Robotics and Computer–Integrated Manufacturing 73 (2022) 102223

Contents lists available at ScienceDirect

Robotics and Computer-Integrated Manufacturing

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

Full length article

Real-time motion control of robotic manipulators for safe human–robot coexistence

Kelly Merckaert [a,b,*], Bryan Convens [a,c], Chi-ju Wu [d], Alessandro Roncone [d], Marco M. Nicotra [e], Bram Vanderborght [a,c]

[a] Robotics and Multibody Mechanics (R&MM), Department of Mechanical Engineering, Vrije Universiteit Brussel, Brussels, Belgium
[b] Flanders Make, Leuven, Belgium
[c] Imec, Leuven, Belgium
[d] Human Interaction and RObotics (HIRO), Department of Computer Science, University of Colorado Boulder, Boulder, CO 80309, USA
[e] Robotics, Optimization, and Constrained Control (ROCC), Department of Electrical, Computer, and Energy Engineering, University of Colorado Boulder, Boulder, CO 80309, USA

ARTICLE INFO

Keywords:
Human–robot collaboration
Collision avoidance
Constrained control
Robot arm

ABSTRACT

This paper introduces a computationally efficient control scheme for safe human–robot interaction. The method relies on the Explicit Reference Governor (ERG) formalism to enforce input and state constraints in real-time, thus ensuring that the robot can safely operate in close proximity to humans. The resulting constrained control method can steer the robot arm to the desired end-effector pose (or a steady-state admissible approximation thereof) in the presence of actuator saturation, limited joint ranges, speed limits, static obstacles, and humans. The effectiveness of the proposed solution is supported by theoretical results and numerous experimental validations on the Franka Emika Panda robotic manipulator, a commercially available collaborative 7-DOF robot arm.

1. Introduction

In recent years, the paradigm of manufacturing is shifting from mass production to mass customization with high-mix low-volume production. To this end, an increased flexibility in the production environment is required, which can be obtained by combining the complementary qualities of humans and robots. On the one hand, robots excel at simple, repetitive tasks; on the other hand, humans have unique cognitive skills for understanding and adapting to any changes in the task [1,2]. While traditional industrial robots are typically segregated within safety cages, collaborative robots, or cobots, are able to work directly in the proximity of human operators, sharing the same workspace and performing combined operations. By doing so, they can increase flexibility and productivity while also improving the ergonomics of the workplace [3].

Although Human–Robot Collaboration (HRC) can bring the production line to a new level of flexibility and efficiency, the safety issue of robots working in close proximity and without barriers with human operators becomes dominant [4]. The publication of the ISO/TS 15066 directives on this matter defined the safety functions and performance of collaborative robots, within four types of collaborative robot operation: Safety Monitored Stop (SMS), Hand Guiding (HG), Speed and Separation Monitoring (SSM), and Power and Force Limiting (PFL). SMS pauses a robot's motion while an operator is in the collaborative workspace, HG allows a collaborative robot to move through direct input from an operator, in SSM the collaborative robot is able to move concurrently with the operator as long as they maintain a predetermined distance apart, and PFL requires a special robot that has power or force feedback built in which lets the collaborative robot detect contact with a person [5]. Although a robot in the SSM operation mode, is not necessarily brought to standstill, the robot moves very slowly in proximity to the human operator. This dramatically reduces the performance of the robot in terms of productivity, ultimately jeopardizing the economic attractiveness of a collaborative workstation. This is especially true since a significant part of the tasks in today's industry consists of handling and moving objects, operations that do not really add value to a product and should therefore be performed as quickly and seamlessly as possible. The execution time is therefore critical and even relatively small savings on the robot motions may influence the overall cycle time of a system, and can be the difference between whether a given setup is economically feasible or not [6]. To boost their return on investment, control approaches have to be revised

* Corresponding author.
E-mail address: kelly.merckaert@vub.be (K. Merckaert).

https://doi.org/10.1016/j.rcim.2021.102223
Received 18 January 2021; Received in revised form 22 July 2021; Accepted 23 July 2021
Available online 5 August 2021

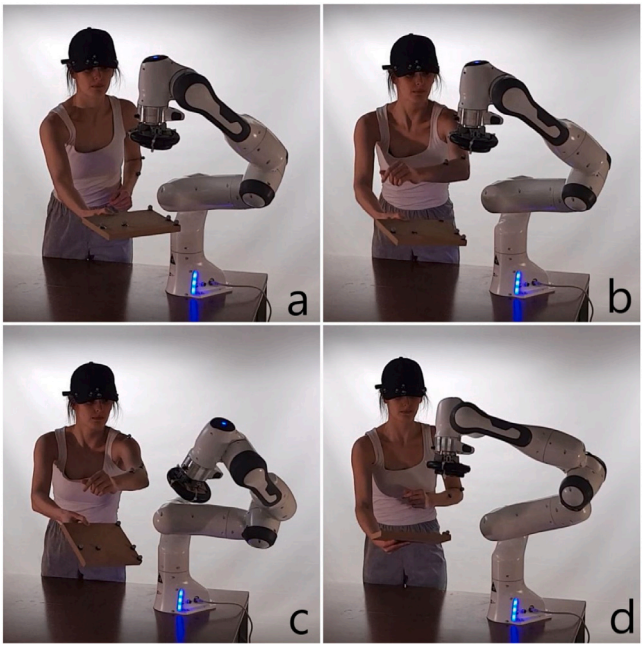


Fig. 1. Experimental Validation of the Trajectory-Based ERG. (a) the robot's end-effector pose reference is defined using a wooden plate with reflective markers, (b) the arm of the human operator gets in between the robot and its reference, (c) the robot safely moves away from the human, (d) the robot goes back to its reference once it is safe to do so. See https://youtu.be/UzbhMzcKSbE for the full video.

and low-level motion control of the cobots needs to be modulated online and in real-time to improve safety and performance while keeping the productivity high.

In this work, we aim to reduce the current drawbacks related to the SSM operating mode. We focus on highly dynamic environments with a robot that operates in close proximity with a human operator and needs to avoid the unpredictable human motions. We develop a provably safe real-time constrained control methodology that can be implemented on a generic serial chain robotic manipulator to guarantee efficient task realization and safety towards the human and the environment. The algorithm is validated on the Franka Emika Panda (hereafter denoted as the Panda robot), and videoframes from the extensive experimental validation are shown in Fig. 1.

2. Related work

To achieve safe Human–Robot Collaboration (HRC), most industrial robot manipulators are re-engineered with either passively compliant actuators, [7], actively compliant actuators [8,9], or collision sensors [10].

The *feeling of safety* and comfort of the human operator can be improved in HRC tasks by making it easier for the human to interpret the robot's intents, which can be achieved by making the robot motions more human-like [11]. However, it is important to remark that this strategy cannot guarantee safety without accounting for state and actuator input constraints [12]. A combination of both requirements is achieved in [13], which proposes a motion planner that generates consistent trajectories for similar tasks, making it easier for the human to predict the robot's actions, while it *guarantees safety* by enforcing constraints.

Sampling-based motion planning algorithms, as Rapidly-exploring Random Trees (RRT), are used for planning safe motions of robotic manipulators with kinematic constraints, e.g. [6,14,15]. Although all variants have their own advantages, none of them take into account the robot dynamics. Path planning approaches that consider both kinematic and dynamic constraints are computationally demanding and therefore hard to implement in real-time, e.g. [16], where solution trajectories for cluttered environments are found after more than 3 min.

A motion planning scheme based on quadratic programming for redundant robot manipulators that allows to track complex end-effector paths with low joint-angle drift under convex constraints is proposed in [17] and extended to nonconvex constraints in [18]. The control scheme considers joint angle and joint velocity limits, but does not account for higher-order manipulator dynamics and torque input constraints, neither does it provide real-time collision avoidance capabilities.

In contrast to the latter motion planning algorithms, potential field methods are reactive and computationally efficient, making them particularly well-suited for real-time collision avoidance, see e.g. [19–24]. Another computationally efficient method is the danger field, which indicates how dangerous are the current posture and velocity of the robot with respect to the objects in the environment [25]. The two approaches can be combined using the kinetostatic safety field [26]. However, all these approaches do not take into account actuator input constraints (i.e. the joint torques), also called input saturation constraints.

The Saturation in the Null Space algorithm generates velocity commands that allows real-time control of robots for a large number of hard limits [27]. Although this iterative algorithm guarantees an optimal solution, it also does not take into account actuator input constraints.

A general control solution able to handle both state and input constraints in real-time is Model Predictive Control (MPC), which is based on the idea of solving a constrained finite time optimization problem at each sampling time [28,29]. Although recent advancements in computational performances have made it possible to implement MPC on robots, see e.g. [30,31], the application possibilities are limited due to the typically non-negligible computational cost.

For robot control in dynamic environments where the human operator can be the dynamic obstacle, strategies based on the SSM and the PFL criterion change the velocity of the robot based on the distance between the human operator and the robot [32–35]. Similarly, in [36] gesture recognition of the operator commands is used to change the workmode and so the velocity of the robot. However, these velocity modulations strategies have usually no rigorous proof of convergence in finite time and often slow down the robot excessively.

Methodologies that guarantee obstacle avoidance under certain assumptions in HRC environments are pseudo-distance algorithms, that are based on distance calculations of the dynamic environment. Despite the fact that the experiments in [37] indicate that the proposed method can perform safe and timely dynamic avoidance for redundant manipulators, the case that obstacles are continuously moving is not considered with their discrete detection algorithm. In [38] a safe set of states including dynamic constraints is determined, which is then rendered controlled positively invariant, thus keeping the system in a safe configuration. Instead of generating invariant sets, another way to tackle the HRC safety problem is to generate reachable occupancies which account for all human movement, as in [39] where reachable occupancy and a kinematic model of a human is used to ensure that the robot avoids the human before coming to a stop. Although safe, these methods result in a more conservative robot behavior.

To the best of our knowledge, the literature does not provide any control techniques with high provable safety that achieve real-time control of a robotic manipulator without dramatically reducing the performance of the robot in the presence of actuator saturation, joint range limitations, speed constraints, obstacles, and moving objects.

Contributions

This work is based on the Explicit Reference Governor (ERG), which is a closed-form feedback control scheme that can enforce both state and input constraints of nonlinear pre-stabilized systems without

having to solve an online optimization problem [40,41]. The idea of the trajectory-based ERG for a generic robotic manipulator with actuator constraints, joint angle limitations, and static spherical obstacles was explored in [42]. However, the method was validated only numerically for a 2-DOF robotic manipulator. The core contributions of this article are listed below.

(1) The paper considers a broader range of constraints. Notably, we include auxiliary speed limits, i.e. the robot joint velocities and the Cartesian end-effector velocity, as well as wall and cylindrical obstacles.
(2) The paper discusses stability and constraint reinforcement in presence of dynamic constraints.
(3) The control law is validated experimentally on a commercially available 7-DOF robot, i.e. the Panda robot. The experimental validation includes case studies in static environments, where it is possible to guarantee the absence of collisions.
(4) The experimental validations also include dynamic obstacles with a human in a HRC experiment, where it is not possible to guarantee the absence of collisions. Although the robot will always try to avoid collisions with obstacles, collisions with dynamic obstacles can happen as a result of the environment (e.g. human) hitting the robot, as opposed to the other way around.

This paper is organized as follows. The problem is formulated in Section 3. The proposed control framework is explained in Section 4. The results are presented in Section 5 and discussed in Section 6. Conclusions and future perspectives are given in Section 7.

3. Problem formulation

We present the dynamic model of a robotic manipulator, the constraints to which it is subject, and the control objectives.

3.1. Dynamic model

Consider the joint space dynamic model of a robotic manipulator with n joints,

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau}\,, \tag{1}$$

where $\boldsymbol{q} \in \mathbb{R}^n$ is the vector of joint variables, $\boldsymbol{M}(\boldsymbol{q}) > 0$ is the mass matrix, $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}}$ accounts for the Coriolis and centrifugal forces, $\boldsymbol{g}(\boldsymbol{q})$ is the influence of the gravity on the manipulator, and $\boldsymbol{\tau} \in \mathbb{R}^n$ is the torque control input vector.

We will denote $\boldsymbol{x}_i = [\boldsymbol{p}_i, \boldsymbol{\Theta}_i]^T$ as the Cartesian pose of the i-th joint $\forall i = \{1,\dots,n\}$, with $\boldsymbol{p}_i = [x_i, y_i, z_i]^T \in \mathbb{R}^3$ the position of joint i and $\boldsymbol{\Theta}_i \in \mathbb{H}$ representing the orientation of joint i in quaternion space. Although the pose of the end-effector can be denoted as $\boldsymbol{x}_{n+1}$, in this work we use the notation $\boldsymbol{x}_e$. Similarly, the end-effector's twist will be denoted as $\dot{\boldsymbol{x}}_e \in \mathbb{R}^6$, with $\dot{\boldsymbol{p}}_e = [\dot{x}_e, \dot{y}_e, \dot{z}_e] \in \mathbb{R}^3$ the linear velocity and $\boldsymbol{\omega}_e = [\omega_{e,x}, \omega_{e,y}, \omega_{e,z}] \in \mathbb{R}^3$ the angular velocity.

We will furthermore define as $\boldsymbol{k}_i(\cdot) : \mathbb{R}^n \mapsto \mathbb{R}^3$ the direct kinematics to obtain the position of joint i, i.e. $\boldsymbol{p}_i = \boldsymbol{k}_i(\boldsymbol{q})$, $i = \{1,\dots,n+1\}$. We omit the orientation part since the direct kinematics will only be computed to obtain the positions of the joints. We define the Jacobian of the robotic manipulator $\boldsymbol{J}_i(\cdot) : \mathbb{R}^i \mapsto \mathbb{R}^3$ so that $\dot{\boldsymbol{p}}_i = \boldsymbol{J}_i(\boldsymbol{q}_{1:i})\dot{\boldsymbol{q}}_{1:i}$, where we only take into account the translation part of the Jacobian. This is not a limitation, since we only use the direct kinematics and the Jacobian to compute the distance between an obstacle and the robot arm, for which the joint positions are enough to obtain all necessary information of the robot link position and orientation. However, for the inverse kinematics, both the position and orientation of the end-effector are taken into account.

3.2. State and input constraints

The system is subject to a variety of input and state constraints. Specifically, we consider the following classes of constraints:

3.2.1. Actuator saturation

Whenever one of the joint motors is subject to saturation, the control law is unable to generate an arbitrary torque vector, which can lead to oscillatory or even unstable robotic manipulator motions. To prevent this scenario, each motor torque is required to stay within its lower and upper saturation limits,

$$\tau_{\min,i} \le \tau_i \le \tau_{\max,i}, \quad \forall i = \{1,\dots,n\}\,. \tag{2}$$

3.2.2. Operating region

The robot arm has a limited operating range,

$$q_{\min,i} \le q_i \le q_{\max,i}, \quad \forall i = \{1,\dots,n\}\,, \tag{3}$$

which represents the range constraints of the n actuators.

3.2.3. Speed limitation

To allow human–robot collaboration, we have to take into account the cobot's inherent joint velocity limits,

$$\dot{q}_{\min,i} \le \dot{q}_i \le \dot{q}_{\max,i}, \quad \forall i = \{1,\dots,n\}\,, \tag{4}$$

and its Cartesian end-effector velocity limits,

$$\dot{\boldsymbol{x}}_{\min,e} \le \dot{\boldsymbol{x}}_e \le \dot{\boldsymbol{x}}_{\max,e}\,. \tag{5}$$

For the purposes of constrained control algorithms, these limits are not strictly necessary to ensure provably safe robot motions. However, many industrial robots include fail-safe mechanisms that activate when the joint velocities or end-effector velocities are too fast. Omitting these constraints can therefore cause the robot to stop unexpectedly.

3.2.4. Obstacles

It is assumed that the robot should avoid a collection of possibly moving obstacles. We will consider N_w walls, N_s spherical obstacles, and N_c cylindrical obstacles.

Wall Constraints: Planar walls can be avoided by enforcing

$$\boldsymbol{c}_j^\mathrm{w} \cdot \boldsymbol{p}_i \le d_j^\mathrm{w}, \quad \forall i \in \{1,\dots,n+1\},\ \forall j \in \{1,\dots,N_\mathrm{w}\}\,, \tag{6}$$

where $\boldsymbol{c}_j^\mathrm{w} \in \mathbb{R}^3$ is the unit vector normal to the j-th wall (pointing in the inadmissible direction) and $d_j^\mathrm{w} \in \mathbb{R}$ describing the distance between the plane and the robot base.

Spherical Obstacles: Collision with spherical obstacles can be avoided by enforcing

$$\|\boldsymbol{p}_{ij}^\mathrm{s} - \boldsymbol{c}_j^\mathrm{s}\| \ge r_j^\mathrm{s}, \quad \forall i \in \{1,\dots,n\},\ \forall j \in \{1,\dots,N_\mathrm{s}\}, \tag{7}$$

where $\boldsymbol{c}_j^\mathrm{s}$ and r_j^s are the center and the radius of sphere j, respectively, and $\boldsymbol{p}_{ij}^\mathrm{s}$ is the point on the robot link i that is closest to sphere j. See Appendix A for details on how to compute $\boldsymbol{p}_{ij}^\mathrm{s}$.

Cylindrical Obstacles: Collision with cylindrical obstacles can be avoided by enforcing

$$\|\boldsymbol{p}_{ij}^\mathrm{c} - \boldsymbol{t}_{ij}^\mathrm{c}\| \ge r_j^\mathrm{c}, \quad \forall i \in \{1,\dots,n\},\ \forall j \in \{1,\dots,N_\mathrm{c}\}, \tag{8}$$

where r_j^c is the radius of cylinder j, $\boldsymbol{p}_{ij}^\mathrm{c}$ is the point on link i that is closest to cylinder j and $\boldsymbol{t}_{ij}^\mathrm{c}$ is the point on cylinder j that is closest to link i. See Appendix B for details on how to compute $\boldsymbol{p}_{ij}^\mathrm{c}$ and $\boldsymbol{t}_{ij}^\mathrm{c}$.

In the context of HRC, it is worth noting that a robot workspace can be limited by using virtual walls and that typical workspace objects or the human body (e.g. head, limbs) can be approximated using spherical and cylindrical geometries.

3.3. Control objectives

The main objective of this paper is to design a computationally efficient control scheme for a robotic manipulator with the joint space dynamic model in (1) that has to reach an end-effector pose reference $\boldsymbol{x}_{e,r}$, or a steady-state admissible approximation, while also satisfying the listed constraints. The following objectives need to be achieved.

3.3.1. Safety

For any piece-wise continuous reference $\boldsymbol{q}_r(t)$, the control law guarantees constraint satisfaction of the state and input constraints (2)–(5). Moreover, the satisfaction of the collision constraints (6)–(8) is guaranteed in the case of static obstacles. For the case of dynamic obstacles, the control law avoids collision if possible and halts the robot otherwise.

3.3.2. Asymptotic stability

If the reference $\boldsymbol{q}_r$ is constant and steady-state admissible, the closed-loop system satisfies $\lim_{t\to\infty} \boldsymbol{q}(t) = \boldsymbol{q}_r$;

3.3.3. Reactiveness

The control law must run in real-time without relying on offline pre-generated trajectories.

It is worth noting that the *Safety* objective is consistent with the ISO/TS 15066 directives, even though it does not prevent collision in every scenario. Specifically, it is acceptable for a motionless robot to be hit by a human operator due to the relatively low energy exchange of such collisions.

4. Proposed control framework

The main control challenge is to ensure that the nonlinear dynamics of the robotic manipulator satisfy the numerous input and state constraints. Based on this problem statement, we propose a multi-layer control architecture to decouple the control problem into more manageable sub-tasks. The proposed control framework is illustrated in Fig. 2.

The first task, which is handled by the ***Control Layer***, consists in pre-stabilizing the dynamics of the robotic manipulator to the applied reference $\boldsymbol{q}_v$. As employed in most commercially available manipulators, this will be done by a classic PD control law with gravity compensation that does not account for any system constraints. The second task, which is handled by the ***trajectory-based Explicit Reference Governor (ERG)***, consists in dynamically filtering the reference $\boldsymbol{q}_r$ so that all the constraints are satisfied. This layer is also responsible for reaching the target configuration $\boldsymbol{q}_r$. In case a Cartesian end-effector pose reference $\boldsymbol{x}_{e,r}$ is given to the robotic manipulator, it needs to be transformed by means of a kinematic inversion into a joint reference $\boldsymbol{q}_r$. To do so, we will use an inverse kinematics algorithm based on Newton–Raphson iterations.

The detailed design of the control layer and trajectory-based ERG will be addressed in Sections 4.1 and 4.2, respectively.

4.1. Pre-stabilizing control layer

The goal of the control layer is to pre-stabilize the robotic manipulator without accounting for system constraints. This is achieved by the classic PD with gravity compensation approach

$$\boldsymbol{\tau} = \boldsymbol{K}_P(\boldsymbol{q}_v - \boldsymbol{q}) - \boldsymbol{K}_D\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q})\,. \tag{9}$$

Combined with (1) this leads to the following closed-loop system dynamics

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} = \boldsymbol{K}_P(\boldsymbol{q}_v - \boldsymbol{q}) - \boldsymbol{K}_D\dot{\boldsymbol{q}}\,. \tag{10}$$

In absence of constraints it is possible to prove with the Lyapunov function,

$$V\left(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{q}_v\right) = \frac{1}{2}\dot{\boldsymbol{q}}^T\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{q}} + \frac{1}{2}(\boldsymbol{q}_v - \boldsymbol{q})^T\boldsymbol{K}_P(\boldsymbol{q}_v - \boldsymbol{q})\,, \tag{11}$$

that the equilibrium configuration $\boldsymbol{q} = \boldsymbol{q}_v$, $\dot{\boldsymbol{q}} = \boldsymbol{0}$ of (10) is Globally Asymptotically Stable (GAS) whenever $\boldsymbol{K}_P$ and $\boldsymbol{K}_D$ are positive definite diagonal matrices [43].

4.2. Trajectory-based explicit reference governor

Consider the pre-stabilized system (10) such that, if the applied reference $\boldsymbol{q}_v$ remains constant, the closed-loop equilibrium configuration $\bar{\boldsymbol{q}}_v$ is asymptotically stable. Given a continuous steady-state admissible path $\boldsymbol{\Phi} : [0,1] \mapsto \mathbb{R}^7$ between an initial reference $\boldsymbol{\Phi}(0) = \boldsymbol{q}_v(0)$ and a target reference $\boldsymbol{\Phi}(1) = \boldsymbol{q}_r$, the principle behind the ERG is to generate a reference $\boldsymbol{q}_v(t) \in \{\boldsymbol{\Phi}(s) \mid s \in [0,1]\}$ such that the transient dynamics of the closed-loop system cannot cause a constraint violation and $\lim_{t\to\infty} \boldsymbol{q}_v(t) = \boldsymbol{\Phi}(1)$.

Rather than pre-computing a suitable trajectory $\boldsymbol{q}_v(t)$, the ERG achieves these objectives by continuously manipulating the derivative of the applied reference as the product of $\rho\left(\boldsymbol{q}_v,\boldsymbol{q}_r\right)$ and $\Delta\left(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{q}_v\right)$,

$$\dot{\boldsymbol{q}}_v = \rho\left(\boldsymbol{q}_v,\boldsymbol{q}_r\right)\,\Delta\left(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{q}_v\right)\,, \tag{12}$$

where $\rho\left(\boldsymbol{q}_v,\boldsymbol{q}_r\right)$ is the **Navigation Field** (NF), i.e. a vector field that generates the desired steady-state admissible path $\boldsymbol{\Phi}(s)$, and $\Delta\left(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{q}_v\right)$ is the **Dynamic Safety Margin** (DSM), i.e. a scalar that quantifies the distance between the predicted transient dynamics of the pre-stabilized system and the constraint boundaries if the current $\boldsymbol{q}_v(t)$ were to remain constant.

For more details on the theoretical properties of the ERG framework, the reader is referred to [41]. The following sections detail the NF and the DSM used in this work.

4.2.1. Navigation field

The Navigation Field can be designed using a traditional attraction and repulsion field,

$$\rho\left(\boldsymbol{q}_v,\boldsymbol{q}_r\right) = \rho^{\text{att}} + \rho^{\text{rep}}\,, \tag{13}$$

where the attraction field is

$$\rho^{\text{att}}\left(\boldsymbol{q}_r,\boldsymbol{q}_v\right) = \frac{\boldsymbol{q}_r - \boldsymbol{q}_v}{\max\left(\left\|\boldsymbol{q}_r - \boldsymbol{q}_v\right\|,\eta\right)}\,, \tag{14}$$

with $\eta > 0$ a smoothing parameter ensuring ρ^{att} is a class C^1 function.

The repulsion field is the sum of linear repulsion fields pushing the robot arm away from steady-state inadmissible regions because of joint angles constraints (q), wall constraints (w), spherical obstacles (s), and cylindrical obstacles (c), i.e.

$$\rho^{\text{rep}} = \rho^{\text{q}} + \rho^{\text{w}} + \rho^{\text{s}} + \rho^{\text{c}}\,. \tag{15}$$

For the repulsion field due to the joint angle constraints $\boldsymbol{q}_{\min} \le \boldsymbol{q} \le \boldsymbol{q}_{\max}$ it is enough to add for each actuator $i \in \{1,\dots,n\}$ a repulsion term

$$\rho_i^{\text{q}} = \max\left(\frac{\zeta^{\text{q}} - \left|q_{v,i} - q_{\min,i}\right|}{\zeta^{\text{q}} - \delta^{\text{q}}}, 0\right) - \max\left(\frac{\zeta^{\text{q}} - \left|q_{v,i} - q_{\max,i}\right|}{\zeta^{\text{q}} - \delta^{\text{q}}}, 0\right), \tag{16}$$

where $\delta^q > 0$ is the static safety margin of all the joint angles and $\zeta^q > \delta^q$ is the influence margin. Assuming that the influence regions of the upper and lower joint angle limits do not overlap, at least one term in (16) will always be zero for each actuator i. We will denote the joint repulsion field as

$$\boldsymbol{\rho}^{\text{q}} = [\rho_1^{\text{q}},\dots,\rho_n^{\text{q}}]^T\,. \tag{17}$$

For the repulsion field due to the obstacle constraints, we first need to compute the Cartesian position of each joint angle i, with $i = \{1,\dots,n+1\}$, for the applied reference configuration $\boldsymbol{q}_v$,

$$\boldsymbol{p}_{v,i} = \boldsymbol{k}_i(\boldsymbol{q}_v)\,. \tag{18}$$

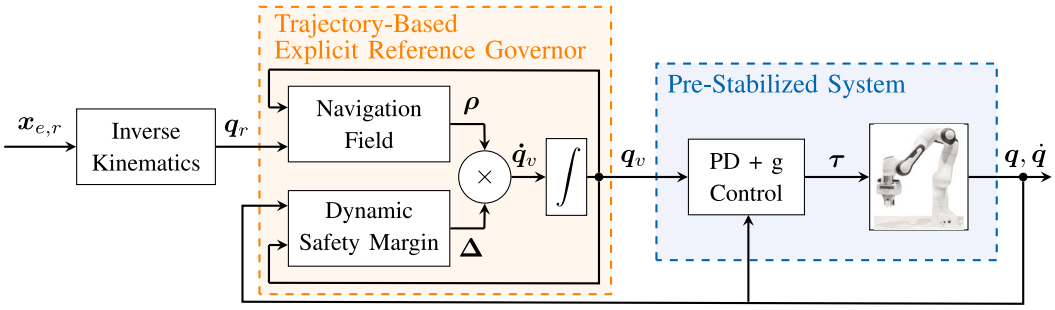


Fig. 2. Proposed Constrained Control Architecture – The desired end-effector pose $\boldsymbol{x}_{e,r}$ is transformed by means of *inverse kinematics* into a desired joint reference $\boldsymbol{q}_r$. This desired joint reference $\boldsymbol{q}_r$ is dynamically filtered by the *trajectory-based ERG* to the auxiliary joint reference $\boldsymbol{q}_v$ such that the target configuration will be reached while satisfying the system constraints. The trajectory-based ERG is the product of a *Navigation Field*, which determines the direction of $\dot{\boldsymbol{q}}_v$, and a *Dynamic Safety Margin*, which regulates the modulus of $\dot{\boldsymbol{q}}_v$. The *PD+g control unit* pre-stabilizes the robot dynamics to the applied reference $\boldsymbol{q}_v$ and sends the computed torques $\boldsymbol{\tau}$ to the robot.

To avoid collisions with the wall constraints $j = \{1, \dots, N_\mathrm{w}\}$, we first denote the direction of the repulsion field for wall j in Cartesian space by

$$\boldsymbol{v}_{ij}^\mathrm{w} = -\boldsymbol{c}_j^\mathrm{w}\,, \tag{19}$$

which points in the admissible direction. Note that the direction of the repulsion field due to wall j is the same for all robot links i, $\forall i \in \{1, \dots, n\}$. By using the pseudoinverse Jacobian, we can transform this repulsion field to joint space,

$$\hat{\boldsymbol{\rho}}_{ij}^\mathrm{w} = \frac{\boldsymbol{J}_i^\dagger(\boldsymbol{q}_{v,1:i})\boldsymbol{v}_{ij}^\mathrm{w}}{\max\left(\left\|\boldsymbol{J}_i^\dagger(\boldsymbol{q}_{v,1:i})\boldsymbol{v}_{ij}^\mathrm{w}\right\|, \eta^\mathrm{w}\right)}\,, \tag{20}$$

with the smoothing parameter $\eta^\mathrm{w} = 10^{-3}$. This is a unit direction vector in joint space for the endpoint of link i and wall j. The amplitude of this vector depends on the distance between the endpoint of link i and wall j,

$$\boldsymbol{\rho}_{ij}^\mathrm{w} = \max\left(\frac{\zeta^\mathrm{w} - \left(d_j^\mathrm{w} - \boldsymbol{c}_j^\mathrm{w} \cdot \boldsymbol{p}_{v,i}\right)}{\zeta^\mathrm{w} - \delta^\mathrm{w}}, 0\right)\hat{\boldsymbol{\rho}}_{ij}^\mathrm{w}\,, \tag{21}$$

with ζ^w the influence margin and δ^w the safety margin. The repulsion field for the full robot arm and for N_w walls becomes

$$\boldsymbol{\rho}^\mathrm{w} = \sum_{j=1}^{N_\mathrm{w}}\sum_{i=2}^{n+1}\boldsymbol{\rho}_{ij}^\mathrm{w}\,, \tag{22}$$

in which we only take into account the endpoint of all the links, whereby the base joint, i.e. $i = 1$, is neglected.

To avoid collisions with the spherical obstacles $j = \{1, \dots, N_\mathrm{s}\}$, we compute $\boldsymbol{p}_{v,ij}^\mathrm{s}$ as detailed in Appendix A.[1] The direction of the repulsion field for link i and spherical obstacle j can then be computed in task space as

$$\boldsymbol{v}_{ij}^\mathrm{s} = \boldsymbol{p}_{v,ij}^\mathrm{s} - \boldsymbol{c}_j^\mathrm{s}\,. \tag{23}$$

[1] Appendix A, the subscript v is omitted for simplicity of notation. However, when computing the NF, all the joint positions $\boldsymbol{p}_i$ should be interpreted as the joint positions of the applied reference configuration, i.e. $\boldsymbol{p}_{v,i}$.

This repulsion field is transformed to joint space by using the pseudoinverse Jacobian,[2]

$$\hat{\boldsymbol{\rho}}_{ij}^\mathrm{s} = \frac{\boldsymbol{J}_{\boldsymbol{p}_{v,ij}^\mathrm{s}}^\dagger(\boldsymbol{q}_v)\,\boldsymbol{v}_{ij}^\mathrm{s}}{\max\left(\left\|\boldsymbol{J}_{\boldsymbol{p}_{v,ij}^\mathrm{s}}^\dagger(\boldsymbol{q}_v)\,\boldsymbol{v}_{ij}^\mathrm{s}\right\|, \eta^\mathrm{s}\right)}\,, \tag{24}$$

with the smoothing parameter $\eta^\mathrm{s} = 10^{-3}$. This represents a unit direction vector in joint space from sphere j to link i. The amplitude of this vector depends on the shortest distance between link i and sphere j,

$$\boldsymbol{\rho}_{ij}^\mathrm{s} = \max\left(\frac{\zeta^\mathrm{s} - \left(\left\|\boldsymbol{p}_{v,ij}^\mathrm{s} - \boldsymbol{c}_j^\mathrm{s}\right\| - r_j^\mathrm{s}\right)}{\zeta^\mathrm{s} - \delta^\mathrm{s}}, 0\right)\hat{\boldsymbol{\rho}}_{ij}^\mathrm{s}\,, \tag{25}$$

with ζ^s the influence margin and δ^s the static safety margin. The resulting repulsion field for the full robot arm and for N_s spheres is

$$\boldsymbol{\rho}^\mathrm{s} = \sum_{j=1}^{N_\mathrm{s}}\sum_{i=1}^{n}\boldsymbol{\rho}_{ij}^\mathrm{s}\,. \tag{26}$$

To avoid collisions with the cylindrical obstacles $j = \{1, \dots, N_\mathrm{c}\}$, we compute $\boldsymbol{p}_{v,ij}^\mathrm{c}$ and $\boldsymbol{t}_{v,ij}^\mathrm{c}$ as detailed in Appendix B. The direction of the repulsion field for link i and cylindrical obstacle j can then be computed as

$$\boldsymbol{v}_{ij}^\mathrm{c} = \boldsymbol{p}_{v,ij}^\mathrm{c} - \boldsymbol{t}_{v,ij}^\mathrm{c}\,. \tag{27}$$

This repulsion field is transformed to joint space by using the pseudoinverse Jacobian,[3]

$$\hat{\boldsymbol{\rho}}_{ij}^\mathrm{c} = \frac{\boldsymbol{J}_{\boldsymbol{p}_{v,ij}^\mathrm{c}}^\dagger(\boldsymbol{q}_v)\,\boldsymbol{v}_{ij}^\mathrm{c}}{\max\left(\left\|\boldsymbol{J}_{\boldsymbol{p}_{v,ij}^\mathrm{c}}^\dagger(\boldsymbol{q}_v)\,\boldsymbol{v}_{ij}^\mathrm{c}\right\|, \eta^\mathrm{c}\right)}\,, \tag{28}$$

with the smoothing parameter $\eta^\mathrm{c} = 10^{-3}$. This represents a unit direction vector in joint space from cylinder j to link i. The amplitude

[2] The translational Jacobian for point $\boldsymbol{p}_{v,ij}^\mathrm{s}$ and the applied joint angle configuration $\boldsymbol{q}_v$ is denoted by $\boldsymbol{J}_{\boldsymbol{p}_{v,ij}^\mathrm{s}}^\dagger(\boldsymbol{q}_v)$. However, in practice we can only obtain the numerical values of the Jacobian for the n joints and end-effector. Therefore, (24) becomes in practice, $\hat{\boldsymbol{\rho}}_{ij}^\mathrm{s} = \frac{\left(\left(1-\lambda_{ij}^\mathrm{s}\right)\boldsymbol{J}_i^\dagger(\boldsymbol{q}_{v,1:i})+\lambda_{ij}^\mathrm{s}\boldsymbol{J}_{i+1}^\dagger(\boldsymbol{q}_{v,1:i+1})\right)\boldsymbol{v}_{ij}^\mathrm{s}}{\max\left(\left\|\left(\left(1-\lambda_{ij}^\mathrm{s}\right)\boldsymbol{J}_i^\dagger(\boldsymbol{q}_{v,1:i})+\lambda_{ij}^\mathrm{s}\boldsymbol{J}_{i+1}^\dagger(\boldsymbol{q}_{v,1:i+1})\right)\boldsymbol{v}_{ij}^\mathrm{s}\right\|,\eta^\mathrm{s}\right)}$.

[3] The translational Jacobian for point $\boldsymbol{p}_{v,ij}^\mathrm{c}$ and the applied joint angle configuration $\boldsymbol{q}_v$ is denoted by $\boldsymbol{J}_{\boldsymbol{p}_{v,ij}^\mathrm{c}}^\dagger(\boldsymbol{q}_v)$. However, in practice we can only obtain the numerical values of the Jacobian for the n joints and end-effector. Therefore, (28) becomes in practice, $\hat{\boldsymbol{\rho}}_{ij}^\mathrm{c} = \frac{\left(\left(1-\lambda_{ij}^\mathrm{c}\right)\boldsymbol{J}_i^\dagger(\boldsymbol{q}_{v,1:i})+\lambda_{ij}^\mathrm{c}\boldsymbol{J}_{i+1}^\dagger(\boldsymbol{q}_{v,1:i+1})\right)\boldsymbol{v}_{ij}^\mathrm{c}}{\max\left(\left\|\left(\left(1-\lambda_{ij}^\mathrm{c}\right)\boldsymbol{J}_i^\dagger(\boldsymbol{q}_{v,1:i})+\lambda_{ij}^\mathrm{c}\boldsymbol{J}_{i+1}^\dagger(\boldsymbol{q}_{v,1:i+1})\right)\boldsymbol{v}_{ij}^\mathrm{c}\right\|,\eta^\mathrm{c}\right)}$.

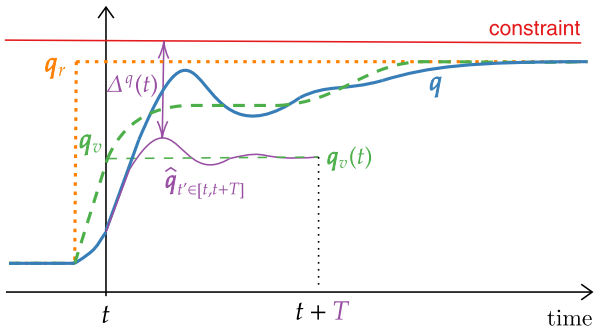


Fig. 3. Trajectory-Based DSM Principle – An arbitrary reference $\boldsymbol{q}_r$ is given to the trajectory-based ERG, which returns the feasible reference $\boldsymbol{q}_v$ that can be applied to the system without causing any constraint violations, resulting in the real robot motion $\boldsymbol{q}$. Before time t, a step reference is reached by the robot (i.e. $\boldsymbol{q}_r = \boldsymbol{q}_v = \boldsymbol{q}$), whereafter another step reference $\boldsymbol{q}_r$ is given to the system. Starting from the current robot states $(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t))$ at time t, the trajectory-based DSM predicts the robot dynamics $\hat{\boldsymbol{q}}_{t'\in[t,t+T]}$ over a finite time horizon T for a constant applied reference $\boldsymbol{q}_v(t)$. The minimal distance between the predicted states and the constraint is denoted by $\Delta^q(t)$ and denotes the guaranteed safe rate of change of the applied reference, i.e. $\dot{\boldsymbol{q}}_v$. In a similar way constraints on other variables can be accounted for.

of this vector depends on the shortest distance between link i and cylinder j,

$$\boldsymbol{\rho}_{ij}^{\mathrm{c}} = \max\left(\frac{\zeta^{\mathrm{c}} - \left(\left\|\boldsymbol{p}_{v,ij}^{\mathrm{c}} - \boldsymbol{t}_{v,ij}^{\mathrm{c}}\right\| - r_j^{\mathrm{c}}\right)}{\zeta^{\mathrm{c}} - \delta^{\mathrm{c}}}, 0\right)\hat{\boldsymbol{\rho}}_{ij}^{\mathrm{c}}, \tag{29}$$

with ζ^{c} the influence margin and δ^{c} the static safety margin. The resulting repulsion field for the full robot arm and for N_{c} cylinders is

$$\boldsymbol{\rho}^{\mathrm{c}} = \sum_{j=1}^{N_{\mathrm{c}}}\sum_{i=1}^{n}\boldsymbol{\rho}_{ij}^{\mathrm{c}}. \tag{30}$$

4.2.2. Dynamic safety margin

The idea behind the trajectory-based DSM is to compute the trajectories of the pre-stabilized system $\hat{\boldsymbol{q}}(t|\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{q}_v) = \hat{\boldsymbol{q}}_{t'=t}$ under the assumption that the current applied reference $\boldsymbol{q}_v(t)$ is kept constant. This is done by simulating the forward dynamics with the Simplectic Euler. We first initialize at the current time t the states that will be predicted, $\hat{\boldsymbol{q}}$ and $\hat{\dot{\boldsymbol{q}}}$,

$$\begin{cases}\hat{\dot{\boldsymbol{q}}}_{t'=t} = \dot{\boldsymbol{q}}(t),\\ \hat{\boldsymbol{q}}_{t'=t} = \boldsymbol{q}(t),\end{cases} \tag{31}$$

whereafter we simulate the system dynamics,

$$\begin{cases}\hat{\boldsymbol{\tau}}_{t'+dt} = \boldsymbol{K}_P\left(\boldsymbol{q}_v - \hat{\boldsymbol{q}}_{t'}\right) - \boldsymbol{K}_D\hat{\dot{\boldsymbol{q}}}_{t'} + \boldsymbol{g}(\hat{\boldsymbol{q}}_{t'}),\\ \hat{\ddot{\boldsymbol{q}}}_{t'+dt} = \boldsymbol{M}^{\dagger}(\hat{\boldsymbol{q}}_{t'})\left(\hat{\boldsymbol{\tau}}_{t'+dt} - \boldsymbol{C}(\hat{\boldsymbol{q}}_{t'},\hat{\dot{\boldsymbol{q}}}_{t'}) - \boldsymbol{g}(\hat{\boldsymbol{q}}_{t'})\right),\\ \hat{\dot{\boldsymbol{q}}}_{t'+dt} = \hat{\dot{\boldsymbol{q}}}_{t'} + \hat{\ddot{\boldsymbol{q}}}_{t'+dt}\,dt,\\ \hat{\boldsymbol{q}}_{t'+dt} = \hat{\boldsymbol{q}}_{t'} + \hat{\dot{\boldsymbol{q}}}_{t'+dt}\,dt,\end{cases} \tag{32}$$

with dt the prediction sampling time. Given the trajectory $\hat{\boldsymbol{q}}(t'|\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{q}_v)$ over a finite time window $t' \in [t, t+T]$ with $T > 0$, which is sufficiently large to capture the system dynamics, it is possible to compute the distance between the predicted inputs and states, and their constraints. When the minimal distance of all those constraint boundaries is positive, it is possible to change the currently applied reference $\boldsymbol{q}_v$ without running the risk of violating the constraints in the future. The smaller this worst case distance becomes, the slower the applied reference may change. When this predicted distance is zero, the applied reference is not allowed to change, since this could cause constraint violations in the future. The principle behind the DSM is illustrated in Fig. 3.

The overall trajectory-based DSM Δ_T for all the constraints listed in Section 3.2, can be computed as.

$$\Delta_T(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{q}_v) = \min\left\{\begin{matrix}\kappa^{\tau}\Delta^{\tau}, \kappa^{q}\Delta^{q}, \kappa^{\dot{q}}\Delta^{\dot{q}}, \kappa^{\dot{x}_e}\Delta^{\dot{x}_e},\\ \kappa^{\mathrm{W}}\Delta^{\mathrm{W}}, \kappa^{\mathrm{S}}\Delta^{\mathrm{S}}, \kappa^{\mathrm{C}}\Delta^{\mathrm{C}}\end{matrix}\right\} \tag{33}$$

with κ^{τ}, κ^{q}, $\kappa^{\dot{q}}$, $\kappa^{\dot{x}_e}$, κ^{W}, κ^{S}, and κ^{C} all arbitrary positive scaling factors.

For the DSM due to the actuator saturation constraints, we have to compute

$$\Delta^{\tau} = \min_{t'\in[t,t+T]}\left\{\min_{i\in[1,n]}\left\{\hat{\tau}_{t',i} - \tau_{\min,i}, \tau_{\max,i} - \hat{\tau}_{t',i}\right\}\right\}. \tag{34}$$

The DSM due to joint angle constraints becomes

$$\Delta^{q} = \min_{t'\in[t,t+T]}\left\{\min_{i\in[1,n]}\left\{\hat{q}_{t',i} - q_{\min,i}, q_{\max,i} - \hat{q}_{t',i}\right\}\right\}. \tag{35}$$

The DSM due to the speed limits is

$$\Delta^{\dot{q}} = \min_{t'\in[t,t+T]}\left\{\min_{i\in[1,n]}\left\{\hat{\dot{q}}_{t',i} - \dot{q}_{\min,i}, \dot{q}_{\max,i} - \hat{\dot{q}}_{t',i}\right\}\right\}, \tag{36}$$

$$\Delta^{\dot{x}_e} = \min_{t'\in[t,t+T]}\left\{\hat{\dot{x}}_{t',e} - \dot{x}_{\min,e}, \dot{x}_{\max,e} - \hat{\dot{x}}_{t',e}\right\}. \tag{37}$$

The DSM due to the obstacle constraints can be computed as,

$$\Delta^{\mathrm{W}} = \min_{j\in[1,N_{\mathrm{w}}]}\left\{\min_{i\in[1,n]}\left\{\min_{t'\in[t,t+T]}\left\{\left\|d_j^{\mathrm{W}} - \boldsymbol{c}_j^{\mathrm{W}}\cdot\hat{\boldsymbol{p}}_{t',i}\right\|\right\}\right\}\right\}, \tag{38}$$

$$\Delta^{\mathrm{S}} = \min_{j\in[1,N_{\mathrm{s}}]}\left\{\min_{i\in[1,n]}\left\{\min_{t'\in[t,t+T]}\left\{\left\|\hat{\boldsymbol{p}}_{t',ij}^{\mathrm{S}} - \boldsymbol{c}_j^{\mathrm{S}}\right\| - r_j^{s}\right\}\right\}\right\}, \tag{39}$$

$$\Delta^{\mathrm{C}} = \min_{j\in[1,N_{\mathrm{c}}]}\left\{\min_{i\in[1,n]}\left\{\min_{t'\in[t,t+T]}\left\{\left\|\hat{\boldsymbol{p}}_{t',ij}^{\mathrm{C}} - \hat{\boldsymbol{t}}_{t',ij}^{\mathrm{C}}\right\| - r_j^{c}\right\}\right\}\right\}. \tag{40}$$

Note that the predicted $\hat{\boldsymbol{p}}_{t',ij}^{\mathrm{S}}$, $\hat{\boldsymbol{p}}_{t',ij}^{\mathrm{C}}$, and $\hat{\boldsymbol{t}}_{t',ij}^{\mathrm{C}}$ are computed in Appendices A and B, with the only difference that we do not consider the applied joint angle configuration $\boldsymbol{q}_v$, but the predicted joint angle configuration $\hat{\boldsymbol{q}}_{t'}$.

To extend this result over an infinite time horizon, it is sufficient to ensure that, from time $t+T$ onward, that the closed-loop system dynamics will not exceed the terminal energy constraint,

$$\Delta_V(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{q}_v) = \kappa^{\mathrm{E_{term}}}\left(\mathrm{E_{term}} - V\left(\hat{\boldsymbol{q}}_{t'},\hat{\dot{\boldsymbol{q}}}_{t'},\boldsymbol{q}_v\right)\right), \tag{41}$$

where $\kappa^{\mathrm{E_{term}}}$ is a positive arbitrary scaling factor, $\mathrm{E_{term}} > 0$ is a suitable threshold value for which the tuning will be further explained in Section 5.2, and the predicted closed-loop system energy is given by

$$V(\hat{\boldsymbol{q}}_{t'},\hat{\dot{\boldsymbol{q}}}_{t'},\boldsymbol{q}_v) = \frac{1}{2}\hat{\dot{\boldsymbol{q}}}_{t'}\boldsymbol{M}(\hat{\boldsymbol{q}}_{t'})\hat{\dot{\boldsymbol{q}}}_{t'} + \frac{1}{2}\left(\boldsymbol{q}_v - \hat{\boldsymbol{q}}_{t'}\right)^T\boldsymbol{K}_P\left(\boldsymbol{q}_v - \hat{\boldsymbol{q}}_{t'}\right), \tag{42}$$

for $t' = t+T$. By combining (33) and (41) we obtain the overall DSM for an infinite time horizon,

$$\Delta(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{q}_v) = \max\left\{\min\left(\Delta_T,\Delta_V\right),0\right\}. \tag{43}$$

Note that, to compute the DSMs, we assume that the obstacles are static.

4.2.3. Theoretical guarantees

It is worth noting that, although the ERG design does not differentiate between static and dynamic constraints, the behavior of the ERG will be different for the two cases. Given **static constraints**, the ERG guarantees *constraint satisfaction* and *asymptotic stability* to a local attractor of the navigation field, as detailed in [41]. In the presence of **dynamic constraints**, the navigation field becomes time-varying. As a result, it is not possible to guarantee asymptotic convergence to a local attractor. Due to the DSM, however, we can still guarantee *stability* since the rate of change $\dot{\boldsymbol{q}}_v$ is proportional to $\Delta\left(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{q}_v\right)$, which goes to zero if the robot torques, angular velocity, or terminal energy exceed their limits. Moreover, although the ERG has the overall tendency to

push the robot away from moving obstacles, we are unable to guarantee collision avoidance in every scenario. Nevertheless, if the ERG detects that a collision is immanent (i.e. if the forward trajectory predictions result in $\Delta = 0$), the applied reference will stop changing. Given a constant reference, (9) reduces to a compliant controller [44], with $\boldsymbol{K}_P$ regulating the overall stiffness. Thus, the ERG satisfies the *Power and Force Limiting criterion*, since the robot is passive at the time of collision.

In short, we can guarantee that the robot is never unstable and that it is asymptotically stable if the environment around it remains stationary. Additionally, we show that the robot will not deliberately cause a collision and displays a compliant behavior if something in the environment bumps into it.

5. Results

We present the results of an experimental validation of the trajectory-based ERG by means of experiments without obstacles, with static obstacles, and with dynamic obstacles (i.e. a human). A video of the experiments can be found at https://youtu.be/UzbhMzcKSbE. A summary of these results can be found in Section 6.

5.1. Experimental setup

The experiments are performed with the Panda collaborative robot. The pre-stabilizing control and trajectory-based ERG of Sections 4.1 and 4.2 are implemented in C++ on a desktop with an AMD® Ryzen 9 3900x 12-core processor x 24. The pre-stabilizing control runs at 1 kHz, which is the frequency at which commands should be sent to the robot. The ERG runs in parallel at 100 Hz, which has shown to be sufficiently fast for the purposes of these demonstrations.

Experiments that require online localization of the end-effector reference or online localization of the dynamic obstacles are executed with the Vicon Nexus software in a Vicon motion capture system which sends data at 100 Hz. The Orocos Kinematics and Dynamics Library [45] is used for kinematic inversion for experiments where an end-effector reference pose is given to the robot.

The experimental validation is based on pick and place experiments, in which we do not study how to grasp objects, but observe the robot dynamics in between consecutive reference positions. To make the influence of the robot dynamics more significant, we replaced the standard Panda end-effector of 0.7 kg with a mass of 2.5 kg (i.e. 60% of Panda's maximum payload) and attached it with tie wraps to the flange. The updated mass matrix, Coriolis force vector, and gravity vector are obtained by the C++ library functions provided by Franka Emika.

Panda's actuator saturation, operating region, and speed limitation values, as listed in Section 3.2, are given by Franka Emika [46], i.e. torque limits $\boldsymbol{\tau}_{\text{limit}} = [87.0, 87.0, 87.0, 87.0, 12.0, 12.0, 12.0]$ Nm, joint angle lower limits $\boldsymbol{q}_{\min} = [-2.8973, -1.7628, -2.8973, -3.0718, -2.8973, -0.0175, -2.8973]$ rad, joint angle upper limits $\boldsymbol{q}_{\max} = [2.8973, 1.7628, 2.8973, -0.0698, 2.8973, 3.7525, 2.8973]$ rad, joint velocity limits $\dot{\boldsymbol{q}}_{\text{limit}} = [2.1750, 2.1750, 2.1750, 2.1750, 2.6100, 2.6100, 2.6100]$ rad/s, and Cartesian velocity limits $\dot{\boldsymbol{x}}_{e,\text{limit}} = [1.7\,\text{m/s}, 2.5\,\text{rad/s}]$.

5.2. Tuning guidelines

Here, we list guidelines for the tuning of the main parameters of the pre-stabilizing control layer and the trajectory-based ERG and how this relates to the obtained performance and robustness.

(1) First tune the pre-stabilizing control loop gains $\boldsymbol{K}_P > 0$ and $\boldsymbol{K}_D > 0$ for stable regulation control performance. This step is accomplished without worrying about the effect on any of the input or state constraints. The stiffer the pre-stabilized closed-loop system is tuned, the more aggressive (e.g. including some overshoots) the robot behavior will be far away from constraints. If the operator is likely to actively come into contact with the robot, then it is better to make the pre-stabilizing control-loop more compliant, by reducing $\boldsymbol{K}_P$.

(2) Choose a prediction time horizon that can take into account the system dynamics. The longer the prediction horizon, the faster the system will be able to react to eventual constraint violations in the further future, but also the higher the computational cost will be.

(3) Eliminate numerical noise in the attraction field that can occur when $\boldsymbol{q}_v$ is very close to $\boldsymbol{q}_r$ by selecting a strictly positive, but suitably small value for the smoothing parameter η. If η is chosen too large, the attraction field will weaken too early, slowing the convergence to the reference configuration, i.e. $\lim_{t\to\infty} \boldsymbol{q}_v(t) = \boldsymbol{q}_r$.

(4) Eliminate numerical noise in the repulsion field that can occur when the distance between one of the robot links and one of the obstacles becomes small by selecting a strictly positive, but small, value for the smoothing parameters η^{w}, η^{s}, and η^{c}. Typically, these parameters are chosen smaller than η.

(5) Increase the DSM gains κ until no further performance increase is obtained. These gains are chosen such that the DSMs of the active constraints have the same order of magnitude.

(6) Choose a terminal energy constraint E_{term} by letting the robot move in an obstacle-free environment at increasing speeds with the robot's constraints included in the ERG while measuring the closed-loop system energy. Take the maximum closed-loop system energy for which the robot does not incur into a hard fail-safe.

(7) Choose medium influence margins ζ defining from how far the obstacles are considered in the repulsion field. Too large values will require too large sensing ranges for obstacles, whereas too small values will decrease the reaction time too much.

(8) Choose strictly positive static safety margins δ to increase robustness. This also ensures the NF's repulsion term achieves its maximum amplitude while the DSM stays strictly positive. Hence this allows moving (and not blocking) the reference in directions pointing outward the obstacle constraint.

In all the experiments, the control gains of the pre-stabilizing control detailed in Section 4.1 are $\boldsymbol{K}_P = \text{diag}(120.0, 120.0, 120.0, 100.0, 50.0, 45.0, 15.0)$ and $\boldsymbol{K}_D = \text{diag}(8.0, 8.0, 8.0, 5.0, 2.0, 2.0, 1.0)$, giving moderately aggressive performance. The smoothing parameter of the attraction field defined in Section 4.2 is chosen as $\eta = 0.05$, the prediction sampling time is fixed to 2 ms, and with 100 prediction samples we predict over a time horizon of 200 ms. The terminal energy constraint is set to $\text{E}_{\text{term}} = 12$ J. Other parameters defined in Section 4.2 are specified in the following sections.

5.3. Obstacle-free environment

In case a large step reference is given to the pre-stabilized robot in an obstacle-free environment, the robot will stop immediately due to an effort or velocity constraint violation. To avoid any constraint violation in an obstacle-free environment, we take into account the actuator saturation, operating region, and speed limitation constraints. The ERG parameters used in this experiment are $\kappa^{\tau} = 9.0$, $\delta^{q} = 0.1$ rad, $\zeta^{q} = 0.15$ rad, $\kappa^{q} = 115.0$, $\kappa^{\dot{q}} = 70.0$, $\kappa^{\dot{x}_e} = 70.0$, and $\kappa^{\text{E}_{\text{term}}} = 7.5$.

The robot starts in the initial configuration $\boldsymbol{q}(t \le 0.5\,\text{s}) = \left[0, \frac{-\pi}{4}, 0, \frac{-3\pi}{4}, 0, \frac{\pi}{2}, \frac{\pi}{4}\right]$ rad. At $t = 0.5$ s, at $t = 3.5$ s, and at $t = 6.5$ s, the robot is asked to go to the reference configurations $\boldsymbol{q}_r(2\,\text{s} < t \le 3.5\,\text{s}) = [0.72, 0.21, 0.05, -2.47, -0.07, 2.69, 0.75]$ rad, $\boldsymbol{q}_r(3.5\,\text{s} < t \le 6.5\,\text{s}) = [0.18, 0.24, -0.85, -1.64, 0.14, 1.81, 0.85]$ rad, and back to the initial configuration $\boldsymbol{q}_r(t > 6.5\,\text{s}) = \boldsymbol{q}(t \le 0.5\,\text{s})$.

As depicted in Fig. 4, the desired configurations $\boldsymbol{q}_r$ are always reached in a stable and safe (i.e. $\Delta \ge 0$) manner. We can see that at the moment a new reference step is given, the future torques, joint velocities, or Cartesian end-effector velocities are closest to violation. The DSM values of the latter constraints slow down the rate of change of the applied reference, such that a feasible reference $\boldsymbol{q}_v$ is applied to the robot. Fig. 5 shows the resulting end-effector positions giving the joint space step references $\boldsymbol{q}_r$.

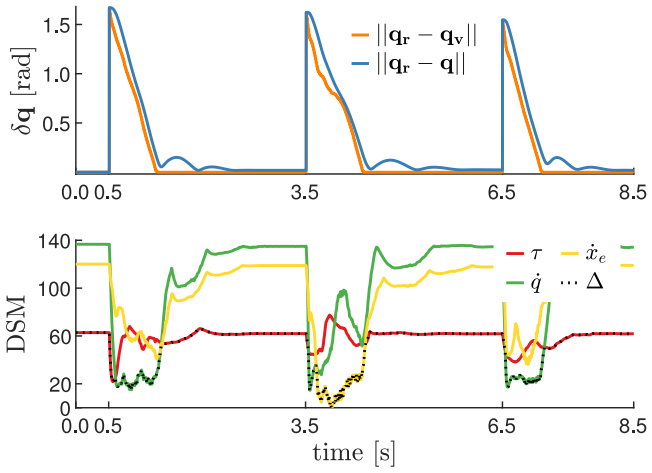


Fig. 4. Obstacle-Free Environment (Exp 1) – The applied reference $\boldsymbol{q}_v$ and the robot configuration $\boldsymbol{q}$ reach different reference configurations $\boldsymbol{q}_r$ (i.e. $\|\boldsymbol{q}_r - \boldsymbol{q}_v\| \to 0$ and $\|\boldsymbol{q}_r - \boldsymbol{q}\| \to 0$) in a stable and safe manner (i.e. DSM ≥ 0).

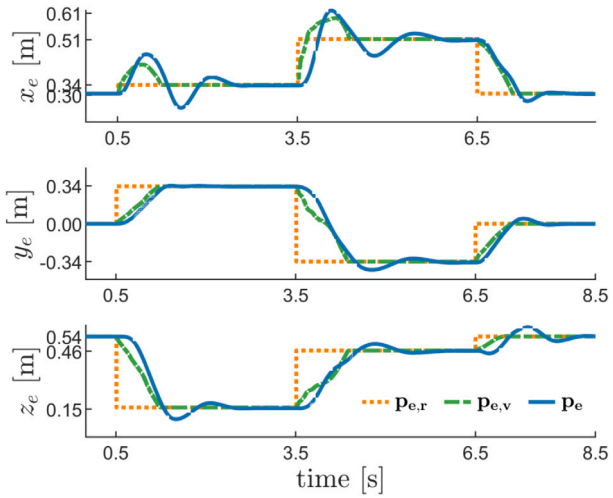


Fig. 5. Obstacle-Free Environment (Exp 1) – The given joint space step references $\boldsymbol{q}_r$ result in position step references $\boldsymbol{p}_{e,r}$ in Cartesian space for the end-effector, which the applied reference and the robot follows, i.e. $\boldsymbol{p}_{e,v}$ and $\boldsymbol{p}_e$ respectively.

5.4. Static obstacle environment

In this section we study the behavior of the robot when it has to avoid the wall constraint $\boldsymbol{c}^{\mathrm{w}} \cdot \boldsymbol{p}_i \leq d^{\mathrm{w}}$ with $\boldsymbol{c}^{\mathrm{w}} = [\cos\theta, \sin\theta, 0]^T$, $d^{\mathrm{w}} = 0.55\,\mathrm{m}$, and where $\theta = 310°$ represents the rotation about the z-axis with respect to the robot base frame. For the following experiments, we use the ERG parameters as defined in Section 5.3 and the wall constraint ERG parameters which will be specified per experiment.

5.4.1. Influence of the wall constraint DSM

To observe the importance of the DSM in the ERG, we compare the robot behavior without and with the wall constraint DSM, i.e. Δ^{w}. The task is for both experiments the same as in Section 5.3: the robot starts in the same initial configuration and receives the same reference configurations at $t = 0.5\,\mathrm{s}$ and $t = 3.5\,\mathrm{s}$. For the collision avoidance with the wall constraint, we use $\delta^{\mathrm{w}} = 0.01$, $\zeta^{\mathrm{w}} = 0.25$, and $\kappa^{\mathrm{w}} = 100$. To investigate the effect of the Δ^{w}, we neglect the Cartesian end-effector velocity constraint, i.e. $\Delta^{\dot{x}_e}$, since it is an auxiliary constraint that will not be violated in both experiments, but that can slow down the robot too much around $t = 3.5\,\mathrm{s}$.

In the experiment without Δ^{w}, we can see in Fig. 6 that the predicted joint velocity $\dot{\boldsymbol{q}}$ violated around $t = 3.6\,\mathrm{s}$ its constraint. This smaller overall Δ slows down the applied reference and the robot such that the real $\dot{\boldsymbol{q}}$ does not violate its constraint during this experiment, as can be seen in Fig. 7. However, in this figure we can see that the wall constraint is violated around $t = 4.3\,\mathrm{s}$ with almost 2 cm. This means that the repulsion field alone is not able to enforce constraints without a DSM because the applied reference rate is too high to change the direction and avoid collisions.

Whereas when the wall constraint is included in the computation of the overall DSM, the applied reference $\boldsymbol{q}_v$ changes slower since the overall Δ is smaller, especially around $t = 3.5\,\mathrm{s}$. And although the predicted wall constraints are almost violated, as can be seen in the DSM graph of Fig. 8, the real wall constraints are not, as is depicted in Fig. 9.

For both experiments, the reference given at $t = 3.5\,\mathrm{s}$ would cause the robot to move its end-effector behind the wall and is therefore a steady-state *inadmissible* reference. This is the reason why, in both cases, the error $\|\boldsymbol{q}_r - \boldsymbol{q}_v\|$ does not converge to zero at $t = 6.5\,\mathrm{s}$, but converges to a steady-state admissible configuration. This shows that the navigation field is designed to handle steady-state inadmissible references.

5.4.2. Robot follows pose of reference object while avoiding human's safe workspace behind virtual wall

In this experiment we show how the trajectory-based ERG could be implemented in a scenario where a human works near a robot with a virtual wall separating the human's and the robot's workspaces, as visualized in Fig. 10. Here we use for the collision avoidance with the virtual wall constraint the following ERG parameters: $\delta^{\mathrm{w}} = 0.01$, $\zeta^{\mathrm{w}} = 0.10$, and $\kappa^{\mathrm{w}} = 150$. Five reflective markers are attached to the reference object to let the Vicon motion capture system detect its pose. To avoid marker occlusions as much as possible and prevent the robot from touching the wooden plate, an extra upwards translation is added, such that the end-effector pose reference is always 25 cm displaced orthogonal to the center of the wooden reference plate.

When the human gives a reference relatively far away from the wall, the applied reference can follow the reference given by the human. When the human moves the wooden reference plate faster or when the human gives a reference close to or behind the wall, the overall DSM decreases and as such the applied reference is unable to change as fast as the reference. This experiment shows that the robot has a moderately aggressive behavior far away from the virtual wall, but moves slower when close to the wall. Since the robot never penetrates the wall, the human can work safely in its presence.

5.5. Dynamic obstacle environment

In this section we demonstrate how the trajectory-based ERG could be used in a scenario where the robot and the human share a workspace in a safe way. Therefore, we analyze the behavior of the robot in case it has to avoid the head, upper arm, and lower arm of a human coworker. The human head is seen by the robot as a spherical obstacle with a radius $r^{\mathrm{s}}_{\mathrm{head}} = 10\,\mathrm{cm}$, the center $\boldsymbol{c}^{\mathrm{s}}_{\mathrm{head}}$ is defined by the reflective markers attached to the baseball cap. The human upper and lower arm are seen as cylindrical obstacles with both a radius $r^{\mathrm{c}}_{\mathrm{upperarm}} = r^{\mathrm{c}}_{\mathrm{lowerarm}} = 5\,\mathrm{cm}$. The begin and end points of the cylinder that represents the upper arm are given by the reflective markers attached to the elbow and shoulder, respectively. The end point of the cylinder that represents the lower arm is also given by the marker attached to the elbow. The begin point of the lower arm should represent the position of the wrist, but is attached in between the wrist and the elbow to make a shorter link in Vicon such that the Vicon system would easily see the difference between the upper and lower arm. To make it correct, an extra translation towards the actual wrist location is added programmatically.

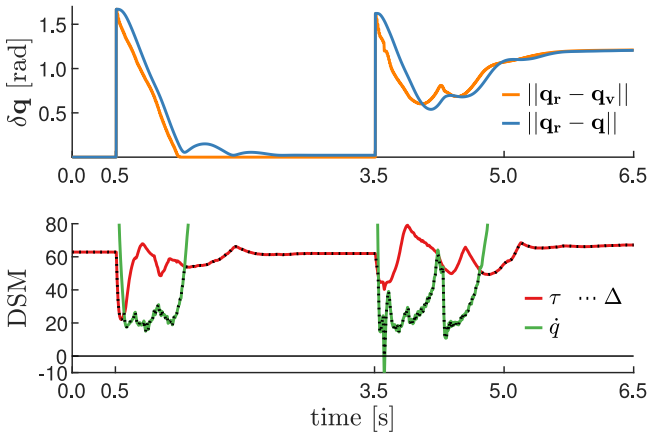


Fig. 6. Static Obstacle Environment Without Wall Constraint DSM (Exp 2) – The robot reaches the steady-state reference given at $t = 0.5$ s and approaches the steady-state inadmissible reference given at $t = 3.5$ s. When the step reference at $t = 3.5$ s is given, a joint velocity constraint violation is detected in the future, i.e. $\Delta^{\dot{q}} < 0$, such that $\Delta = 0$ at 3.6 s.

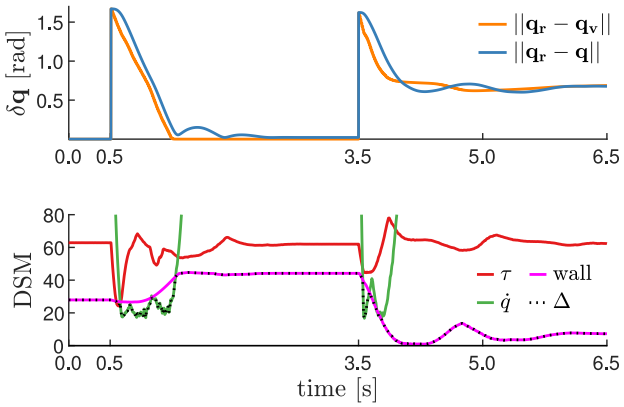


Fig. 8. Static Obstacle Environment With Wall Constraint DSM (Exp 3) – The robot reaches the steady-state reference given at $t = 0.5$ s and approaches the steady-state inadmissible reference given at $t = 3.5$ s in a stable and safe way (i.e. $\Delta > 0$). Since the overall Δ is smaller, the applied reference changes slower than in Fig. 6.

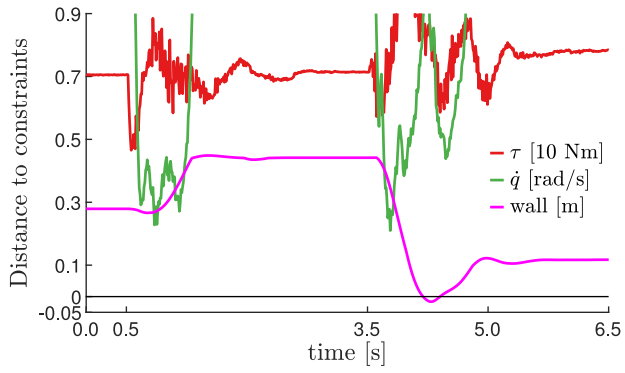


Fig. 7. Static Obstacle Environment without Wall Constraint DSM (Exp 2) – Although $\Delta^{\dot{q}} < 0$ around 3.6 s in Fig. 6, the distance of the real (not predicted) $\dot{q}$ to its constraints is not violated. However, since Δ^{w} is not taken into account, the wall constraint is violated around $t = 4.3$ s by almost 2 cm.

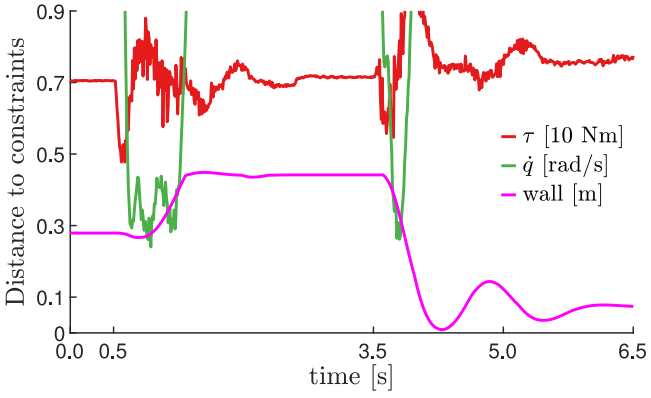


Fig. 9. Static Obstacle Environment With Wall Constraint DSM (Exp 3) – In this experiment Δ^{w} is taken into account to compute the overall Δ. Thus, the applied reference is slowed down enough to avoid any constraint violations.

For the following experiments, we use the ERG parameters as defined in Section 5.3, the spherical constraint ERG parameters $\delta^{s} = 0.01$, $\zeta^{s} = 0.40$, and $\kappa^{s} = 150$, and the cylindrical constraint ERG parameters $\delta^{c} = 0.01$, $\zeta^{c} = 0.40$, and $\kappa^{c} = 150$.

5.5.1. Robot stays in initial configuration while avoiding the human's arm and head

As shown in Fig. 11, when the human arm and head approach the robot, the robot moves away and keeps a safe distance from the human. Whereas the moment the human arm and head are far enough, the robot moves quickly back to its initial configuration. This is also depicted in Fig. 12 at time $t = \{0.5, 4.5, 9.2, 13.0, 15.9, 19.5, 23.0\}$ s. Around $t = 21$ s and $t = 28$ s, the human moves faster than the robot, whereby the robot cannot avoid the human, resulting in human arm constraint violations. Thereby, the overall DSM becomes zero, i.e. $\Delta = 0$, such that the applied reference does not change for a moment and the robot comes to a standstill. This is the worst case scenario, but the human will not get hurt by the robot since the robot's speed is zero at the moment the human touches the robot.

5.5.2. Robot follows reference object pose while avoiding the human's arm and head

For this experiment, the same wooden reference plate is used as in Section 5.4.2, as can be seen in Fig. 1. As shown in Fig. 13 at $t = 1.1$ s and at $t = 11.4$ s, when the human is far enough away from the robot, the robot can increase its speed and can reach the reference pose. Whereas when the reference pose is closer to the human arm at $t = 6.2$s and at $t = 17.3$ s or to the human head at $t = 8.9$ s, the robot moves away from the human and goes to a steady-state admissible configuration. Around $t = 25.2$ s the human arm moved faster to the robot than the robot could move away, and the robot comes to a standstill for about 1.5 s.

6. Discussion

In Section 5, we presented an extensive set of experimental studies of the proposed trajectory-based ERG framework, including highly dynamic human–robot coexistence experiments. These studies demonstrate the following key results when applied to the Panda collaborative robotic manipulator.

- R1: The method is computationally efficient and allows high-rate real-time (1 kHz) computation of the control commands.
- R2: The applied reference is updated at a rate of 100 Hz. This rate is imposed by the perception layer, which relies on the Vicon system. The trajectory-based ERG has an average computational time of 1.07 ms for the most computationally demanding experiment (Exp 6), as can be seen in Fig. 14.

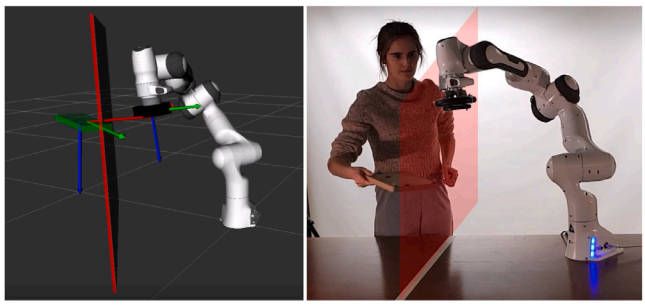


Fig. 10. Static Obstacle Environment with Virtual Wall Between Human and Robot (Exp 4) – The robot follows the wooden reference plate with five reflective markers detected by the Vicon motion capture system while avoiding the virtual wall. The end-effector pose reference, depicted as the green plate in the RVIZ visualization figure, is always 25 cm displaced orthogonal to the center of the wooden reference plate in the upwards direction to avoid marker occlusions. The robot has a moderately aggressive behavior far away from the virtual wall, moves slower close to the wall, and never penetrates the wall but converges to a steady-state admissible approximation of the reference. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

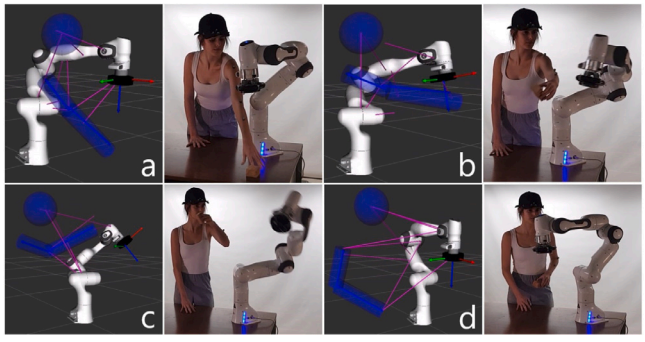


Fig. 11. Dynamic Obstacle Environment with Robot in Initial Configuration (Exp 5) – The robot tries to stay in its initial configuration. When the human arm and head (blue cylinders and sphere in RVIZ figure) approach the robot, the robot moves away from the human arm and moves quickly back to its initial configuration once the human arm is far enough away. The distance between the human arm and head, and the applied robot configuration $\boldsymbol{q}_v$ is depicted with the magenta lines in the RVIZ figure. This distance and direction is used to compute the repulsion field of the moving spherical (i.e. head) and cylindrical (i.e. upper and lower arm) obstacles.. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

- R3: To exploit multi-core processor capabilities, the NF and DSM run in parallel. Due to the trajectory predictions, the time required to compute the DSM is typically an order of magnitude larger than the time needed to compute the NF. Therefore, the computational time of the ERG is equal to the time required to compute the DSM plus the time required to compute the update of the applied reference $\boldsymbol{q}_v$.
- R4: The method converges to either the target reference or a steady-state admissible approximation thereof.
- R5: The trajectory-based ERG provides provably safety under actuator inputs and state constraints, including collision avoidance towards static and dynamic obstacles. Note that by only implementing the navigation field, the robot can only move safe when it is moving slowly. The moment the robot needs to move fast, it has to be able to quickly accelerate and decelerate, resulting in a robot that cannot stop immediately if necessary. The dynamic safety margin avoids constraint violations that might otherwise happen as a result of transient dynamics.

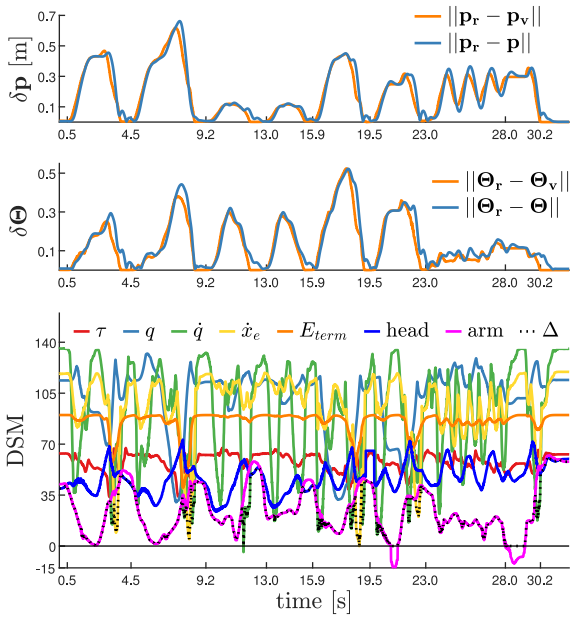


Fig. 12. Dynamic Obstacle Environment with Robot in Initial Configuration (Exp 5) – At time $t = \{0.5, 4.5, 9.2, 13.0, 15.9, 19.5, 23.0\}$ s the robot moves away from the approaching human arm (i.e. $\Delta^{\text{arm}} \rightarrow 0$) and moves quickly back to its initial configuration once the human arm is far enough (i.e. increasing Δ^{arm}). Around $t = 21$ s and $t = 28$ s, the human moves faster than the robot, whereby the robot cannot avoid the human, resulting in a safe collision, since $\Delta = 0$, as such $\dot{\boldsymbol{q}}_v = 0$, which means that the robot will be hit by the human when it is at standstill.

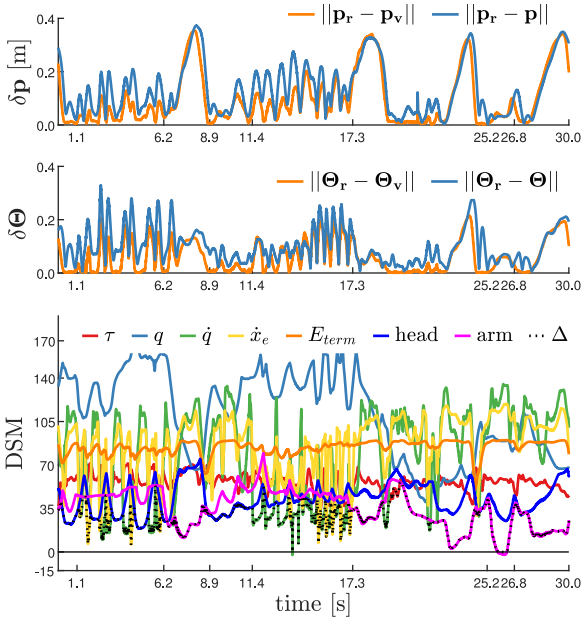


Fig. 13. Dynamic Obstacle Environment with Robot Following Reference Object Pose (Exp 6) – The robot can follow the reference object pose, but cannot track it perfectly the moment the human arm (at $t = 6.2$ s and $t = 17.3$ s) or head (at $t = 8.9$ s) is nearby and as such goes to a steady-state admissible configuration. In the worst case scenario, when the human arm moves faster than the robot arm, the robot comes to a standstill resulting in a constraint violation (around $t = 25.2$ s) which is safe since the robot energy is zero at the moment of collision.

7. Conclusion

In this article, we formulated the trajectory-based ERG algorithm for a robotic manipulator in the presence of actuator saturations, limited

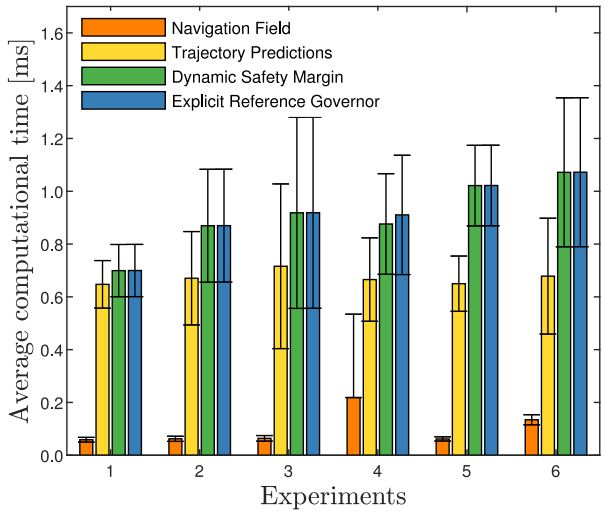


Fig. 14. Average Computational Time of Trajectory-Based ERG with Standard Deviation for Exp 1 to Exp 6 – The NF and DSM run in parallel on a multi-core processor and because the time required to compute the DSM is mostly an order larger than the time necessary to compute the NF, the computational time of the ERG is equal to the time required to compute the DSM plus the time required to compute the update of the applied reference $\boldsymbol{q}_v$. Since the time needed to compute the trajectory predictions are included in the DSM computational time, we can see that they have the highest impact on the overall computational time of the DSM, and so of the ERG. The computational time of the ERG is in average 1.07 ms for the most computational demanding experiment.

joint ranges, velocity constraints, and obstacles. We showed that the proposed computationally efficient constrained control framework is able to steer the robot arm to the desired end-effector pose (or an admissible approximation) while avoiding constraint violations. Moreover, we provided validation cases in which a human and the robot could safely execute their tasks while sharing their workspace in close proximity with each other.

We believe this work shows that collaborative robotic manipulators can be safe towards their environment while also moving fast, thus increasing the speed and so the production efficiency in human–robot co-shared workspaces.

CRediT authorship contribution statement

Kelly Merckaert: Conceptualization, Methodology, Software, Validation, Investigation, Data curation, Writing – original draft, Writing – review & editing, Visualization, Project administration. **Bryan Convens:** Software, Validation, Investigation, Data curation, Writing – review & editing, Visualization. **Chi-ju Wu:** Software. **Alessandro Roncone:** Resources, Writing – review & editing. **Marco M. Nicotra:** Conceptualization, Methodology, Writing – review & editing, Visualization. **Bram Vanderborght:** Resources, Writing – review & editing, 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.

Acknowledgments

This work was supported by Fonds Wetenschappelijk Onderzoek (FWO), Belgium under grant numbers 37472, 60523, and 62062, by the EU H2020 project under grant number 871237, and by the Flemish Government under the program "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen".

Appendix A. Robot parametrization for spherical obstacles

Referring to Fig. 15, we note that the point $\boldsymbol{p}_{ij}^{\mathrm{s}}$ can be computed using

$$\boldsymbol{p}_{ij}^{\mathrm{s}} = \boldsymbol{p}_i + \lambda_{ij}^{\mathrm{s}} \left(\boldsymbol{p}_{i+1} - \boldsymbol{p}_i\right), \tag{44}$$

where $\lambda_{ij}^{\mathrm{s}} \in [0,1]$ is a parametrization factor. If we project $\boldsymbol{c}_j^{\mathrm{s}}$ onto link i, i.e. $\overline{\boldsymbol{p}_i\boldsymbol{p}_{i+1}}$, we obtain the point $\boldsymbol{p}_{ij}^{\mathrm{s}}$, which means that $\overline{\boldsymbol{p}_i\boldsymbol{p}_{i+1}} \perp \overline{\boldsymbol{p}_{ij}^{\mathrm{s}}\boldsymbol{c}_j^{\mathrm{s}}}$,

$$\left(\boldsymbol{p}_{i+1} - \boldsymbol{p}_i\right)^T \left(\boldsymbol{c}_j^{\mathrm{s}} - \boldsymbol{p}_{ij}^{\mathrm{s}}\right) = 0, \tag{45}$$

and that $\boldsymbol{p}_{ij}^{\mathrm{s}}$ belongs to $\overline{\boldsymbol{p}_i\boldsymbol{p}_{i+1}}$. Combining (44) and (45) gives us the parametrization factor $\lambda_{ij}^{\mathrm{s}} \in [0,1]$,

$$\lambda_{ij}^{\mathrm{s}} = \frac{\left(\boldsymbol{p}_{v,i+1} - \boldsymbol{p}_{v,i}\right)^T \left(\boldsymbol{c}_j^{\mathrm{s}} - \boldsymbol{p}_{v,i}\right)}{\left(\boldsymbol{p}_{v,i+1} - \boldsymbol{p}_{v,i}\right)^T \left(\boldsymbol{p}_{v,i+1} - \boldsymbol{p}_{v,i}\right)}, \tag{46}$$

where

$$\lambda_{ij}^{\mathrm{s}} = 0 \qquad \text{if } \lambda_{ij}^{\mathrm{s}} < 0, \tag{47a}$$

$$\lambda_{ij}^{\mathrm{s}} = 1 \qquad \text{if } \lambda_{ij}^{\mathrm{s}} > 1. \tag{47b}$$

Appendix B. Robot parametrization for cylindrical obstacles

Note: In the NF section, all joint angles $\boldsymbol{q}$ represent the applied joint angles, i.e. $\boldsymbol{q}_v$, and all the joint positions $\boldsymbol{p}_i$ represent the joint positions of the applied reference configuration, i.e. $\boldsymbol{p}_{v,i}$. Whereas in the DSM section, all joint angles $\boldsymbol{q}$ represent the predicted joint angles, i.e. $\hat{\boldsymbol{q}}_{t'}$, and all the joint positions $\boldsymbol{p}_i$ represent the joint positions of the predicted reference configuration, i.e. $\hat{\boldsymbol{p}}_{t',i}$.

To compute $\boldsymbol{p}_{ij}^{\mathrm{c}}$ and $\boldsymbol{t}_{ij}^{\mathrm{c}}$, we need to analyze two different cases: the case where the robot link and the cylinder are parallel to each other and the case where they are skew. In the parallel case, the parametrization of the robot link or the cylinder will be simplified to the point-line case, as illustrated in Fig. 16 and detailed in Algorithm 1.

For the skew case, both the robot link and the cylinder need to be parametrized. Similar to (45), the shortest distance $\|\overline{\boldsymbol{p}_{ij}^{\mathrm{c}}\boldsymbol{t}_{ij}^{\mathrm{c}}}\|$ should be perpendicular to link i and cylinder j,

$$\left(\boldsymbol{p}_{i+1} - \boldsymbol{p}_i\right)^T \left(\boldsymbol{t}_{ij}^{\mathrm{c}} - \boldsymbol{p}_{ij}^{\mathrm{c}}\right) = 0, \tag{48a}$$

$$\left(\boldsymbol{p}_j^{\mathrm{c}_1} - \boldsymbol{p}_j^{\mathrm{c}_0}\right)^T \left(\boldsymbol{t}_{ij}^{\mathrm{c}} - \boldsymbol{p}_{ij}^{\mathrm{c}}\right) = 0, \tag{48b}$$

with $\boldsymbol{p}_j^{\mathrm{c}_0}$ and $\boldsymbol{p}_j^{\mathrm{c}_1}$ the start and end point of cylinder j. As in (44), $\boldsymbol{p}_{ij}^{\mathrm{c}}$ belongs to $\overline{\boldsymbol{p}_i\boldsymbol{p}_{i+1}}$ and $\boldsymbol{t}_{ij}^{\mathrm{c}}$ belongs to $\overline{\boldsymbol{p}_j^{\mathrm{c}_0}\boldsymbol{p}_j^{\mathrm{c}_1}}$,

$$\boldsymbol{p}_{ij}^{\mathrm{c}} = \boldsymbol{p}_i + \lambda_{ij}^{\mathrm{c}} \left(\boldsymbol{p}_{i+1} - \boldsymbol{p}_i\right), \tag{49a}$$

$$\boldsymbol{t}_{ij}^{\mathrm{c}} = \boldsymbol{p}_j^{\mathrm{c}_0} + \mu_{ij}^{\mathrm{c}} \left(\boldsymbol{p}_j^{\mathrm{c}_1} - \boldsymbol{p}_j^{\mathrm{c}_0}\right), \tag{49b}$$

with $\lambda_{ij}^{\mathrm{c}} \in [0,1]$ and $\mu_{ij}^{\mathrm{c}} \in [0,1]$ the parametrization factors for link i and cylinder j, respectively. Combining (48) and (49) gives,

$$\lambda_{ij}^{\mathrm{c}} = -\frac{\boldsymbol{c}_0^T \boldsymbol{b}\, \boldsymbol{b}^T \boldsymbol{a} + \boldsymbol{b}^T \boldsymbol{b}\, \boldsymbol{c}_0^T \boldsymbol{a}}{\boldsymbol{b}^T \boldsymbol{b}\, \boldsymbol{a}^T \boldsymbol{a} - \boldsymbol{a}^T \boldsymbol{b}\, \boldsymbol{b}^T \boldsymbol{a}}, \tag{50}$$

$$\mu_{ij}^{\mathrm{c}} = \frac{\boldsymbol{a}^T \boldsymbol{a}}{\boldsymbol{b}^T \boldsymbol{b}} \frac{\boldsymbol{b}^T \boldsymbol{b}\, \boldsymbol{c}_0^T \boldsymbol{a} - \boldsymbol{c}_0^T \boldsymbol{b}\, \boldsymbol{b}^T \boldsymbol{a}}{\boldsymbol{b}^T \boldsymbol{b}\, \boldsymbol{a}^T \boldsymbol{a} - \boldsymbol{b}^T \boldsymbol{a}\, \boldsymbol{b}^T \boldsymbol{a}} - \frac{\boldsymbol{c}_0^T \boldsymbol{a}}{\boldsymbol{b}^T \boldsymbol{a}}, \tag{51}$$

where

$$\lambda_{ij}^{\mathrm{c}} = 0 \ \text{ if } \lambda_{ij}^{\mathrm{c}} < 0, \quad \lambda_{ij}^{\mathrm{c}} = 1 \ \text{ if } \lambda_{ij}^{\mathrm{c}} > 1, \tag{52a}$$

$$\mu_{ij}^{\mathrm{c}} = 0 \ \text{ if } \mu_{ij}^{\mathrm{c}} < 0, \quad \mu_{ij}^{\mathrm{c}} = 1 \ \text{ if } \mu_{ij}^{\mathrm{c}} > 1 \tag{52b}$$

with $\boldsymbol{a} = \overline{\boldsymbol{p}_i\boldsymbol{p}_{i+1}}$, $\boldsymbol{b} = \overline{\boldsymbol{p}_j^{\mathrm{c}_0}\boldsymbol{p}_j^{\mathrm{c}_1}}$, $\boldsymbol{c}_0 = \overline{\boldsymbol{p}_i\boldsymbol{p}_j^{\mathrm{c}_0}}$, and $\boldsymbol{c}_1 = \overline{\boldsymbol{p}_i\boldsymbol{p}_j^{\mathrm{c}_1}}$. This is illustrated in Fig. 17 and detailed in Algorithm 2.

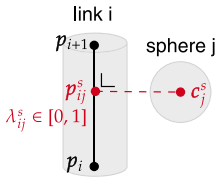
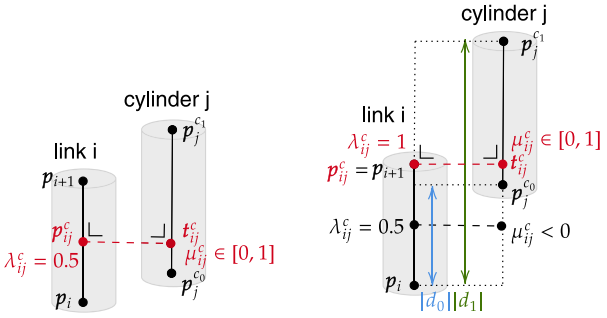


Fig. 15. Parametrization of Robot Link i with respect to Sphere j – The distance $\|\overline{p_{ij}^s c_j^s}\|$ defines the shortest distance between link i and sphere j. The point p_{ij}^s can be found by orthogonally projecting c_j^s onto $\overline{p_i p_{i+1}}$.



(a) $\mu_{ij}^c \in [0,1]$ for $\lambda_{ij}^c = 0.5$ (b) $\mu_{ij}^c < 0$ for $\lambda_{ij}^c = 0.5$

Fig. 16. Parametrization of Robot Link i and Cylinder j – The distance $\|\overline{p_{ij}^c t_{ij}^c}\|$ defines the shortest distance between link i and cylinder j in the case they are parallel to each other.

Algorithm 1: Code to obtain the points p_{ij}^c and t_{ij}^c for the case that robot link i and cylinder j are parallel to each other.

Input : p_i, p_{i+1}, $p_j^{c_0}$, and $p_j^{c_1}$
Output: λ_{ij}^c, p_{ij}^c, and t_{ij}^c

Project cylinder j on robot segment i: **begin**
  $d_0 = \frac{a}{\|a\|} \cdot c_0$;
  $d_1 = \frac{a}{\|a\|} \cdot c_1$;
Find λ_{ij}^c and μ_{ij}^c: **begin**
  $\mu_{ij}^c = \frac{b^T\left(\frac{p_i+p_{i+1}}{2} - p_j^{c_0}\right)}{b^T b}$ ← parametrization factor for closest point on cylinder j w.r.t. center of robot link i;
  if $0 \le \mu_{ij}^c \le 1$ **then**
    $\lambda_{ij}^c = 0.5$ ← center of robot segment i is good estimation
  else if $\left(\mu_{ij}^c < 0 \wedge d_1 \ge \|a\|\right) \vee \left(\mu_{ij}^c > 1 \wedge d_0 \ge \|a\|\right)$ **then**
    $\lambda_{ij}^c = 1$;
    $\mu_{ij}^c = \frac{b^T\left(p_{i+1} - p_j^{c_0}\right)}{b^T b}$
  else if $\left(\mu_{ij}^c < 0 \wedge d_1 \le 0\right) \vee \left(\mu_{ij}^c > 1 \wedge d_0 \le 0\right)$ **then**
    $\lambda_{ij}^c = 0$;
    $\mu_{ij}^c = \frac{b^T\left(p_i - p_j^{c_0}\right)}{b^T b}$
  else if $\left(\mu_{ij}^c < 0 \wedge 0 < d_1 < \|a\|\right) \vee \left(\mu_{ij}^c > 1 \wedge 0 < d_0 < \|a\|\right)$ **then**
    $\mu_{ij}^c = 0.5$;
    $\lambda_{ij}^c = \frac{a^T\left(\frac{p_j^{c_0} + p_j^{c_1}}{2} - p_i\right)}{a^T a}$
Find p_{ij}^c and t_{ij}^c: **begin**
  $p_{ij}^c = p_i + \lambda_{ij}^c\left(p_{i+1} - p_i\right)$ ← closest point on robot link i to cylinder j;
  $t_{ij}^c = p_j^{c_0} + \mu_{ij}^c\left(p_j^{c_1} - p_j^{c_0}\right)$ ← closest point on cylinder j to robot link i;

Algorithm 2: Code to obtain the points p_{ij}^c and t_{ij}^c for the case that robot link i and cylinder j are skew to each other.

Input : p_i, p_{i+1}, $p_j^{c_0}$, and $p_j^{c_1}$,
initial λ_{ij}^c and μ_{ij}^c from (50)
Output: λ_{ij}^c, p_{ij}^c, and t_{ij}^c

Find λ_{ij}^c, μ_{ij}^c, p_{ij}^c, and t_{ij}^c: **begin**
  if $\lambda_{ij}^c \in [0,1]$ *and* $\mu_{ij}^c \in [0,1]$ **then**
    $p_{ij}^c = p_i + \lambda_{ij}^c\left(p_{i+1} - p_i\right)$;
    $t_{ij}^c = p_j^{c_0} + \mu_{ij}^c\left(p_j^{c_1} - p_j^{c_0}\right)$;
  else if $\lambda_{ij}^c \notin [0,1]$ *and* $\mu_{ij}^c \in [0,1]$ **then**
    if $\lambda_{ij}^c < 0$ **then**
      $\lambda_{ij}^c = 0$;
    else if $\lambda_{ij}^c > 1$ **then**
      $\lambda_{ij}^c = 1$;
    $p_{ij}^c = p_i + \lambda_{ij}^c\left(p_{i+1} - p_i\right)$;
    $\mu_{ij}^c = \frac{b^T\left(p_{ij}^c - p_j^{c_0}\right)}{b^T b}$;
    if $\mu_{ij}^c < 0$ **then**
      $\mu_{ij}^c = 0$;
    else if $\mu_{ij}^c > 1$ **then**
      $\mu_{ij}^c = 1$;
    $t_{ij}^c = p_j^{c_0} + \mu_{ij}^c\left(p_j^{c_1} - p_j^{c_0}\right)$;
  else if $\lambda_{ij}^c \in [0,1]$ *and* $\mu_{ij}^c \notin [0,1]$ **then**
    if $\mu_{ij}^c < 0$ **then**
      $\mu_{ij}^c = 0$;
    else if $\mu_{ij}^c > 1$ **then**
      $\mu_{ij}^c = 1$;
    $t_{ij}^c = p_j^{c_0} + \mu_{ij}^c\left(p_j^{c_1} - p_j^{c_0}\right)$;
    $\lambda_{ij}^c = \frac{a^T\left(t_{ij}^c - p_i\right)}{a^T a}$;
    if $\lambda_{ij}^c < 0$ **then**
      $\lambda_{ij}^c = 0$;
    else if $\lambda_{ij}^c > 1$ **then**
      $\lambda_{ij}^c = 1$;
    $p_{ij}^c = p_i + \lambda_{ij}^c\left(p_{i+1} - p_i\right)$;
  else if $\lambda_{ij}^c \notin [0,1]$ *and* $\mu_{ij}^c \notin [0,1]$ **then**
    $\mu_{ij\,|\lambda_{ij}^c=0} = \frac{b^T(p_i - p_j^{c_0})}{b^T b}$;
    $\mu_{ij\,|\lambda_{ij}^c=1} = \frac{b^T(p_{i+1} - p_j^{c_0})}{b^T b}$;
    if $\mu_{ij\,|\lambda_{ij}^c=0} < 0$ **then**
      $\mu_{ij\,|\lambda_{ij}^c=0} = 0$;
    else if $\mu_{ij\,|\lambda_{ij}^c=0} > 1$ **then**
      $\mu_{ij\,|\lambda_{ij}^c=0} = 1$;
    if $\mu_{ij\,|\lambda_{ij}^c=1} < 0$ **then**
      $\mu_{ij\,|\lambda_{ij}^c=1} = 0$;
    else if $\mu_{ij\,|\lambda_{ij}^c=1} > 1$ **then**
      $\mu_{ij\,|\lambda_{ij}^c=1} = 1$;
    $t_{ij\,|\lambda_{ij}^c=0}^c = p_j^{c_0} + \mu_{ij\,|\lambda_{ij}^c=0}\left(p_j^{c_1} - p_j^{c_0}\right)$;
    $t_{ij\,|\lambda_{ij}^c=1}^c = p_j^{c_0} + \mu_{ij\,|\lambda_{ij}^c=1}\left(p_j^{c_1} - p_j^{c_0}\right)$;
    if $\|p_i t_{ij\,|\lambda_{ij}^c=0}^c\| < \|p_{i+1} t_{ij\,|\lambda_{ij}^c=1}^c\|$ **then**
      $p_{ij}^c = p_i$;
      $t_{ij}^c = t_{ij\,|\lambda_{ij}^c=0}^c$;
    else
      $p_{ij}^c = p_{i+1}$;
      $t_{ij}^c = t_{ij\,|\lambda_{ij}^c=1}^c$;

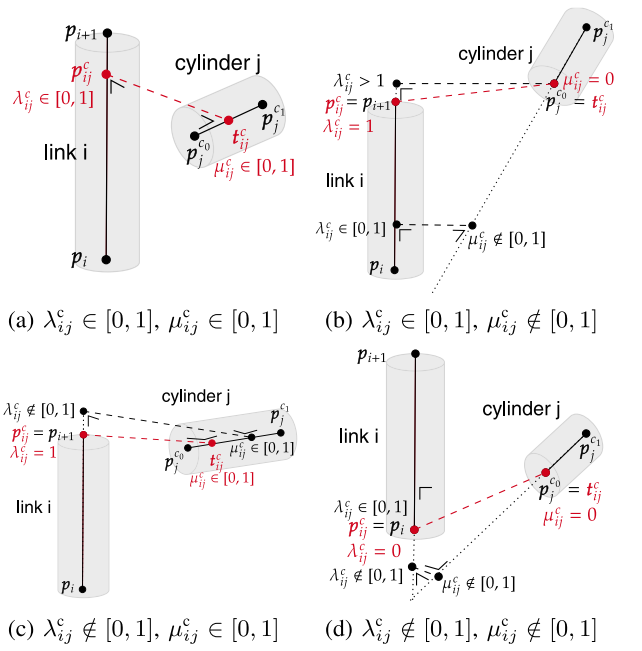


Fig. 17. Parametrization of Robot Link i and Cylinder j – The distance $\|\overline{p_{ij}^c t_{ij}^c}\|$ defines the shortest distance between link i and cylinder j in the case they are skew to each other.

References

[1] F. Ferraguti, A. Pertosa, C. Secchi, C. Fantuzzi, M. Bonfè, A methodology for comparative analysis of collaborative robots for industry 4.0, in: 2019 Design Automation & Test in Europe Conference & Exhibition, DATE, Florence, Italy, 2019, pp. 1070–1075, http://dx.doi.org/10.23919/DATE.2019.8714830.

[2] B. Vanderborght, Unlocking the Potential of Industrial Human–Robot Collaboration, European Commission, 2019, http://dx.doi.org/10.2777/568116.

[3] A. Ajoudani, P. Albrecht, M. Bianchi, A. Cherubini, S. Del Ferraro, P. Fraisse, L. Fritzsche, M. Garabini, A. Ranavolo, P.H. Rosen, M. Sartori, N. Tsagarakis, B. Vanderborght, S. Wischniewski, Smart collaborative systems for enabling flexible and ergonomic work practices, IEEE Robot. Autom. Mag. 27 (2) (2020) 169–176, http://dx.doi.org/10.1109/MRA.2020.2985344.

[4] A.M. Zanchettin, P. Rocco, S. Chiappa, R. Rossi, Towards and optimal avoidance strategy for collaborative robots, Robot. Comput.-Integr. Manuf. 59 (2019) 47–55, http://dx.doi.org/10.1016/j.rcim.2019.01.015.

[5] Robotic Industries Association (RIA), Four types of collaborative robot operation, https://www.controleng.com/articles/four-types-of-collaborative-robot-operation/.

[6] L.-P. Ellekilde, H. Gordon Petersen, Motion planning efficient trajectories for industrial bin-picking, Int. J. Robot. Res. 32 (9–10) (2013) 991–1004, http://dx.doi.org/10.1177/0278364913487237.

[7] B. Vanderborght, A. Albu-Schäffer, A. Bicchi, E. Burdet, D.G. Caldwell, R. Carloni, M. Catalano, O. Eiberger, W. Friedl, G. Ganesh, M. Garabini, M. Grebenstein, G. Grioli, S. Haddadin, H. Hoppner, A. Jafari, M. Laffranchi, D. Lefeber, F. Petit, S. Stramigioli, N. Tsagarakis, M. Van Damme, R. Van Ham, L.C. Visser, S. Wolf, Variable impedance actuators: A review, Robot. Auton. Syst. 61 (12) (2013) 1601–1614, http://dx.doi.org/10.1016/j.robot.2013.06.009.

[8] A. Albu-Schäffer, S. Haddadin, C. Ott, A. Stemmer, T. Wimböck, G. Hirzinger, The DLR lightweight robot: design and control concepts for robots in human environments, Ind. Robot 34 (5) (2007) 376–385, http://dx.doi.org/10.1108/01439910710774386.

[9] N. Kashiri, J. Malzahn, N.G. Tsagarakis, On the sensor design of torque controlled actuators: A comparison study of strain gauge and encoder-based principles, IEEE Robot. Autom. Lett. 2 (2) (2017) 1186–1194, http://dx.doi.org/10.1109/LRA.2017.2662744.

[10] Blue Danube Robotics GmbH, AIRSKIN comprehensive safety technology for robots and grippers, https://www.bluedanuberobotics.com/airskin/.

[11] J. Gielniak, C.K. Liu, A.L. Thomaz, Generating human-like motion for robots, Int. J. Robot. Res. 32 (11) (2013) 1275–1301, http://dx.doi.org/10.1177/0278364913490533.

[12] C. Bodden, D. Rakita, B. Mutlu, M. Gleicher, A flexible optimization-based method for synthesizing intent-expressive robot arm motion, Int. J. Robot. Res. 37 (11) (2018) 1376–1394, http://dx.doi.org/10.1177/0278364918792295.

[13] B. Cohen, S. Chitta, M. Likhachev, Single- and dual-arm motion planning with heuristic search, Int. J. Robot. Res. 33 (2) (2014) 305–320, http://dx.doi.org/10.1177/0278364913507983.

[14] I. Ko, B. Kim, F.C. Park, Randomized path planning on vector fields, Int. J. Robot. Res. 33 (13) (2014) 1664–1682, http://dx.doi.org/10.1177/0278364914545812.

[15] L. Jaillet, J.M. Porta, Path planning under kinematic constraints by rapidly exploring manifolds, IEEE Trans. Robot. 29 (1) (2013) 105–117, http://dx.doi.org/10.1109/TRO.2012.2222272.

[16] M. Kazemi, K.K. Gupta, M. Mehrandezh, Randomized kinodynamic planning for robust visual servoing, IEEE Trans. Robot. 29 (5) (2013) 1197–1211, http://dx.doi.org/10.1109/TRO.2013.2264865.

[17] Z. Zhang, Y. Lin, S. Li, Y. Li, Z. Yu, Y. Luo, Tricriteria optimization-coordination motion of dual-redundant-robot manipulators for complex path planning, IEEE Trans. Control Syst. Technol. 26 (4) (2018) 1345–1357, http://dx.doi.org/10.1109/TCST.2017.2709276.

[18] Z. Zhang, S. Chen, S. Li, Compatible convex–nonconvex constrained QP-based dual neural networks for motion planning of redundant robot manipulators, IEEE Trans. Control Syst. Technol. 27 (3) (2019) 1250–1258, http://dx.doi.org/10.1109/TCST.2018.2799990.

[19] O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, Int. J. Robot. Res. 5 (1) (1986) 90–98, http://dx.doi.org/10.1177/027836498600500106.

[20] S. Haddadin, H. Urbanek, S. Parusel, D. Burschka, J. Roßmann, A. Albu-Schäffer, G. Hirzinger, Real-time reactive motion generation based on variable attractor dynamics and shaped velocities, in: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan, 2010, pp. 3109–3116, http://dx.doi.org/10.1109/IROS.2010.5650246.

[21] F. Flacco, T. Kröger, A. De Luca, O. Khatib, A depth space approach to human–robot collision avoidance, in: 2012 IEEE International Conference on Robotics and Automation, Saint Paul, MN, USA, 2012, pp. 338–345, http://dx.doi.org/10.1109/ICRA.2012.6225245.

[22] D.H.P. Nguyen, M. Hoffmann, A. Roncone, U. Pattacini, G. Metta, Compact real-time avoidance on a humanoid robot for human–robot interaction, in: HRI '18: Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction, Chicago, IL, USA, 2018, pp. 416–424, 10.1145/3171221.3171245.

[23] W. Merkt, V. Ivan, S. Vijayakumar, Continuous-time collision avoidance for trajectory optimization in dynamic environments, in: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, Macau, 2019, pp. 7248–7255, http://dx.doi.org/10.1109/IROS40897.2019.8967641.

[24] H. Nascimento, M. Mujica, M. Benoussaad, Collision avoidance interaction between human and a hidden robot based on kinect and robot data fusion, IEEE Robot. Autom. Lett. 6 (1) (2021) 88–94, http://dx.doi.org/10.1109/LRA.2020.3032104.

[25] B. Lacevic, P. Rocco, A.M. Zanchettin, Safety assessment and control of robotic manipulators using danger field, IEEE Trans. Robot. 29 (5) (2013) 1257–1270, http://dx.doi.org/10.1109/TRO.2013.2271097.

[26] M. Parigi Polverini, A.M. Zanchettin, P. Rocco, A computationally efficient safety assessment for collaborative robotics applications, Robot. Comput.-Integr. Manuf. 46 (2017) 25–37, http://dx.doi.org/10.1016/j.rcim.2016.11.002.

[27] F. Flacco, A. De Luca, O. Khatib, Control of redundant robots under hard joint constraints: Saturation in the null space, IEEE Trans. Robot. 31 (3) (2015) 637–654, http://dx.doi.org/10.1109/TRO.2015.2418582.

[28] D.Q. Mayne, J.B. Rawlings, C.V. Rao, P.O. Scokaert, Constrained model predictive control: Stability and optimality, Automatica 36 (6) (2000) 789–814, http://dx.doi.org/10.1016/S0005-1098(99)00214-9.

[29] E.F. Camacho, C. Bordons, Model Predictive Control, second ed., Springer, London, 2007, http://dx.doi.org/10.1007/978-0-85729-398-5.

[30] G. Buizza Avanzini, A.M. Zanchettin, P. Rocco, Constrained model predictive control for mobile robotic manipulators, Robotica 36 (1) (2018) 19–38, http://dx.doi.org/10.1017/S0263574717000133.

[31] A.S. Sathya, J. Gillis, G. Pipeleers, J. Swevers, Real-time robot arm motion planning and control with nonlinear model predictive control using augmented Lagrangian on a first-order solver, in: 2020 European Control Conference, ECC, St. Petersburg, Russia, 2020, pp. 507–512, http://dx.doi.org/10.23919/ECC51009.2020.9143732.

[32] N. Lucci, B. Lacevic, A.M. Zanchettin, P. Rocco, Combining speed and separation monitoring with power and force limiting for safe collaborative robotics applications, IEEE Robot. Autom. Lett. 5 (4) (2020) 6121–6128, http://dx.doi.org/10.1109/LRA.2020.3010211.

[33] L. Joseph, J. Pickard, V. Padois, D. Daney, Online velocity constraint adaptation for safe and efficient human–robot workspace sharing, in: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, Las Vegas, NV, USA, 2020, pp. 11045–11051, http://dx.doi.org/10.1109/IROS45743.2020.9340961.

[34] C. Byner, B. Matthias, H. Ding, Dynamic speed and separation monitoring for collaborative robot applications – Concepts and performance, Robot. Comput.-Integr. Manuf. 58 (2019) 239–252, http://dx.doi.org/10.1016/j.rcim.2018.11.002.

[35] P. Svarny, M. Tesar, J.K. Behrens, M. Hoffmann, Safe physical HRI: Toward a unified treatment of speed and separation monitoring together with power and force limiting, in: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, Macau, 2019, pp. 7580–7587, http://dx.doi.org/10.1109/IROS40897.2019.8968463.

[36] E. Magrini, F. Ferraguti, A.J. Ronga, F. Pini, A. De Luca, F. Leali, Human–robot coexistence and interaction in open industrial cells, Robot. Comput.-Integr. Manuf. 61 (2020) 101846, http://dx.doi.org/10.1016/j.rcim.2019.101846.

[37] D. Han, H. Nie, J. Chen, M. Chen, Dynamic obstacle avoidance for manipulators using distance calculation and discrete detection, Robot. Comput.-Integr. Manuf. 49 (2018) 98–104, http://dx.doi.org/10.1016/j.rcim.2017.05.013.

[38] M. Kimmel, S. Hirche, Invariance control for safe human–robot interaction in dynamic environments, IEEE Trans. Robot. 33 (6) (2017) 1327–1342, http://dx.doi.org/10.1109/TRO.2017.2750697.

[39] A. Pereira, M. Althoff, Calculating human reachable occupancy for guaranteed collision-free planning, in: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, Vancouver, BC, Canada, 2017, pp. 4473–4480, http://dx.doi.org/10.1109/IROS.2017.8206314.

[40] E. Garone, M. Nicotra, Explicit reference governor for constrained nonlinear systems, IEEE Trans. Automat. Control 61 (5) (2016) 1379–1384, http://dx.doi.org/10.1080/00207179.2017.1317832.

[41] M.M. Nicotra, E. Garone, The explicit reference governor: A general framework for the closed-form control of constrained nonlinear systems, IEEE Control Syst. Mag. 38 (4) (2018) 89–107, http://dx.doi.org/10.1109/MCS.2018.2830081.

[42] K. Merckaert, M.M. Nicotra, B. Vanderborght, E. Garone, Constrained control of robotic manipulators using the explicit reference governor, in: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, Madrid, Spain, 2018, pp. 5155–5162, http://dx.doi.org/10.1109/IROS.2018.8593857.

[43] B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo, Robotics: Modeling, Planning, and Control, Springer, London, 2010, http://dx.doi.org/10.1007/978-1-84628-642-1.

[44] H. Kazerooni, Compliant motion control for robot manipulators, Internat. J. Control 49 (3) (1989) 745–760, http://dx.doi.org/10.1080/00207178908559664.

[45] R. Smits, KDL: Kinematics and Dynamics Library, http://www.orocos.org/kdl.

[46] Franka Emika GmbH, Robot and interface specification, https://frankaemika.github.io/docs/control_parameters.html#constants.