 26–30, http://dx.doi.org/10.15607/rss.2018.xiv.043.

[50] P. Praveena, D. Rakita, B. Mutlu, M. Gleicher, User-guided offline synthesis of robot arm motion from 6-DoF paths, in: International Conference on Robotics and Automation, IEEE, Montreal, 2019, pp. 8825–8831, http://dx.doi.org/10.1109/icra.2019.8793483.

[51] KUKA Deutschland GmbH, Spez KR CYBERTECH nano V2, 2018.

[52] B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo, Robotics: Modelling, Planning and Control, first ed., Springer, London, ISBN: 978-1-84628-642-1, 2009.

[53] B. Mirtich, V-Clip: fast and robust polyhedral collision detection, Trans. Graph. 17 (3) (1998) 177–208, http://dx.doi.org/10.1145/285857.285860.

[54] H. Anton, C. Rorres, Elementary Linear Algebra, eleventh ed., Wiley, New Jersey, ISBN: 978-1-119-62569-8, 2019.

[55] R.A. Horn, C.R. Johnson, Matrix Analysis, second ed., Cambridge University Press, New York, ISBN: 978-0-521-83940-2, 2013.

[56] J. Nocedal, S. Wright, Numerical Optimization, second ed., in: Springer Series in Operations Research and Financial Engineering, Springer, New York, ISBN: 978-0387-30303-1, 2006.

[57] R.H. Byrd, J.C. Gilbert, J. Nocedal, A trust region method based on interior point techniques for nonlinear programming, Math. Program. 89 (1) (2000) 149–185, http://dx.doi.org/10.1007/pl00011391.

[58] R. Waltz, J. Morales, J. Nocedal, D. Orban, An interior algorithm for nonlinear optimization that combines line search and trust region steps, Math. Program. 107 (3) (2005) 391–408, http://dx.doi.org/10.1007/s10107-004-0560-5.

[59] F.N. Fritsch, R.E. Carlson, Monotone piecewise cubic interpolation, SIAM J. Numer. Anal. 17 (2) (1980) 238–246, http://dx.doi.org/10.1137/0717021.

[60] KUKA Robot GmbH, BA LBR iiwa V5, 2015.

[61] A. Wächter, L.T. Biegler, On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming, Math. Program. 106 (1) (2005) 25–57, http://dx.doi.org/10.1007/s10107-004-0559-y.

[62] M. Montanari, N. Petrinic, OpenGJK for C, C# and Matlab: Reliable solutions to distance queries between convex bodies in three-dimensional space, SoftwareX 7 (2018) 352–355, http://dx.doi.org/10.1016/j.softx.2018.10.002.