TYPE Original Research
PUBLISHED 03 January 2025
DOI 10.3389/frobt.2024.1447351

Check for updates

OPEN ACCESS

EDITED BY
Holger Voos,
University of Luxembourg, Luxembourg

REVIEWED BY
Sunan Huang,
National University of Singapore, Singapore
Jundong Wu,
China University of Geosciences
Wuhan, China

*CORRESPONDENCE
Marvin Becker,
becker@irt.uni-hannover.de

RECEIVED 11 June 2024
ACCEPTED 12 November 2024
PUBLISHED 03 January 2025

CITATION
Becker M, Caspers P, Lilge T, Haddadin S and Müller MA (2025) Informed circular fields: a global reactive obstacle avoidance framework for robotic manipulators.
Front. Robot. AI 11:1447351.
doi: 10.3389/frobt.2024.1447351

Informed circular fields: a global reactive obstacle avoidance framework for robotic manipulators

Marvin Becker[1]*, Philipp Caspers[1], Torsten Lilge[1], Sami Haddadin[2] and Matthias A. Müller[1]

[1]Institute of Automatic Control, Leibniz University Hannover, Hannover, Germany, [2]Munich Institute of Robotics and Machine Intelligence, Technische Universität München (TUM), Munich, Germany

In this paper, we present a global reactive motion planning framework designed for robotic manipulators navigating in complex dynamic environments. Utilizing local minima-free circular fields, our methodology generates reactive control commands while also leveraging global environmental information from arbitrary configuration space motion planners to identify promising trajectories around obstacles. Furthermore, we extend the virtual agents framework introduced in Becker et al. (2021) to incorporate this global information, simulating multiple robot trajectories with varying parameter sets to enhance avoidance strategies. Consequently, the proposed unified robotic motion planning framework seamlessly combines global trajectory planning with local reactive control and ensures comprehensive obstacle avoidance for the entire body of a robotic manipulator. The efficacy of the proposed approach is demonstrated through rigorous testing in over 4,000 simulation scenarios, where it consistently outperforms existing motion planners. Additionally, we validate our framework's performance in real-world experiments using a collaborative Franka Emika robot with vision feedback. Our experiments illustrate the robot's ability to promptly adapt its motion plan and effectively avoid unpredictable movements by humans within its workspace. Overall, our contributions offer a robust and versatile solution for global reactive motion planning in dynamic environments.

KEYWORDS

autonomous robotic systems, guidance navigation and control, real-time collision avoidance, robotic manipulation arm, motion planning

1 Introduction

1.1 Motivation

In recent years, the prerequisites of industrial production and assembly have changed significantly and the ensuing challenges are constantly evolving. Continuously decreasing product life cycles, uncertain product volumes and rapidly growing product variants due to the growing trend towards mass customization have led to an increasing demand for flexible, adaptive multi-purpose manufacturing and assembly systems (El Zaatari et al., 2019; Krüger et al., 2009; Reinhart and Loy, 2010; Fechter et al., 2016).

Although classical industrial robots have proven to be efficient tools for repeatable tasks in traditional mass production, they need to be operated behind safety fences in dedicated areas to ensure the safety of human coworkers, which limits their flexibility and reusability while increasing their changeover times, costs and space requirements (Matheson et al., 2019; Barbazza et al., 2017).

Human robot interaction (HRI) is generally seen as a promising solution to increase flexibility while reducing production and assembly costs in continuously changing and uncertain market environments (Matheson et al., 2019; Kragic et al., 2018; Fechter et al., 2016; Krüger et al., 2009; El Zaatari et al., 2019; Faccio et al., 2019; Meziane et al., 2017). However, despite the widely recognized potential and the long-standing availability of the technology, the use of collaborative systems is still almost exclusively limited to areas where no direct contact with humans is necessary and the workspaces are structurally separated from each other. Moreover, robots employed in industry typically adhere to rigidly pre-programmed trajectories and routines and possess only limited capabilities to interact or react to changing environmental conditions (Kragic et al., 2018; El Zaatari et al., 2019).

To cope with the challenges in the dynamic and unpredictable environments around humans, robots must be able to adapt quickly to changing conditions and generate new trajectories in real time (Kappler et al., 2018; Liu et al., 2022).

Collision avoidance is a key component for solving such motion planning problems and while a lot of research has been conducted in this field, challenges persist, in particular in dynamic environments Huang et al. (2019); Niloy et al. (2021).

1.2 Related work

The subsequent overview of the related work is organized by following the allocation from (Kappler et al., 2018), which classifies motion planning approaches into the categories *sense-plan-act*, *locally reactive control* and *reactive planning*.

System architectures that follow the *sense-plan-act* paradigm are characterized by a rather strict separation of perception, motion planning and control. Typically, sensor feedback is considered at the initial stage to update an environment model. The motion planner then uses this model to identify a preliminary, coarse path towards the goal. Subsequently, a separate controller is employed to track this collision-free path (Kappler et al., 2018). Due to the vast amount of research in this field, we focus our review on sampling-based planning approaches in this category.

Sampling-based planners typically establish a connectivity graph between the initial and goal configurations of the robot by connecting random samples in the search space. The concept received significant attention in the field of motion planning over the last decades due to its ability to handle high degrees of freedom, and its straightforward implementation (Connell and La, 2018).

The rapidly-exploring random trees (RRT) (LaValle, 1998) and the probabilistic roadmap method (PRM) (Kavraki et al., 1996) are two of the most commonly used sampling-based approaches in robotics (Short et al., 2016).

The RRT algorithm incrementally constructs a search tree in the robot's configuration space to find a feasible path to the goal pose starting from the initial robot configuration. A disadvantage of the RRT planner and its variants is the lack of quality in terms of path length, which led to the development of an asymptotically optimal variant, the RRT* planner (Karaman and Frazzoli, 2011). Given enough run-time, the RRT*was shown to converge to an optimal solution by rewiring the search tree and continuously adding new nodes even after an initial solution was found.

In contrast to RRT approaches, the PRM algorithm consists of two phases, a learning phase and a query phase. During the learning phase, the configuration space of the robot is randomly sampled and robot configurations that collide with known obstacles in the task space are rejected. Afterwards, the resulting roadmap is used as a basis for the query phase, where the shortest path from a start to a goal pose is calculated (Kavraki et al., 1996). Similar to RRT, an asymptotically optimal variant, the PRM*was developed, which led to significantly improved path quality (Karaman and Frazzoli, 2011).

Despite recent advances and efficient implementations with replanning capabilities, such as the RRT*FN-Dynamic (RRT*FND) planner (Adiyatov and Varol, 2017), Batch Informed Trees (BIT*) (Gammell et al., 2020), the Bidirectional Informed RRT*(BI^{2}RRT*) (Burget et al., 2016) and a dynamically replanned RRT*(Connell and La, 2017), sampling-based approaches still face significant challenges. They tend to have high computational costs (Grothe et al., 2022) and often require post-processing steps to smooth and shorten the generated trajectories (Schulman et al., 2014). Additionally, their performance can degrade notably when navigating through narrow passages (Li and Dantam, 2023).

A seminal work in the area of *locally reactive control* is the artificial potential field (APF) approach, where the robot is controlled by artificial repulsive and attractive forces for a collision-free motion to the goal pose (Khatib, 1986). While the algorithm requires low computational resources, it suffers from local minima. This can cause the robot to converge towards them instead of reaching the goal pose, depending on the environment. Many variants of APFs or related approaches were proposed to overcome this limitation and to enable goal convergence in a wider range of applications, notably the harmonic potential functions (Connolly et al., 1990), which was further extended by Khansari-Zadeh and Billard (2012), and successfully used to avoid static and dynamic obstacles with the endeffector (EE) of a 7-degree of freedom (DoF) manipulator. Related methods use repulsive forces or velocities for collision avoidance of the whole structure of the robot, which were either applied on predefined control points along the robot structure (Chen and Song, 2018; Li et al., 2019) or use the closest distance between robot and obstacle (Wang et al., 2018). Further extensions exploit the robot's nullspace in order to achieve collision avoidance and simultaneously maintain a predefined EE trajectory, e.g., the approach from Cefalo et al. (2017), where a focus is placed on parallel computation and collision checks on a GPU. Similarly, the approach by (Flacco et al., 2012) developed a fast method for the minimal distance calculation, which was employed to determine repulsive forces for collision avoidance and successfully implemented on a 7-DoF robot.

Inspired by the behavior of charged particles in electromagnetic fields, the authors in (Singh et al., 1996; Singh et al., 1997) developed the circular field (CF) approach. CFs apply a virtual force similar to the Lorentz force on the robot, which results in smooth trajectories around obstacles. The virtual force does not induce any additional energy into the system as it always acts perpendicular to the robot's velocity and thus does not suffer from local minima. The original algorithm was extended in Haddadin et al. (2011) as it suffered from oscillations due to inconsistently defined artificial currents. Therefore, a rotation vector is introduced for each obstacle in order to define a consistent artificial current flow for each obstacle. An alternative approach was presented in a series of works (Ataka et al., 2018a; Ataka et al., 2018b; Ataka et al., 2018c; Ataka et al., 2022), enabling the algorithm to be used in unknown environments. This is achieved by projecting the robot's velocity vector onto the obstacle to define a continuous artificial current, thus avoiding the calculation of the rotation vector.

Nevertheless, current CF and APF approaches can only serve as local planners and perform poorly for finding global optimal or even suboptimal solutions because of their limited exploration possibilities.

Reactive planners are hybrid motion planners designed to respond quickly to local changes while improving and correcting the global path in response to larger environmental changes (Kappler et al., 2018). Such hybrid approaches have been extensively studied in the literature of mobile robotics and unmanned aerial vehicles (UAV). Implementations often combine PRM and APF (Ravankar et al., 2020), RRT and APF (Yingqi et al., 2021) or variations of existing sampling-based strategies with custom reactive controllers, e.g., lazy PRM with a reactive controller for dynamic obstacles (Sánchez et al., 2006) or RRT-connect with a reactive control law employing a sliding mode control scheme (Elmokadem and Savkin, 2021).

While these hybrid planners work effectively in lower DoF mobile robotic systems, their applications to systems with higher DoF, such as robotic manipulators, are relatively sparse (Kappler et al., 2018). For example, (Liu and Jiang, 2018) proposes an approach that alternates between RRT for static obstacles and an APF variant for dynamic obstacles, while (Li et al., 2021) deform the trajectory of an RRT planner using locally reactive control for a 7-DoF manipulator. Similarly, the elastic strip framework (Brock and Khatib, 2002) modifies global candidate paths using APF for real-time reactions to environmental changes.

The authors in Kappler et al. (2018) use global Riemannian motion optimization and locally reactive control for several experiments with a 7-DoF robot arm. In their work, they conduct a comprehensive comparison of *sense-plan-act* methods, *locally reactive control* and *reactive planning*. Their findings suggest that reactive methods have distinct advantages over traditional approaches, particularly in dynamic and uncertain environments. Even in static environments, *reactive planners* showed competitive, and sometimes superior, performance over traditional planning methods, due to their ability to initiate movement without excessive pre-planning. *Reactive planners*, by integrating global information, provide superior adaptability in complex environments compared to purely local methods.

However, a significant limitation of these existing hybrid approaches is the strict separation between global and local planning. In such methods, the global planner and local planner are typically treated as distinct modules, with the local planner often following the global planner's path without fully leveraging the global insights. This can lead to inefficient planning, where local reactivity may disrupt global goals, or the global planner's computational cost is prohibitive in dynamic environments.

In our previous work Becker et al. (2021), we introduced the circular field predictions (CFP) method, which enhanced the original CF approach by incorporating a predictive virtual agent framework for global path exploration. The CFP planner demonstrated computational efficiency and high-quality path generation on a 7-DoF manipulator. In subsequent work Becker et al. (2023b), we provided rigorous proofs of collision avoidance and goal convergence, validating the theoretical soundness of the approach. However, these efforts were primarily focused on end-effector obstacle avoidance and did not fully explore full-body obstacle avoidance for manipulators.

This paper builds on these previous efforts by extending the CFP planner to handle full-body obstacle avoidance. Unlike traditional methods, which often focus only on the end-effector or parts of the robot, our approach ensures that the entire manipulator avoids obstacles while maintaining smooth, collision-free trajectories. This full-body approach is particularly useful in environments where dynamic and cluttered obstacles pose significant challenges to the whole body of the robot.

One significant advantage of our approach over existing hybrid planners is the efficient use of global planning insights within the local planning strategy. By avoiding the strict separation between global and local planning found in other methods, our approach ensures that global information is dynamically leveraged in real-time, leading to more efficient and smooth obstacle avoidance. This allows the local reactive controller to not just follow the global path but to dynamically adjust the trajectory based on the global strategy. Moreover, our motion planning strategy is flexible enough to integrate arbitrary global planners, allowing the framework to adapt to different problem settings and robot configurations.

In summary, traditional motion planning methods, while effective in handling high DoF systems, often face significant computational challenges, especially in dynamic environments. Reactive planners provide advantages in real-time reactivity but struggle with global optimization and goal convergence. The proposed approach presents a hybrid planning framework that balances global trajectory planning with local reactivity, ensuring full-body obstacle avoidance in highly dynamic and uncertain environments. This novel approach extends the state of the art in motion planning for robotic manipulators, offering a scalable, computationally efficient solution for future robotic applications in complex environments.

1.3 Contribution

In this paper, we extend the CFP algorithm to encompass full-body avoidance for the entire structure of a robotic manipulator, thereby developing a comprehensive robotic motion planning framework that bridges the gap between global trajectory planning and reactive control. The main contributions of this paper are.

- Development of a motion planning framework for full-body obstacle avoidance of robotic manipulators by introducing additional control points along the robot structure and defining suitable control forces.
- Integration of global environment information from arbitrary global planners in the virtual agent

framework from (Becker et al., 2021) resulting in the informed circular field (ICF) planner.
- Two algorithms for leveraging global information about promising avoidance directions with the reactive ICF planner.
- The performance of 20 different global planners within the framework is compared in 10 different environments with a total of 4,000 simulations.
- Extensive comparison of the ICF planner against widely used global and local motion planning approaches in a total of more than 200 simulations.
- Verification of the proposed algorithm in real-world experiments, where a 7-DoF Franka Emika Research 3 robot avoids dynamic motions of a human in its workspace.
- Making the planning framework available for the community by providing the source code[1].

1.3.1 Relationship to previous publications

Note that a preliminary version of parts of this paper has appeared in the conference paper (Becker et al., 2023a). In contrast to our prior work, this article provides a more thorough description of the planner details. Specifically, we expand on the motivation and related work section, provide a detailed description of the calculations for all forces and methods involved in generating robot control commands, and explain the extension of the virtual agent framework, including the new reward function and its structure in detail–elements that had to be omitted in Becker et al. (2023a) due to space limitations. Furthermore, we extend the methodology for the robot control signal calculation, provide an open-source implementation of our code base, and conduct additional extensive simulations to determine appropriate global pre-planners. Lastly, we implement the planning framework on a 7-DoF Franka Emika robot, integrating a vision system for the detection and tracking of humans within the robot workspace for performing real-world experiments to demonstrate the efficacy of the motion planning framework.

2 Motion planning framework

In this section, we introduce our unified motion planning framework that consists of a reactive CF obstacle avoidance algorithm inspired by electromagnetic fields combined with a global motion planning approach. The global planning component leverages the results from existing joint space motion planners by extracting information about potential avoidance directions around obstacles, which are subsequently exploited by a virtual agent framework for efficient global exploration.

2.1 Reactive motion planning

We start the description of the reactive planning component with an introduction to the steering forces acting on the EE of the robot. Then, we continue with a presentation of the forces for full-body obstacle avoidance and additional forces and supplementary methods for enhancing the overall motion planning strategy of robotic manipulators. The combination of all control commands for generating a total reference signal for joint space control is described in Section 2.1.9.

Our reactive vector-field approach is based on the definition of several virtual forces to generate efficient, collision-free trajectories, which are superimposed to form the resulting steering force $\boldsymbol{f}_{\mathrm{s}}$ for controlling the EE

$$\boldsymbol{f}_{\mathrm{s}} = \boldsymbol{f}_{\mathrm{vlc}} + \boldsymbol{f}_{\mathrm{cf}} + \boldsymbol{f}_{\mathrm{cf,rep}}. \tag{1}$$

It consists of an attractive potential field force $\boldsymbol{f}_{\mathrm{vlc}}$ for goal convergence and CF-based obstacle avoidance forces $\boldsymbol{f}_{\mathrm{cf}}$ and $\boldsymbol{f}_{\mathrm{cf,rep}}$, which are explained in detail in the following sections.

2.1.1 Attractive goal force

In order to guide the robot EE to its goal pose, we extend the definition of potential field attractor dynamics with the proposed velocity limiting controller (VLC) from Khatib (1986). For this purpose, we define an artificial desired velocity in the form

$$\boldsymbol{v}_{\mathrm{d}} = \frac{k_{\mathrm{p}}}{k_{\mathrm{v}}}\left(\boldsymbol{x}_{\mathrm{g}} - \boldsymbol{x}\right),$$

where $\boldsymbol{x} \in \mathbb{R}^3$ is the current robot position, $\boldsymbol{x}_{\mathrm{g}} \in \mathbb{R}^3$ is the translational part of the goal pose, $k_{\mathrm{p}} > 0$ is the position gain and $k_{\mathrm{v}} > 0$ the velocity gain. Note that the orientation is considered separately below. The virtual attractive force $\boldsymbol{f}_{\mathrm{vlc,t}}$ is then calculated from the difference of the current robot velocity $\dot{\boldsymbol{x}}$ and the artificial desired velocity $\boldsymbol{v}_{\mathrm{d}}$

$$\boldsymbol{f}_{\mathrm{vlc,t}} = k_{\mathrm{v}}\left(\nu \boldsymbol{v}_{\mathrm{d}} - \dot{\boldsymbol{x}}\right).$$

The factor ν is used to limit the force when the robot velocity reaches a defined maximum magnitude $\dot{x}_{\max}$ in the direction of the goal

$$\nu = \min\left(1, \frac{\dot{x}_{\max}}{\|\boldsymbol{v}_{\mathrm{d}}\|}\right).$$

Compared to a classical potential field, the resulting control law is more appropriate for longer distances between the robot and the goal pose. In fact, the generated virtual force is equal to zero when the robot moves towards the goal pose at maximum velocity [as shown in (Becker et al., 2023b)]. This leads to a constant velocity magnitude except in the vicinity of the start, goal and obstacles when the robot is subject to further virtual forces. Note that while the VLC controller explicitly enforces a velocity limit, the acceleration limit is only implicitly considered in the current formulation, and it is not directly constrained. Furthermore, even though the control force is restricted by the velocity limit, the gains k_{p} and k_{v} should still be selected with caution to avoid undesirable dynamic behaviors such as oscillations or overshooting. The VLC controller functions similarly to a proportional-derivative (PD) controller, where the balance between these gains is critical. In particular, to achieve stable and smooth control, the gains should be chosen to approximate critical damping in a PD-control scheme, which typically requires setting $k_{\mathrm{v}} \approx 2\sqrt{k_{\mathrm{p}}}$. By choosing k_{p} and k_{v} carefully, the robot can maintain smooth and stable motion even in challenging scenarios, while adhering to the velocity limits imposed by the controller.

We extend the approach to include an additional rotation for the robot to achieve the desired orientation $\boldsymbol{x}_{\mathrm{g,r}}$ represented in Euler

1 https://gitlab.com/roboterfabrik1/icf_planner

angles. To do this, we define an artificial desired angular velocity $\boldsymbol{\omega}_\mathrm{d}$ based on the orientation error $\boldsymbol{x}_{\mathrm{g,r}} - \boldsymbol{x}_\mathrm{r}$. This error is calculated using the difference between the quaternion representing the robot's current orientation, $\boldsymbol{p} = \begin{pmatrix} p_0 & \boldsymbol{p}_{\mathrm{im}}^\top \end{pmatrix}^\top = \begin{pmatrix} p_0 & p_1 & p_2 & p_3 \end{pmatrix}^\top$, and the quaternion for the desired goal orientation, $\boldsymbol{p}_\mathrm{g} = \begin{pmatrix} p_{\mathrm{g},0} & \boldsymbol{p}_{\mathrm{g,im}}^\top \end{pmatrix}^\top = \begin{pmatrix} p_{g,0} & p_{g,1} & p_{g,2} & p_{g,3} \end{pmatrix}^\top$, as described by Yuan, (1988)

$$\boldsymbol{x}_{\mathrm{g,r}} - \boldsymbol{x}_\mathrm{r} = p_0 \boldsymbol{p}_{\mathrm{g,im}} - p_{\mathrm{g},0} \boldsymbol{p}_{\mathrm{im}} - \boldsymbol{p}_{\mathrm{im}} \times \boldsymbol{p}_{\mathrm{g,im}}.$$

In this formulation, p_0 and $p_{\mathrm{g},0}$ are the scalar components, while $\boldsymbol{p}_{\mathrm{im}}$ and $\boldsymbol{p}_{\mathrm{g,im}}$ represent the vector (imaginary) components of the quaternions describing the robot's current and desired orientations, respectively.

The resulting rotational component is then calculated with.

$$\boldsymbol{\omega}_\mathrm{d} = \frac{k_\mathrm{p}}{k_\mathrm{v}} \left(\boldsymbol{x}_{\mathrm{g,r}} - \boldsymbol{x}_\mathrm{r} \right),$$

$$v_\mathrm{r} = \min\left(1, \frac{\omega_{\max}}{\|\boldsymbol{\omega}_\mathrm{d}\|}\right),$$

$$\boldsymbol{f}_{\mathrm{vlc,r}} = k_\mathrm{v} \left(v_\mathrm{r} \boldsymbol{\omega}_\mathrm{d} - \boldsymbol{\omega} \right),$$

where $\omega_{\max}$ defines the maximum angular velocity and $\boldsymbol{\omega}$ the current angular velocity of the robot. Please note that the rotation of the robot is only used for reaching a desired orientation and not for avoiding obstacles. This also implies that the additional scaling k_{vlc} is not needed for the calculation of $\boldsymbol{f}_{\mathrm{vlc,r}}$.The total attractive force is a concatenation of the translational and rotational part

$$\boldsymbol{f}_{\mathrm{vlc}} = \begin{bmatrix} \boldsymbol{f}_{\mathrm{vlc,t}} \\ \boldsymbol{f}_{\mathrm{vlc,r}} \end{bmatrix}.$$

2.1.2 Endeffector obstacle avoidance

In this section, we present our adaption of the CF algorithm, which is used in our motion planning framework to efficiently avoid obstacles in the robot's environment. First, we present the definition and our extensions to nominal boundary following CFs. Then, we introduce the second obstacle avoidance force, which adds a repulsive component. Our proposed algorithm works directly with point-cloud data, thus avoiding computationally intensive and error-prone segmentation of obstacle surfaces. For this purpose, we consider $j = 1, \ldots, n_\mathrm{o}$ obstacles which are each characterized by a cloud of points ${}^j_i\boldsymbol{x}_\mathrm{o} \in {}^j\mathbb{O} \subset \mathbb{R}^{3\times l_j}$, where $i = 1, \ldots, l_j$. As a result, each point ${}^j_i\boldsymbol{x}_\mathrm{o}$ in an obstacle point cloud ${}^j\mathbb{O}$ generates its own magnetic field and its own obstacle avoiding force instead of relying on obstacle surfaces. Moreover, in contrast to a majority of approaches in the literature, we exploit more information about the obstacles beyond the single obstacle point with the minimum distance. This approach reduces sensitivity to sensor noise and improves avoidance behavior in the presence of multiple obstacles, without causing oscillations. The computational load can be managed by reducing the resolution of the sensor point cloud through downsampling. Furthermore, our implementation offers the option to include only the closest $m_j \in [0, l_j]$ points of each obstacle for the force calculations. We assume that the obstacle data originates from common motion tracking devices such as laser scanners or camera modules and make the assumption that the point cloud points are reasonably evenly distributed.

2.1.2.1 Nominal circular fields

CFs were first introduced in Singh et al. (1996) and are inspired by the forces acting on a moving charged particle in an electromagnetic field. More specifically, the law of Biot-Savart states that the magnetic field in a distance $\boldsymbol{x}$ from a wire of infinitesimal length dl carrying the current I is defined by

$$d\boldsymbol{B}(\boldsymbol{x}) = \frac{\mu_0}{4\pi} \frac{I dl \times \boldsymbol{x}}{\|\boldsymbol{x}\|^3}$$

and will apply the Lorentz force

$$\boldsymbol{F} = q\dot{\boldsymbol{x}} \times \boldsymbol{B}$$

on a particle charged with q and moving with velocity $\dot{\boldsymbol{x}}$, where μ_0 specifies a permeability constant (Halliday et al., 2013). The Lorentz force acts perpendicular to the direction of motion of the charged particle, altering its direction of movement without changing its velocity magnitude.

In our application on collision avoidance, we use this physical law as an inspiration and interpret the robot as a charged particle that moves in virtual electromagnetic fields which are generated by virtual currents $\boldsymbol{c}$ on all obstacles points. Figure 1 illustrates the virtual electromagnetic fields generated by a single obstacle in two-dimensional (2D) space and the resulting force on the robot. The direction of the virtual current vector defines the direction of the CF force originating from an electromagnetic field of an obstacle point. As stated in Haddadin et al. (2011), the original definition of the current vector from Singh et al. (1996) is not sufficient because inconsistent orientations of the current vectors on an obstacle lead to oscillations. In order to generate consistent current vectors, we define a magnetic field vector ${}^j\boldsymbol{b} \in \mathbb{R}^3$ with $\|{}^j\boldsymbol{b}\| = 1$ for each obstacle j, which determines the direction of the current vectors uniformly over the entire obstacle [cf. (Haddadin et al., 2011)].

Note that the magnetic field vector is a crucial element in our obstacle avoidance strategy as it is used to calculate the virtual current vectors, and thus defines the direction in which the robot will pass an obstacle. The choice of the magnetic field vectors therefore has a decisive influence on the overall behavior of the robot and can also be interpreted as the global planning component of our motion planning framework (cf. Section 2.2).

After the definition of the magnetic field vector, the artificial current vector for a point i of an obstacle j can be calculated with

$${}^j_i\boldsymbol{c} = {}^j_i\boldsymbol{n} \times {}^j\boldsymbol{b},$$

where ${}^j_i\boldsymbol{n}$ is the normalized obstacle surface normal pointing outside of the obstacle. Various approaches for surface normal approximation of point clouds exist in the literature. Throughout this paper, we use the functionality provided by the Point Cloud Library (Rusu and Cousins, 2011).

In contrast to previous approaches, our definition of the current vector (12) leads to a continuous current direction over the surfaces of the obstacles (in contrast to (Singh et al., 1996)) and can be easily used to explore multiple trajectories to evade obstacles without depending on the current robot velocity (in contrast to (Ataka et al., 2018b)).

We modify the Biot-Savart law for the use-case of obstacle avoidance, and in our formulation, each point i on an obstacle j

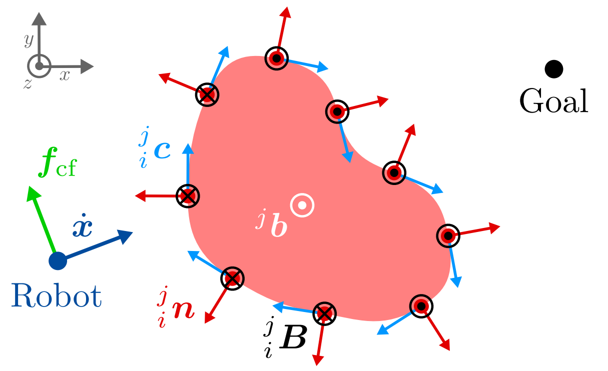

FIGURE 1
Generation of circular fields (black) and CF force (green) for an obstacle (light red), that is approximated by a point cloud (dark red points) in 2D. In this figure, we use the convention of representing 3D vectors perpendicular to the plane of the diagram with circles: a circle with a dot indicates a vector pointing out of the plane towards the viewer (positive z-direction), while a circle with an inscribed cross represents a vector pointing into the plane, away from the viewer (negative z-direction). A coordinate system is provided to clarify the orientation of the vectors. In this example, the magnetic field vector (white) is defined as $^j\boldsymbol{b} = (0 \quad 0 \quad 1)^\top$ and points outside of the page. The current vectors $^j_i\boldsymbol{c}$ are shown in light blue and the surface normals $^j_i\boldsymbol{n}$ in dark red. The resulting circular fields $^j_i\boldsymbol{B}$ are depicted in black and point either in the positive or negative z-direction.

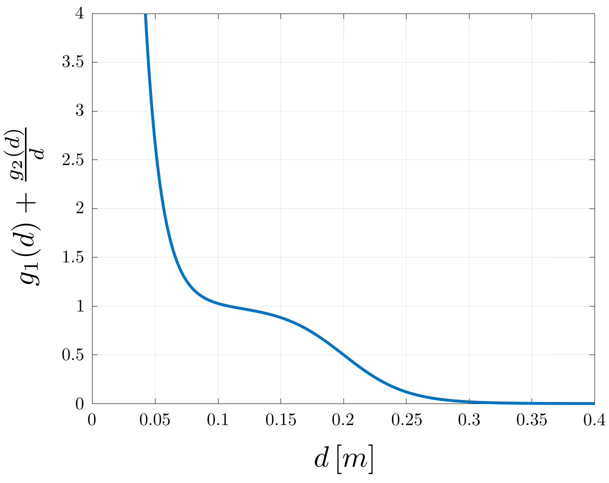

FIGURE 2
Visualization of the resulting scaling factor $g_1(^j_id) + \frac{g_2(^j_id)}{^j_id}$ using the settings $\gamma_{\text{sl},1} = 20.0, \gamma_{\text{d},1} = 0.2\text{m}, \gamma_{\text{sl},2} = 30.0, \gamma_{\text{d},2} = 0.01\text{m}$.

generates its own artificial electromagnetic field, that is, its own CF defined as

$$^j_i\boldsymbol{B} = {^j_i\boldsymbol{c}} \times \frac{^j_i\dot{\boldsymbol{d}}}{\|^j_i\dot{\boldsymbol{d}}\|}.$$

Here, $^j_i\boldsymbol{d} = {^j_i\boldsymbol{x}_{\text{o}}} - \boldsymbol{x}$ is the distance vector between the robot's position $\boldsymbol{x}$ and the position of the obstacle point $^j_i\boldsymbol{x}_{\text{o}}$ and $^j_i\dot{\boldsymbol{d}}$ is the respective relative velocity. An example of the electromagnetic fields on a static obstacle is shown in Figure 1.

When the robot moves in such a virtual electromagnetic field, the CF force (a modified version of the Lorentz force) is generated

$$^j_i\hat{\boldsymbol{f}}_{\text{cf}} = k_{\text{cf}}\left(g_1\left(^j_id\right) + \frac{g_2\left(^j_id\right)}{^j_id}\right)\frac{^j_i\dot{\boldsymbol{d}}}{\|^j_i\dot{\boldsymbol{d}}\|} \times {^j_i\boldsymbol{B}},$$

where $k_{\text{cf}} > 0$ describes a constant gain and $^j_id = \|^j_i\boldsymbol{d}\| - d_{\text{s}}$ is the distance between an obstacle and the robot including a safety margin d_{s}. We also use the logistic amplitude functions $g_1(^j_id)$ and $g_2(^j_id)$ defined as

$$g_r\left(^j_id\right) = \frac{1}{2}\left(1 + \tanh\left(\gamma_{\text{sl},r}\left(\gamma_{\text{d},r} - {^j_id}\right)\right)\right). \quad (2)$$

Here, the subscript r refers to the parameters associated with the logistic scaling functions g_1 and g_2, which modulate the circular field force magnitude based on the distance between the robot and the obstacle [inspired by Luo et al. (2014)]. These parameters, $\gamma_{\text{sl,r}}$ and $\gamma_{\text{d,r}}$ control the slope and activation distance of the force, ensuring a smooth activation of the CF force. The subscript r is used to distinguish between different scaling parameters that are applied in the calculation. The effect of the scaling factor $\left(g_1(^j_id) + \frac{g_2(^j_id)}{^j_id}\right)$ is illustrated in Figure 2.This definition of the CF force guides the robot along the boundary of obstacles, preventing collisions. To improve computational efficiency and to mitigate disturbances from obstacle points which are not relevant for the immediate avoidance maneuver, the planner will ignore obstacle points that.

1. Are outside of a range limit around the robot: $\|\boldsymbol{d}\| \geq d_{\text{max}}$,
2. Are not directed towards the robot, i.e., the absolute value of the angle between the obstacle surface normal $\boldsymbol{n}$ and the robot-obstacle distance vector $\boldsymbol{d}$ is smaller than 90°: $\boldsymbol{n} \cdot \boldsymbol{d} \geq 0$,
3. The robot moves away from: $\boldsymbol{n} \cdot \frac{\dot{\boldsymbol{d}}}{|\dot{\boldsymbol{d}}|} \geq \cos\varphi$ and at the same time the relative velocity points towards the goal: $(\boldsymbol{x}_{\text{g}} - \boldsymbol{x}) \cdot \dot{\boldsymbol{d}} >$ 0. Here, φ describes the angle between the obstacle surface normal and the relative velocity, and is set in this paper to 85°. This choice ensures that the robot takes into account all obstacle points within a 190° area in front of it, including those to which it moves parallel. The additional condition that the relative velocity points towards the goal is needed to ensure that the robot is able to evade trap-like obstacle shapes.

Using these criteria, the CF force takes the form

$$^j_i\boldsymbol{f}_{\text{cf}} = \begin{cases} 0, & \begin{array}{l}\text{if} \quad ^j_id \geq d_{\text{max}} \\ \text{if} \quad ^j_i\boldsymbol{n} \cdot {^j_i\boldsymbol{d}} \geq 0 \\ \text{if} \quad ^j_i\boldsymbol{n} \cdot \frac{^j_i\dot{\boldsymbol{d}}}{\|^j_i\dot{\boldsymbol{d}}\|} \geq \cos\left(^j_i\varphi\right) \wedge \left(\boldsymbol{x}_{\text{g}} - \boldsymbol{x}\right) \cdot {^j_i\dot{\boldsymbol{d}}} > 0\end{array} \\ ^j_i\hat{\boldsymbol{f}}_{\text{cf}}, & \text{otherwise.}\end{cases}$$

The total CF force from n_{o} obstacles with m_j relevant obstacle points (applying the above criteria) then results in

$$\boldsymbol{f}_{\text{cf}} = \sum_{j=0}^{n_{\text{o}}} \frac{1}{m_j} \sum_{i=0}^{m_j} {^j_i\boldsymbol{f}_{\text{cf}}}. \quad (3)$$

Note that the CF force only acts on the translation of the robot, i.e., $\boldsymbol{f}_{\text{cf}} \in \mathbb{R}^3$. Thus, we append zeros to the force vector for calculating the steering force in Equation 1.

In contrast to other reactive controllers, e.g., the APF approach, our CF planner has multiple advantages. The force is perpendicular to the robot's velocity, thus it does not dissipate any energy from the system and will not change the velocity magnitude of the robot. Moreover, as shown for point mass robots in Becker et al. (2023b), the planner does not suffer from local minima and consequently will not change the convergence property of attractive fields when no collision with obstacles occurs.

2.1.2.2 Repulsive circular fields

Although the boundary following CF force definition is typically sufficient to prevent collisions, we introduce an additional repulsive force to enable a more robust behavior when the motion of the robot is constrained and the guiding force in Equation 3 might not be sufficient to keep a safe distance to the obstacle. Towards this end, we add an additional repulsive CF-like force, where the artificial current is defined as the negative distance vector

$$ {}_i^j\hat{\boldsymbol{f}}_{\text{cf,rep}} = -k_{\text{cf,rep}} \quad g_3\left({}_i^j d\right) \quad \frac{{}_i^j\dot{\boldsymbol{d}}}{\|{}_i^j\dot{\boldsymbol{d}}\|} \times \left(\frac{{}_i^j\boldsymbol{d} \times {}_i^j\dot{\boldsymbol{d}}}{\|{}_i^j\boldsymbol{d} \times {}_i^j\dot{\boldsymbol{d}}\|} \right), \tag{4} $$

and we use the scaling factor $k_{\text{cf,rep}} \geq 0$ and the definition of $g_3({}_i^j d)$ from Equation 2. Note that Equation 4 is a simplification of the force from Ataka et al. (2022), which pushes the robot away from the obstacle while maintaining the useful properties of the original CF force, i.e., it is perpendicular to the robot velocity and therefore does not induce local minima.

Additionally, the definition employs similar criteria as the nominal CF force to exclude obstacle points from force generation and superimposes the forces from all obstacle points

$$ {}_i^j\boldsymbol{f}_{\text{cf,rep}} = \begin{cases} 0, & \begin{array}{l} \text{if} \quad {}_i^j d \geq d_{\text{max,rep}} \\ \text{if} \quad {}_i^j\boldsymbol{n} \cdot {}_i^j\boldsymbol{d} \geq 0 \\ \text{if} \quad {}_i^j\boldsymbol{n} \cdot \dfrac{{}_i^j\dot{\boldsymbol{d}}}{\|{}_i^j\dot{\boldsymbol{d}}\|} \geq \cos\left({}_i^j\varphi\right) \ \wedge \ \left(\boldsymbol{x}_{\text{g}} - \boldsymbol{x}\right) \cdot \dot{\boldsymbol{d}} > 0 \end{array} \\ {}_i^j\hat{\boldsymbol{f}}_{\text{cf,rep}}, & \text{otherwise,} \end{cases} $$

$$ \boldsymbol{f}_{\text{cf,rep}} = \sum_{j=0}^{n_{\text{o}}} \frac{1}{m_j} \sum_{i=0}^{m_j} {}_i^j\boldsymbol{f}_{\text{cf,rep}}, $$

where $d_{\text{max,rep}} \leq d_{\text{max}}$ is the distance limit for the repulsive CF force.

2.1.3 Attractive force scaling

The combination of the attractive force with obstacle avoidance forces can introduce new problems such as oscillations of the robot, or even induce new local minima and goal convergence issues as discussed in Ataka et al. (2018b). These problems are particularly noticeable in scenarios involving large or non-convex obstacles, where the robot has to move in a direction opposite to the goal position, causing a mutual cancellation of attractive and obstacle avoidance forces. In such cases, a potential solution is to follow the surface of obstacles until they are successfully bypassed, leveraging the boundary-following property inherent in CF forces. Furthermore, in our approach, safety is prioritized by giving precedence to obstacle avoidance over goal convergence. As a result, scaling factors are used exclusively to modify (or potentially deactivate) the attractive force, leaving the CF force unchanged. This strategy ensures that safety concerns are satisfied while addressing the challenges associated with more complex environments, which we show in our goal convergence analysis in a simplified planer setting in Becker et al. (2023b). Therefore, we modify the translational part of the attractive force by introducing the scaling factor k_{vlc}

$$ \boldsymbol{f}_{\text{vlc}} = \begin{bmatrix} k_{\text{vlc}} \boldsymbol{f}_{\text{vlc,t}} \\ \boldsymbol{f}_{\text{vlc,r}} \end{bmatrix}. $$

Please note that the rotation of the robot is only used for reaching a desired orientation and not for avoiding obstacles. This also implies that the additional scaling k_{vlc} is not needed for the calculation of $\boldsymbol{f}_{\text{vlc,r}}$. The goal force scaling factor is defined as

$$ k_{\text{vlc}} = \begin{cases} 0 & \text{if } \dot{\boldsymbol{x}} \cdot \boldsymbol{f}_{\text{vlc}} \leq 0 \wedge \|\dot{\boldsymbol{x}}\| \leq v_{\text{min}} \wedge \|\boldsymbol{x}_{\text{g}} - \boldsymbol{x}\| > \xi \\ w & \text{otherwise} \end{cases} \tag{5} $$

with $w = w_1 w_2 w_3$ and $w_1, w_2, w_3 \geq 0$.

Note that the CF force does not change the magnitude of the robot velocity (cf. Becker et al., 2023b), which is therefore only modified by the VLC. Using Equation 5 and therefore deactivating $\boldsymbol{f}_{\text{vlc}}$ when it works against the current motion direction while the velocity is below or equal to a defined v_{min}, we ensure that the robot will only decrease its velocity below this minimum, when the robot is in the vicinity $\xi > 0$ of the goal pose.

Consequently, when the attractive goal force is deactivated, the robot does not stop. Instead, the CF forces guide the robot to follow the obstacle's boundary, ensuring smooth obstacle avoidance without halting the motion towards the goal. The first two factors w_1 and w_2 are taken from Ataka et al. (2018b). The first factor is used to limit the VLC force when the robot is close to obstacles

$$ w_1 = 1 - \exp^{-\frac{\|\boldsymbol{d}\|}{\gamma_{\text{o}} d_{\text{max}}}}, $$

where $\gamma_{\text{o}} > 0$ is a constant scaling factor and $\|\boldsymbol{d}\| = \|\boldsymbol{x}_{\text{o}} - \boldsymbol{x}\|$ is the minimal distance between the robot end effector $\boldsymbol{x}$ and the closest obstacle point $\boldsymbol{x}_{\text{o}}$. When the robot approaches an obstacle, the VLC force converges to zero, prioritizing collision avoidance over goal convergence.

The second term reduces the attractive force when an obstacle is between the robot and the goal position and increases the force otherwise

$$ w_2 = 1 - \frac{\left(\boldsymbol{x}_{\text{g}} - \boldsymbol{x}\right) \cdot \boldsymbol{d}}{\|\boldsymbol{x}_{\text{g}} - \boldsymbol{x}\|\|\boldsymbol{d}\|}. $$

This factor is zero when the goal vector $\boldsymbol{x}_{\text{g}} - \boldsymbol{x}$ and the distance vector $\boldsymbol{d}$ point in the same direction, it is equal to one if the two vectors are orthogonal to each other and it doubles the influence of the VLC force when the vectors point in opposite directions.

The factor w_3 is introduced to handle environments with non-convex obstacles, where obstacle configurations might cause opposing VLC and CF forces. This would again result in a decrease of the robot's velocity as the goal force acts against the robot's direction of motion. In such a case the CF force should be dominating to guide the robot along the obstacles' boundaries until the obstacle is passed. This can be achieved by scaling the VLC force with the scalar product of the normalized VLC force and the current robot velocity

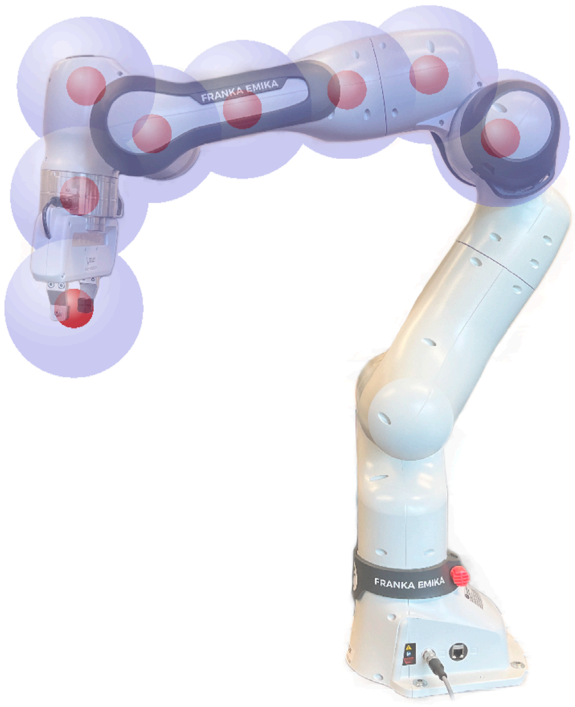

FIGURE 3
Visualization of possible robot control points including a spherical approximation of the robot.

in the form

$$w_3 = \begin{cases} 1 + \dfrac{\dot{\boldsymbol{x}} \cdot \boldsymbol{f}_{\text{vlc}}}{\|\dot{\boldsymbol{x}}\| \|\boldsymbol{f}_{\text{vlc}}\|} & \text{if } \dot{\boldsymbol{x}} \cdot \boldsymbol{f}_{\text{vlc}} < 0 \\ 1 & \text{otherwise.} \end{cases}$$

2.1.4 Robot body obstacle avoidance

To enable obstacle avoidance for the entire robot body, we define n_{cp} additional control points along its structure, as exemplarily shown for a 7-DoF robot arm in Figure 3. The control points should be placed on prominent points of the robot so that the whole structure can be moved away from obstacles. To ensure that the robot structure keeps a safe distance to obstacles, the distance to the control points is also calculated using a safety margin, resulting in a spherical approximation of the robot arm. Note that for the example in Figure 3, we have refrained from adding more control points on the lower links as their obstacle avoidance capabilities are limited. We assume that the robot needs to reach a Cartesian goal pose and that we do not have any information about possible final joint configurations. Consequently, we do not apply an attractive goal force on the control points and the force on a control point $k \in [1, n_{\text{cp}}]$ only consists of the respective CF-based obstacle avoidance forces

$${}^{k}\boldsymbol{f}_{\text{cp}} = {}^{k}\boldsymbol{f}_{\text{cf}} + {}^{k}\boldsymbol{f}_{\text{cf,rep}}.$$

2.1.5 Joint limit avoidance

To ensure safe operation of robotic manipulators, it is crucial to consider their physical constraints, specifically their joint limits.

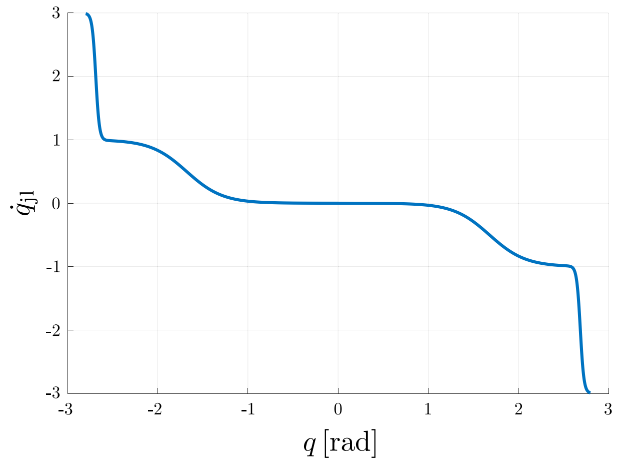

FIGURE 4
Visualization of the repulsive vector $\dot{q}_{\text{jl}}$ for a joint with the limits $q_{\text{max}} = 2.8$ rad and $q_{\text{min}} = -2.8$ rad using the settings $k_{\text{jc}} = 1.0, k_{\text{jl}} = 2.0, \gamma_{\text{sl,jc}} = 7.0, \gamma_{\text{d,jc}} = 0.4, \gamma_{\text{sl,jl}} = 70.0, \gamma_{\text{d,jl}} = 0.04$.

We use sigmoidal scaling functions as defined in Equation 2 to construct a repulsive vector that pushes the joints towards their center position. The repulsive vector consists of two parts. The first element pushes the joints towards their center with a comparatively small magnitude to keep the robot in configurations that allow a wide range of motions. The second component is activated when a joint is close to its limits and generates repulsive vectors with greater magnitude to move the robot away from its limits. Towards this end we first normalize each joint $p \in [1, n_{\text{dof}}]$ to the range ${}^{p}q_n \in [-1.0, 1.0]$ using

$${}^{p}q_n = -1 + 2\frac{{}^{p}q - {}^{p}q_{\text{min}}}{{}^{p}q_{\text{max}} - {}^{p}q_{\text{min}}}.$$

Here, ${}^{p}q_{\text{min}}$ and ${}^{p}q_{\text{max}}$ are the minimum and maximum position limits of a joint p. The repulsive vector is then calculated using each joint individually

$${}^{p}\dot{q}_{\text{jl}} = \begin{cases} k_{\text{jc}} \quad g_{\text{jc}}\left({}^{p}q_{\text{n}}\right) + k_{\text{jl}} \quad g_{\text{jl}}\left({}^{p}q_{\text{n}}\right) & \text{if } {}^{p}q_n < 0 \\ -k_{\text{jc}} \quad g_{\text{jc}}\left({}^{p}q_{\text{n}}\right) - k_{\text{jl}} \quad g_{\text{jl}}\left({}^{p}q_{\text{n}}\right) & \text{if } {}^{p}q_n > 0 \\ 0 & \text{otherwise,} \end{cases}$$

where $k_{\text{jc}}, k_{\text{jl}} \geq 0$ are scaling factors of the repulsive vector for the joint centering and the joint limit avoidance components, respectively and $g_{\text{jc}}({}^{p}q_{\text{n}}), g_{\text{jl}}({}^{p}q_{\text{n}})$ are sigmoidal functions as defined in Equation 2. An illustrative example for the magnitude of this repulsive vector for a single joint is shown in Figure 4.

2.1.6 Manipulability

The manipulability of a robotic manipulator is a useful criterion for quantifying the influence of joint movements on the end effector, thus providing a measure how easy or difficult it is to change the robots pose in its Cartesian operational workspace. A high manipulability indicates that only small movements of the joints are necessary to result in a large motion of the EE while the manipulability converges to zero when the robot is close to a singularity. Thus, it also provides information about the relative distance to singular configurations. The manipulability index, introduced in Yoshikawa (1985), is given by

$$\mu(\boldsymbol{q}) = \sqrt{\det\left(\boldsymbol{J}_{\text{ee}}(\boldsymbol{q})\boldsymbol{J}_{\text{ee}}(\boldsymbol{q})^{\top}\right)}.$$

The gradient of the manipulability can be used to calculate joint velocities that maximize the manipulability as presented in Dufour and Suleiman (2020)

$$\dot{\boldsymbol{q}}_{\text{m}} = k_{\text{m}}\left(\frac{\partial\mu(\boldsymbol{q})}{\partial\boldsymbol{q}}\right)^{\top},$$

where $k_{\text{m}} > 0$ is a constant gain.

2.1.7 Self-collision avoidance

To avoid self-collisions, we approximate the structure of the robot by spherical obstacles with a radius d_{sc} and use repulsive forces similar to Luo et al. (2014)

$$ {}_{l}^{k}\boldsymbol{f}_{\text{sc}} = \begin{cases} -k_{\text{sc}} \quad g_{\text{sc}}\left({}_{l}^{k}d\right) \quad \dfrac{{}_{l}^{k}\boldsymbol{d}_{\text{sc}}}{\|{}_{l}^{k}\boldsymbol{d}_{\text{sc}}\|} & \text{if} {}_{l}^{k}d < d_{\text{max,sc}} \\ 0 & \text{otherwise} \end{cases}, \tag{6}$$

where ${}_{l}^{k}\boldsymbol{d}_{\text{sc}} = {}_{l}\boldsymbol{x}_{\text{rs}} - {}^{k}\boldsymbol{x}_{\text{cp}}$ defines the distance vector between a control point k and a repulsive sphere l, $g_{\text{sc}}\left({}_{l}^{k}d\right)$ is a sigmoidal function as defined in Equation 2, and $k_{\text{sc}} > 0$ is a scaling factor. The force is applied only if the distance ${}_{l}^{k}d = \|{}_{l}^{k}\boldsymbol{d}_{\text{sc}}\| - d_{\text{sc}}$ is less than a threshold $d_{\text{max,sc}}$. Similar to the CF forces, the repulsive forces of all n_{sc} self-collision obstacles are superposed, leading to the following self-collision avoidance force on a control point $k \in [0, n_{\text{cp}}]$

$${}^{k}\boldsymbol{f}_{\text{sc}} = \sum_{l=0}^{n_{\text{sc}}} {}_{l}^{k}\boldsymbol{f}_{\text{sc}},$$

where the EE is considered as the control point with $k = 0$. When the self-collision avoidance force is used, the EE steering force and the forces on the control points are updated to

$$\begin{aligned} \boldsymbol{f}_{s_{\text{ee}}} &= \boldsymbol{f}_{\text{vlc}} + \boldsymbol{f}_{\text{cf}} + \boldsymbol{f}_{\text{cf,rep}} + {}^{0}\boldsymbol{f}_{\text{sc}}, \\ {}^{k}\boldsymbol{f}_{\text{cp}} &= {}^{k}\boldsymbol{f}_{\text{cf}} + {}^{k}\boldsymbol{f}_{\text{cf,rep}} + {}^{k}\boldsymbol{f}_{\text{sc}}. \end{aligned}$$

Note that in practical implementations, many of these repulsive forces may not be necessary or may even lead to infeasible or detrimental forces, e.g., if a control point is located on the same link as a repulsive sphere. Therefore special care has to be taken when defining the repulsive spheres on the robot structure. In our implementations, it was sufficient to place a single virtual repulsive sphere in the base of the robot which generates a force only on the EE.

2.1.8 Joint velocity damping

We introduce an additional joint velocity component that has a damping effect using

$$\dot{\boldsymbol{q}}_{\text{damp}} = -k_{\text{damp}}\dot{\boldsymbol{q}},$$

with the scaling factor $k_{\text{damp}} \geq 0$. Adding $\dot{\boldsymbol{q}}_{\text{damp}}$, which opposes the current velocity direction, to the total velocity control command resulting from the motion planning strategy, yields a component of the tracking controller output that has a damping effect and prevents undamped motions in the nullspace.

2.1.9 Robot control signal

In this section, we describe how we transform the steering forces, joint velocity commands and repulsive vectors into feasible reference signals for joint space control.

Consider the well-known equations of motion for a robotic manipulator

$$\boldsymbol{\tau}_{\text{ee}} + \boldsymbol{J}_{\text{ext}}^{\top}\boldsymbol{f}_{\text{ext}} = \boldsymbol{J}_{\text{ee}}^{\top}(\boldsymbol{q})\left(\boldsymbol{M}_{\text{c}}(\boldsymbol{q})\ddot{\boldsymbol{x}} + \boldsymbol{c}_{\text{c}}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q})\right), \tag{7}$$

where $\boldsymbol{M}_{\text{c}}(\boldsymbol{q})$ is the Cartesian inertia matrix, $\boldsymbol{c}_{\text{c}}(\boldsymbol{q},\dot{\boldsymbol{q}})$ describes the Coriolis and centrifugal terms, $\boldsymbol{g}(\boldsymbol{q})$ are the gravitational forces, $\boldsymbol{J}_{\text{ee}} \in \mathbb{R}^{6\times n_{\text{dof}}}$ is the Jacobian, describing the relation between the velocities of the n_{dof} joints and the EE velocity and $\boldsymbol{f}_{\text{ext}}$ are external forces acting on the robot, where $\boldsymbol{J}_{\text{ext}}$ is the Jacobian of the location where the external forces apply. This formulation enables direct torque control of the robotic manipulator and accommodates additional external forces commonly encountered during human-robot interaction, as discussed in Haddadin et al. (2010).

However, this paper specifically focuses on collision avoidance and we employ a separate joint space tracking controller for robot control. We assume that this joint space controller sufficiently compensates for gravity and dynamics. Consequently, we only need to transform the virtual task space steering forces generated by the proposed motion planning framework into desired joint reference signals. Furthermore, recall that we do not consider real forces; instead we interpret the artificial steering force as the desired EE acceleration, which allows us to ignore inertial forces. Thus, we use $\boldsymbol{f}_{s_{\text{ee}}} = \ddot{\boldsymbol{x}}$, set the inertia matrix in joint space to the identity matrix $\boldsymbol{M}(\boldsymbol{q}) = \boldsymbol{I}$ and disregard the gravitational and Coriolis terms in Equation 7. This implies $\ddot{\boldsymbol{q}}_{\text{ee}} = \boldsymbol{\tau}_{\text{ee}}$ and leads to the following definition of the Cartesian inertia matrix[2]

$$\begin{aligned} \boldsymbol{M}_{\text{c}}(\boldsymbol{q}) &= \left(\boldsymbol{J}_{\text{ee}}(\boldsymbol{q})\boldsymbol{M}(\boldsymbol{q})^{-1}\boldsymbol{J}_{\text{ee}}^{\top}(\boldsymbol{q})\right)^{-1} \\ &= \left(\boldsymbol{J}_{\text{ee}}(\boldsymbol{q})\boldsymbol{I}\boldsymbol{J}_{\text{ee}}^{\top}(\boldsymbol{q})\right)^{-1}. \end{aligned}$$

This approach simplifies Equation 7, allowing us to eliminate the inertia matrix from our calculations. This speeds up the planning process and avoids known issues associated with modeling inaccuracies of the inertia matrix (cf. Nakanishi et al., 2008).

Inserting $\boldsymbol{M}_{\text{c}}(\boldsymbol{q})$ into Equation 7 and neglecting the gravitation and dynamic components yields

$$\ddot{\boldsymbol{q}}_{\text{ee}} = \boldsymbol{J}_{\text{ee}}^{\#}(\boldsymbol{q})\boldsymbol{f}_{s_{\text{ee}}},$$

where $\boldsymbol{J}_{\text{ee}}^{\#} = \boldsymbol{J}_{\text{ee}}^{\top}\left(\boldsymbol{J}_{\text{ee}}\boldsymbol{J}_{\text{ee}}^{\top}\right)^{-1}$ is the Moore-Penrose pseudo inverse. Note that we additionally limit each joint individually to its maximum acceleration.

Ideally, the forces on the control points should be transformed into the nullspace of the main task, i.e., the control of the EE pose. However, our experiments indicate that consistently achieving full-body obstacle avoidance without interfering with the primary EE

2 Note that this simplification is only possible when the planner is used to calculate a reference trajectory for a tracking controller that considers the robot dynamics. The actual inertia matrix $\boldsymbol{M}(\boldsymbol{q})$ should be used when the resulting torque command $\boldsymbol{\tau}_{\text{ee}}$ is directly applied to the robot or when used in combination with any torque-based control interface.

task is challenging. Avoiding obstacles within the null space of the EE task is only feasible in few scenarios where simple obstacle avoidance movements are sufficient. Thus, the forces on the control points are considered as virtual external forces on the robot body. These forces are then transformed to joint accelerations using

$$^{k}\ddot{\boldsymbol{q}}_{\mathrm{cp}} = {}^{k}\boldsymbol{J}_{\mathrm{cp}}^{\top}(\boldsymbol{q}) \quad {}^{k}\boldsymbol{f}_{\mathrm{cp}},$$

where ${}^{k}\boldsymbol{J}_{\mathrm{cp}}$ is the Jacobian of the position of the kth control point. The total joint acceleration command is calculated by superimposing the desired accelerations from all control points

$$\ddot{\boldsymbol{q}}_{\mathrm{cmd}} = \ddot{\boldsymbol{q}}_{\mathrm{ee}} + \sum_{k=0}^{n_{\mathrm{cp}}} {}^{k}\ddot{\boldsymbol{q}}_{\mathrm{cp}}.$$

This approach ensures an indirect weighting of the joint accelerations based on the magnitude of the respective task space force. The influence of the control point acceleration on the total robot motion increases as the control point gets closer to an obstacle. Additionally, we employ a safety fallback strategy when the distance between a control point and an obstacle exceeds a lower threshold (cf. Section 2.2.4). Note that the simplification described above and the direct application of the forces on the control points affect the ability of the robot to follow the desired EE trajectory exactly. However, the proposed approach leads to efficient convergence to task space goal poses while avoiding obstacles, even in dynamic and complex settings, as demonstrated in Section 3.

We consider obstacle avoidance as the primary objective and the supplementary measures for joint centering, manipulability and damping as less important. Thus, these desired joint velocities and repulsive vectors are projected into the nullspace of the EE using the method described in (Siciliano and Khatib, 2008).

Finally, we calculate the reference signals for the next sampling step of the joint space controller. We integrate the acceleration control reference for generating appropriate inputs to a desired joint controller.

$$\dot{\boldsymbol{q}}_{\mathrm{cmd}} = \dot{\boldsymbol{q}} + \ddot{\boldsymbol{q}}_{\mathrm{cmd}} T_{\mathrm{c}} + \left(\boldsymbol{I} - \boldsymbol{J}_{\mathrm{ee}}^{\#}(\boldsymbol{q})\boldsymbol{J}_{\mathrm{ee}}^{\top}(\boldsymbol{q})\right)\left(\dot{\boldsymbol{q}}_{\mathrm{m}} + \dot{\boldsymbol{q}}_{\mathrm{jl}} + \dot{\boldsymbol{q}}_{\mathrm{damp}}\right), \quad (8)$$

$$\boldsymbol{q}_{\mathrm{cmd}} = \boldsymbol{q} + \dot{\boldsymbol{q}}_{\mathrm{cmd}} T_{\mathrm{c}},$$

where T_{c} is the control step time. Similarly to the joint acceleration, each joint is limited individually to its maximum velocities and angle. In our applications, we use a joint impedance controller, which considers joint velocity and joint angle reference signals as inputs.

2.2 Unifying global planning and reactive control

Although our reactive CF-based control component performs well in a local scope of the environment, it faces inherent limitations typically associated with locally reactive controllers. These limitations have been extensively discussed in the related work (cf. Section 1.2) and primarily stem from the omission of global environmental information and constrained exploration capabilities. Traditional CF approaches, which function solely as local planners, perform suboptimally when attempting to find global solutions, resulting in trajectories that are generally less efficient. Moreover, despite the inherent absence of local minima in CFs, it is still possible to design trap scenarios that restrict the robot's movement. Note that this is not equivalent to the local minima encountered in APF approaches, as the robot does not come to a standstill but instead becomes trapped in limit cycles.

In (Becker et al., 2021) we developed the CFP planner, which uses a predictive virtual agent framework to efficiently explore the global environment by simulating the robot using different settings for the magnetic field vector. However, in contrast to our previous work, the extension to robotic manipulators using additional control points on the robot leads to a significant increase of virtual agents, especially in the case of many obstacles. Furthermore, the framework is not able to determine which avoidance directions around obstacles of the different control points are not compatible with each other before simulating the whole robot motion. Thus, many combinations of magnetic field vectors for the different control points will lead to infeasible trajectories or even to collisions. Consequently, a considerable amount of computing power is wasted by simulating these magnetic field vector sets.

To compensate for these disadvantages, we introduce the concept of ICF, which extends the capabilities of the local CF approach by incorporating and extracting global environmental information. The method is not intended to operate as a standalone global planner; instead, it is tightly integrated into the CF planning component. The integration aims to unify global planning and reactive control, bridging the gap between global environment exploration and reactive collision avoidance. The heavily parallelized design enables instantaneous responses to dynamic obstacle motion, and unpredictable changes in partially unknown environments, even when the global component faces challenges in finding a solution.

The general idea of ICF is to leverage the strengths of global and local motion planning strategies by extracting useful information from a global pre-planner. This information is then transferred to and evaluated by an adapted virtual agents framework to improve reactive motion generation. Throughout the development, a significant emphasis has been placed on maintaining the reactivity of the CF planner, even when the global planning component fails to find a solution.

The process of generating a control reference signal within our ICF framework is divided into four distinct phases.

1. Initially, we employ a global configuration space motion planner to create a (coarse) joint trajectory (cf. Section 2.2.1).
2. Following this, we extract global information from this coarse trajectory in the form of magnetic field vectors for all control points and obstacles, which serves as a basis for reactive CF-based motion planning (cf. Section 2.2.2). It is important to note that when referring to control points, the EE is included unless otherwise specified.
3. Subsequently, multiple predictive agents are created using the extracted magnetic field vectors to process and evaluate the global motion plan (cf. Section 2.2.3).
4. The parameters of the best agent are then transferred to the real robot and used to generate the reactive joint space commands (as described in Section 2.1.9).

These different phases are executed in parallel, each with its respective sampling rates.

2.2.1 Global trajectory generation

The main purpose of the global pre-planner is to infer estimates for feasible global trajectories from the current robot pose to a Cartesian goal pose in the configuration space. In particular, finding non-conflicting avoidance directions around the obstacles for all control points is more important than generating short and smooth paths. Consequently, we use a rather coarse discretization of the global planner for shorter planning cycles T_{global}. The global planner is continuously running and restarted after T_{global} seconds with updated obstacle and robot information. When a successful trajectory is found during the planning time, the trajectory is passed to the global information extraction module.

Due to frequently proven successful results, short planning times, and wide availability, we use sampling-based planners for the global trajectory generation. In particular, we use the MoveIt! motion planning framework because it features a variety of sampling-based planners and supports point clouds as a format for obstacle representation (Coleman et al., 2014). However, other configuration space planners can also be used and exchanged easily. Note that the global motion planning is done in a static snapshot of the environment while the actual reactive creation of the control reference (cf. Section 2.1.9) is always done with the most current environment information.

2.2.2 Global information extraction

Whenever the global pre-planner finds a successful joint trajectory, the global information extraction module is triggered. The joint trajectory is then transformed into separate Cartesian trajectories for each control point using forward kinematics

$$^{k}\boldsymbol{x}_{\text{cp}}(t) = {}^{k}\boldsymbol{T}_{\text{cp}}(\boldsymbol{q}(t))\begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix}^{\top},$$

where $^{k}\boldsymbol{T}_{\text{cp}}$ is the Denavit-Hartenberg (DH) matrix that defines the transformation from the base frame to the control point k. The avoidance direction of each control point around each obstacle is then extracted from the Cartesian trajectories and the corresponding magnetic field vectors are calculated using both of the following algorithms. Visualizations of both algorithms are shown in Figures 5, 6 in 2D example setups.For each control point k and for each obstacle j described by the points $^{j}_{i}\boldsymbol{x}_{\text{o}} \in {}^{j}\mathbb{O} \subset \mathbb{R}^{3\times m_j}$, the following calculations are performed.

The quality of the resulting avoidance motion from both methods highly depends on the environment and the global trajectory. The first algorithm captures the avoidance motion only at the closest obstacle point. Therefore, it only accurately represents the avoidance movement if the obstacle is avoided through a uniform motion without any directional changes. In contrast, the second algorithm defines a surrounding area around the obstacle to recreate a resulting homogeneous avoidance direction around that region. As a result, any additional movements or changes in direction within this area are neglected, as they do not contribute to the avoidance maneuver. However, in some cases, it may be necessary to perform additional motions close to an obstacle. The relevance of these motions always depends on the specific scenario, e.g., additional motions may be necessary to reconfigure the robot joints and reach the goal without hitting the joint limits. Therefore, we use both methods to calculate magnetic field vectors, which are adopted by the predictive agents as described in the next section. Nevertheless,

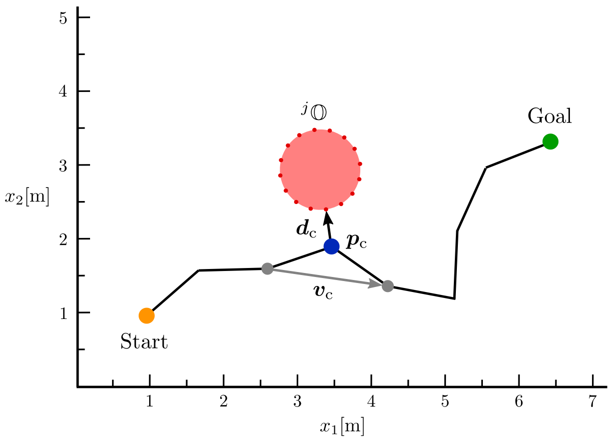
5
4
3
2
1
x_2[m]
$^{j}\mathbb{O}$
Goal
$\boldsymbol{d}_{\text{c}}$
$\boldsymbol{p}_{\text{c}}$
$\boldsymbol{v}_{\text{c}}$
Start
1
2
3
4
5
6
7
x_1[m]

FIGURE 5
Visualization of Algorithm 1 in a simplified 2D representation of the environment.

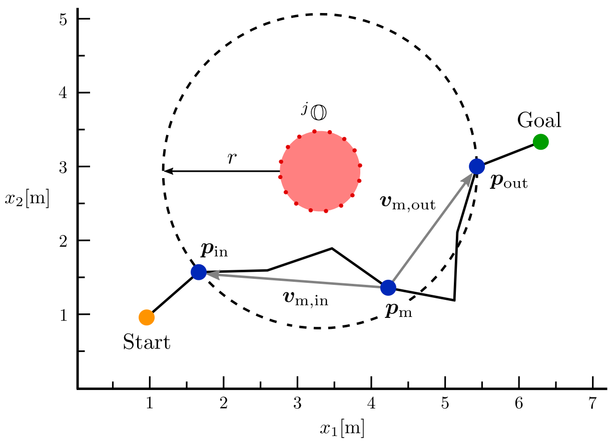
5
4
3
2
1
x_2[m]
$^{j}\mathbb{O}$
Goal
r
$\boldsymbol{p}_{\text{out}}$
$\boldsymbol{v}_{\text{m,out}}$
$\boldsymbol{p}_{\text{in}}$
$\boldsymbol{v}_{\text{m,in}}$
$\boldsymbol{p}_{\text{m}}$
Start
1
2
3
4
5
6
7
x_1[m]

FIGURE 6
Visualization of Algorithm 2 in a simplified 2D representation of the environment.

we simplify a potentially complex avoidance motion to a single vector per obstacle and control point. Thus, the resulting motion of the ICF planner is expected to deviate from the global trajectory. However, we would like to emphasize again that we only use the global planner to estimate feasible, non-contradictory avoidance directions, which are subsequently simulated and evaluated before being used to enable reactive control of the real robot.

2.2.3 Adapting the virtual agent framework

After transforming the trajectories of the global pre-planners into magnetic field vector sets, we utilize the virtual agent framework described in Becker et al. (2021) to process this global information. The framework enables us to identify the influence of selected control parameters, such as the magnetic field vectors, on the robot in a current perception of the environment under simplified dynamics. Within the framework, the robot is represented in this environment snapshot by virtual predictive agents, each with a

```
1. Find the closest point $\boldsymbol{p}_{\mathrm{c}}$ of the trajectory
   ${}^k\mathbb{X}_{\mathrm{cp}} = \left\{ {}^k\boldsymbol{x}_{\mathrm{cp}}(t) \mid t \in [0, T_{\mathrm{global}}] \right\}$ to the obstacle $j$

   $\boldsymbol{p}_{\mathrm{c}} = {}^k\boldsymbol{x}_{\mathrm{cp}}(\tau)$  with
   $\tau = \underset{t \in [0, T_{\mathrm{global}}]}{\mathrm{argmin}} \left( \underset{{}^j_i\boldsymbol{x}_{\mathrm{o}} \in {}^j\mathbb{O}}{\min} \| {}^k\boldsymbol{x}_{\mathrm{cp}}(t) - {}^j_i\boldsymbol{x}_{\mathrm{o}} \| \right).$

2. Calculate the (approximate) direction of
   movement $\boldsymbol{v}_{\mathrm{c}}$ at the closest point $\boldsymbol{p}_{\mathrm{c}}$

   $\boldsymbol{v}_{\mathrm{c}} = {}^k\boldsymbol{x}_{\mathrm{cp}}(\tau + 1) - {}^k\boldsymbol{x}_{\mathrm{cp}}(\tau - 1).$

3. Calculate the vector $\boldsymbol{d}_{\mathrm{c}}$ pointing from $\boldsymbol{p}_{\mathrm{c}}$ to the
   closest obstacle point ${}^j_c\boldsymbol{x}_{\mathrm{o}}$

   $\boldsymbol{d}_{\mathrm{c}} = {}^j_c\boldsymbol{x}_{\mathrm{o}} - \boldsymbol{p}_{\mathrm{c}}$  with
   ${}^j_c\boldsymbol{x}_{\mathrm{o}} = \underset{{}^j_i\boldsymbol{x}_{\mathrm{o}} \in {}^j\mathbb{O}}{argmin} \left( \| {}^j_i\boldsymbol{x}_{\mathrm{o}} - \boldsymbol{p}_{\mathrm{c}} \| \right).$

4. Define the magnetic field vector ${}^j\boldsymbol{b}$ as

   ${}^j\boldsymbol{b} = \dfrac{\boldsymbol{d}_{\mathrm{c}} \times \boldsymbol{v}_{\mathrm{c}}}{\| \boldsymbol{d}_{\mathrm{c}} \times \boldsymbol{v}_{\mathrm{c}} \|}.$
```

Algorithm 1. First option for extracting the avoidance direction around an obstacle from a pre-planned path. This algorithm captures the avoidance motion at the closest obstacle point and is particularly effective when the obstacle is being avoided in a uniform motion without directional changes.

```
1. Find the first point $\boldsymbol{p}_{\mathrm{in}}$ of the trajectory ${}^k\boldsymbol{x}_{\mathrm{cp}}$
   in a ball of a predefined radius $r$ around the
   obstacle $j$

   $\boldsymbol{p}_{\mathrm{in}} = {}^k\boldsymbol{x}_{\mathrm{cp}}(\tau_{\mathrm{min}})$  with
   $\tau_{\mathrm{min}} = \underset{t \in [0, T_{\mathrm{global}}], {}^j_i\boldsymbol{x}_{\mathrm{o}} \in {}^j\mathbb{O}}{\min} \left( t \mid \| {}^k\boldsymbol{x}_{\mathrm{cp}}(t) - {}^j_i\boldsymbol{x}_{\mathrm{o}} \| \le r \right).$

2. Find the last point $\boldsymbol{p}_{\mathrm{out}}$ of the trajectory ${}^k\boldsymbol{x}_{\mathrm{cp}}$
   in a ball of a predefined radius $r$ around the
   obstacle $j$:

   $\boldsymbol{p}_{\mathrm{out}} = {}^k\boldsymbol{x}_{\mathrm{cp}}(\tau_{\mathrm{max}})$  with
   $\tau_{\mathrm{max}} = \underset{t \in [0, T_{\mathrm{global}}]}{\max} \; \underset{{}^j_i\boldsymbol{x}_{\mathrm{o}} \in {}^j\mathbb{O}}{\min} \left( t \mid \| {}^k\boldsymbol{x}_{\mathrm{cp}}(t) - {}^j_i\boldsymbol{x}_{\mathrm{o}} \| \le r \right).$

3. Find the point $\boldsymbol{p}_{\mathrm{m}} = {}^k\boldsymbol{x}_{\mathrm{cp}}(\tau)$ with $\tau$ being the
   largest integer less than or equal to the midpoint
   in $[\tau_{\mathrm{min}}, \tau_{\mathrm{max}}]$, i.e.,

   $\tau = \lfloor 0.5(\tau_{\mathrm{max}} + \tau_{\mathrm{min}}) \rfloor.$

4. Calculate the vectors $\boldsymbol{v}_{\mathrm{m,in}}$ pointing from $\boldsymbol{p}_{\mathrm{m}}$ to
   $\boldsymbol{p}_{\mathrm{in}}$ and $\boldsymbol{v}_{\mathrm{m,out}}$ from $\boldsymbol{p}_{\mathrm{m}}$ to $\boldsymbol{p}_{\mathrm{out}}$

   $\boldsymbol{v}_{\mathrm{m,in}} = \boldsymbol{p}_{\mathrm{in}} - \boldsymbol{p}_{\mathrm{m}},$
   $\boldsymbol{v}_{\mathrm{m,out}} = \boldsymbol{p}_{\mathrm{out}} - \boldsymbol{p}_{\mathrm{m}}$

5. ${}^j\boldsymbol{b}$ is defined perpendicular to the plane
   through the three points $\boldsymbol{p}_{\mathrm{in}}$, $\boldsymbol{p}_{\mathrm{out}}$ and $\boldsymbol{p}_{\mathrm{m}}$

   ${}^j\boldsymbol{b} = \dfrac{\boldsymbol{v}_{\mathrm{m,in}} \times \boldsymbol{v}_{\mathrm{m,out}}}{\| \boldsymbol{v}_{\mathrm{m,in}} \times \boldsymbol{v}_{\mathrm{m,out}} \|}.$
```

Algorithm 2. Second option for extracting the avoidance direction around an obstacle from a pre-planned path. This algorithm generalises the avoidance motion in an environment around the obstacle in order to better map the resulting total evasive direction and to ignore additional movements and reconfigurations that do not contribute to the avoidance direction.

specific set of parameters $\mathbb{P}$. In order to adequately account for the new information from the global pre-planner, the process for creating and deleting predictive agents from Becker et al. (2021) is redesigned as described in the following.

2.2.3.1 Virtual agent creation

Whenever the global pre-planner and information extraction module generate new magnetic field vectors, new virtual agents using these magnetic field vectors are created. Moreover, if the maximum number of predictive agents $n_{a,\mathrm{max}}$ has not been reached, additional agents with different parameter sets $\mathbb{P}$ are created. The creation of new predictive agents follows a specific order upon receiving new magnetic field vectors.

1. Create one agent with the current best parameter set $\mathbb{P}_{\mathrm{best}}$ and the current best magnetic field vectors $\mathbb{B}_{\mathrm{best}}$.
2. Create one agent with the current best parameter set $\mathbb{P}_{\mathrm{best}}$ but with the new magnetic field vector set $\mathbb{B}_{\mathrm{A1}}$ from Algorithm 1.
3. Create one agent with the current best parameter set $\mathbb{P}_{\mathrm{best}}$ but with the new magnetic field vector set $\mathbb{B}_{\mathrm{A2}}$ from Algorithm 2.
4. Create additional agents with a parameter set that differs in an arbitrary parameter m_1 from $\mathbb{P}_{\mathrm{best}}$ if the number of agents is less than $n_{a,\mathrm{max}}$.
5. Create additional agents with a parameter set that differs in parameter m_1 from $\mathbb{P}_{\mathrm{best}}$ and uses $\mathbb{B}_{\mathrm{A1}}$ if the number of agents is less than $n_{a,\mathrm{max}}$.
6. Create additional agents with a parameter set that differs in parameter m_1 from $\mathbb{P}_{\mathrm{best}}$ and uses $\mathbb{B}_{\mathrm{A2}}$ if the number of agents is less than $n_{a,\mathrm{max}}$.
7. Repeat steps 4-6 with other parameters m_p or with another feasible option for the parameter m_1 until a total of $n_{a,\mathrm{max}}$ agents have been created.

In this paper, we use the virtual agent framework primarily for the magnetic field vector $\boldsymbol{b} \in \mathbb{B} \subset \mathbb{R}^{n_{\mathrm{cp}} \times n_{\mathrm{o}}}$ of the n_{cp} control points (including the EE). However, as described previously other parameter choices can also be simulated. In our approach, the supplementary forces described in Sections 2.1.5, 2.1.6 and 2.1.8 are defined as optional parameters for calculating the final control command [cf. Equation 8]. Consequently, we add binary activation flags for all supplementary forces to the parameter vector $\mathbb{P}$. This enables the deactivation of supplementary forces as needed, allowing for only necessary calculations to be performed at any given time. For instance, we define m_1 to be the activation flag of the manipulability gradient calculation method from Section 2.1.6. As a result, instead of including the manipulability velocity vector in the reference command calculation Equation 8 in every simulation, we simulate both a set of agents that applies the manipulability gradient to the resulting velocity command and a set of agents without it. This procedure ensures that the command for the real robot is calculated using the optimal set of forces and prevents situations, where, e.g., the manipulability gradient might interfere with the force on a control point that is close to an obstacle.

2.2.3.2 Virtual agent simulation

The virtual agents are then simulated using the control commands from Section 2.1.9, assuming simplified dynamics and that the controller follows the joint commands perfectly, i.e.,

$$\begin{aligned} \boldsymbol{q}(t+1) &= \boldsymbol{q}(t) + \dot{\boldsymbol{q}}_{\mathrm{cmd}}(t)\Delta T \\ \dot{\boldsymbol{q}}(t+1) &= \dot{\boldsymbol{q}}_{\mathrm{cmd}}(t) \\ \dot{\boldsymbol{x}}_{\mathrm{ee}}(t+1) &= \boldsymbol{J}_{\mathrm{ee}}(\boldsymbol{q}(t+1))\,\dot{\boldsymbol{q}}(t+1). \end{aligned}$$

This procedure is repeated until the simulated agent reaches the goal pose $\boldsymbol{x} = \boldsymbol{x}_{\mathrm{g}}$. However, virtual agents are only simulated for a defined maximum number of prediction steps n_{ps} at a time, after which the framework proceeds with the simulation of a different agent. It is essential to note that parallel computation of multiple agents remains possible and is actively exploited. This method was introduced to ensure that all agents are treated with the same priority and progress through the simulation at a similar rate. In practical implementation on conventional computing devices, it is otherwise difficult to ensure a fair distribution of computing power over several threads. This is particularly important when more agents were created than the computation device can handle concurrently. Specifically, this precautionary measure prevents agents, which might never reach the goal or follow long suboptimal trajectories (e.g., due to poor choices of magnetic field vectors), from indefinitely blocking the available computation slots. During the simulation it is possible to also simulate the motion of dynamic obstacles, for which we use a constant velocity model in our simulations.

2.2.3.3 Virtual agent evaluation

In contrast to Becker et al. (2021), the evaluation of predictive agents is performed asynchronously instead of evaluating all agents simultaneously at a fixed time interval. Each agent is evaluated and compared to the current best agent immediately after its simulation, which is interrupted either after n_{ps} steps or upon reaching the goal. If an agent has a higher reward, it is validated whether the position of the predicted trajectory of the new parameters for the latest time step approximately matches the latest real agent position. Discrepancies may arise when the robot took a different avoidance direction around an obstacle, dynamic obstacles influenced the robot trajectory, or due to discretization errors. In order to mitigate the influence of discretization errors, the threshold for detecting deviations should be adjusted. Nonetheless, with time, the discretization error is expected to surpass the threshold, indicating a potentially outdated predicted trajectory. In all cases, the respective agent is deleted and the parameter set is not updated. Otherwise, the real robot uses the new best parameter set.

We use the following criteria, prioritized by the reward gains $\varrho_{\mathrm{g}} \geq \varrho_{\mathrm{d}} \geq \varrho_{\mathrm{tl}} \geq \varrho_{\mathrm{jl}} \geq \varrho_{\mathrm{s}} \geq \varrho_{\mathrm{o}} \geq \varrho_{\mathrm{mfv}} \geq 0$ for evaluating a virtual agent.

1. Reaching the goal with the EE without colliding is typically a complex task, and thus an agent p that approaches a distance d_{gd} around the goal gets a high reward ϱ_{g}. Otherwise, only a smaller reward ϱ_{d} is granted that decreases exponentially with a higher remaining distance to the goal

$$^{p}r_{\mathrm{gd}} = \begin{cases} \varrho_{\mathrm{g}} & \text{if} \left\| \boldsymbol{x}_{\mathrm{g}} - {}^{p}\boldsymbol{x}_{\mathrm{ee}}\left({}^{p}N_{\mathrm{ps}}\right) \right\| \leq d_{\mathrm{gd}}, \\ \varrho_{\mathrm{d}} e^{-\frac{\left\| \boldsymbol{x}_{\mathrm{g}} - {}^{p}\boldsymbol{x}_{\mathrm{ee}}\left({}^{p}N_{\mathrm{ps}}\right) \right\|}{\gamma_{\mathrm{gd}}}} & \text{otherwise,} \end{cases}$$

where $\gamma_{\mathrm{gd}} > 0$ is a constant scaling factor and ${}^{p}N_{\mathrm{ps}}$ is the total number of steps that agent p has been predicted. Appropriate scaling of ϱ_{g} ensures that an agent that reached the goal without collision will always receive a higher total reward than an agent who has not yet done so.

2. The distance covered by the EE of a manipulator is often a less meaningful evaluation criterion than the path length traveled by a mobile robot. Instead, we use the duration of the current prediction as a criterion for rewarding shorter motions

$$^{p}r_{\mathrm{tl}} = \varrho_{\mathrm{tl}} - {}^{p}N_{\mathrm{ps}} T_{\mathrm{s}}.$$

3. We favour trajectories that result in a lower percentage of joint limit avoidance forces

$$^{p}r_{\mathrm{jl}} = \varrho_{\mathrm{jl}} \left(1 - \sum_{n=0}^{{}^{p}N_{\mathrm{ps}}-1} \frac{\left\| {}^{p}\ddot{\boldsymbol{q}}_{\mathrm{jl}}(n) \right\|}{\left\| {}^{p}\ddot{\boldsymbol{q}}_{\mathrm{ee}}(n) + {}^{p}\ddot{\boldsymbol{q}}_{\mathrm{cp}}(n) + {}^{p}\ddot{\boldsymbol{q}}_{\mathrm{jl}}(n) \right\|} \right).$$

4. We also use the manipulability index to avoid singularities by rewarding a higher minimum manipulability of the agent trajectory

$$^{p}r_{\mathrm{s}} = \varrho_{\mathrm{s}} \min_{n \in \left[0, {}^{p}N_{\mathrm{ps}}\right]} \mu\left({}^{p}\boldsymbol{q}(n)\right).$$

5. We reward agents, which pass obstacles with greater clearance by including the minimal distance of all control points and the EE to all obstacles defined as

$$^{p}d_{\min} = \min_{n \in \left[0, N_{\mathrm{ps}}\right], k \in \left[0, n_{\mathrm{cp}}\right], {}_{i}^{j}\boldsymbol{x}_{\mathrm{o}}(n) \in {}^{j}\mathbb{O}(n), j \in \left[1, n_{\mathrm{o}}\right]} \left(\left\| {}_{p}^{k}\boldsymbol{x}_{\mathrm{cp}}(n) - {}_{i}^{j}\boldsymbol{x}_{\mathrm{o}}(n) \right\| \right),$$

where $k = 0$ again denotes the EE. The resulting reward is then given by

$$^{p}r_{\mathrm{o}} = \varrho_{\mathrm{o}} \left(1 - e^{d_{\min} - {}^{p}d_{\min}} \right).$$

6. To prevent abrupt changes in force direction or oscillations of the robot, we aim to avoid constant switches between agents with similar scores. This is achieved by rewarding agents that have similar avoidance directions as the current best agent. We compare the magnetic field vectors of all control points and obstacles to determine the similarity

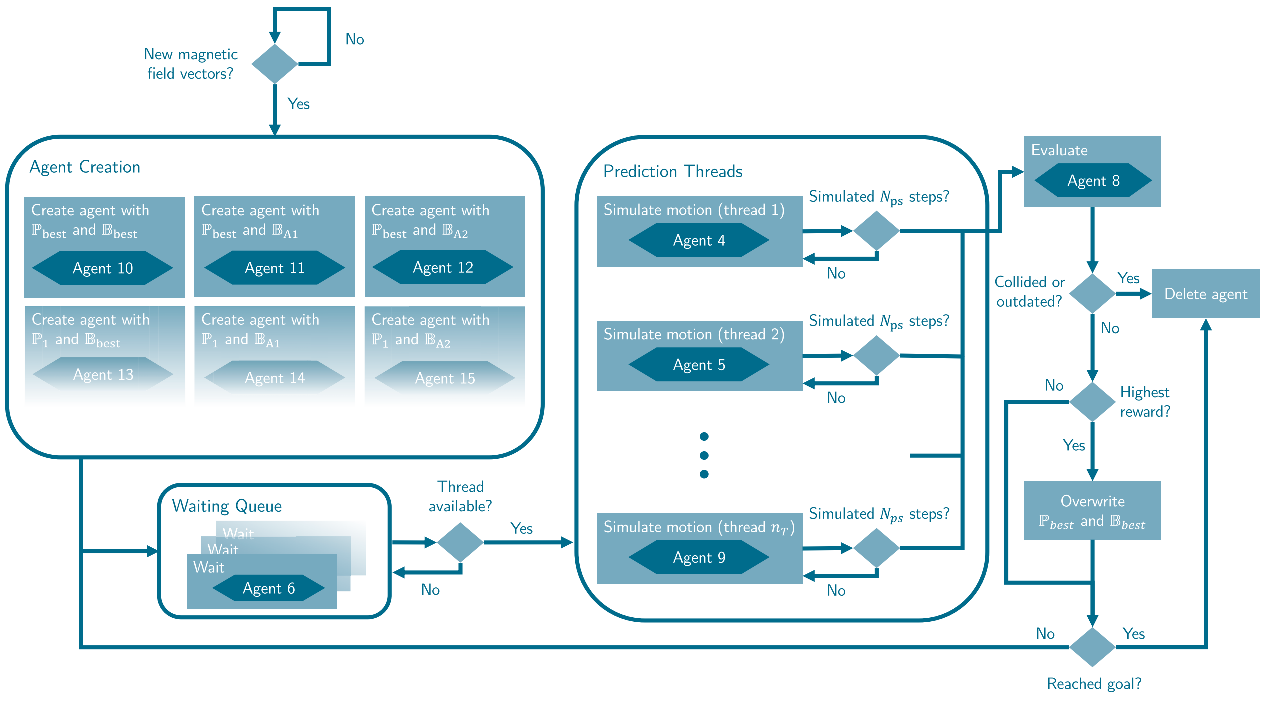
New magnetic field vectors?
No
Yes
Agent Creation
Create agent with $\mathbb{P}_{\text{best}}$ and $\mathbb{B}_{\text{best}}$
Agent 10
Create agent with $\mathbb{P}_{\text{best}}$ and $\mathbb{B}_{\text{A1}}$
Agent 11
Create agent with $\mathbb{P}_{\text{best}}$ and $\mathbb{B}_{\text{A2}}$
Agent 12
Create agent with $\mathbb{P}_1$ and $\mathbb{B}_{\text{best}}$
Agent 13
Create agent with $\mathbb{P}_1$ and $\mathbb{B}_{\text{A1}}$
Agent 14
Create agent with $\mathbb{P}_1$ and $\mathbb{B}_{\text{A2}}$
Agent 15
Prediction Threads
Simulate motion (thread 1)
Agent 4
Simulated N_{ps} steps?
Simulate motion (thread 2)
Agent 5
Simulate motion (thread n_T)
Agent 9
Evaluate
Agent 8
Collided or outdated?
Delete agent
Highest reward?
Overwrite $\mathbb{P}_{best}$ and $\mathbb{B}_{best}$
Reached goal?
Waiting Queue
Wait
Agent 6
Thread available?

FIGURE 7
Activity diagram of adapted virtual agent framework from ICF approach.

$$
{}^{P}r_{\text{mfv}} = \varrho_{\text{mfv}} \left(1 - \frac{1}{n_{\text{o}} n_{\text{cp}}} \sum_{j=0}^{n_{\text{o}}} \sum_{k=0}^{n_{\text{cp}}} \left(\| {}^{j}_{k}\boldsymbol{b}_{\text{best}} - {}^{j}_{k}\boldsymbol{b} \| \right) \right).
$$

The total reward is defined as the sum of all individual rewards

$$
{}^{P}r_{\text{t}} = {}^{P}r_{\text{gd}} + {}^{P}r_{\text{tl}} + {}^{P}r_{\text{jl}} + {}^{P}r_{\text{s}} + {}^{P}r_{\text{o}} + {}^{P}r_{\text{mfv}}.
$$

It is important to note that we do not pass the calculated trajectory to the real robot. Instead, the real robot adapts the parameters of the best agent and calculates its control command in parallel to the simulation of the virtual agents and to the planning of the global pre-planner. This procedure ensures reactive behavior of the overall planning framework and also allows that the simulation of the virtual agents are performed with a coarser step time to speed up the global computation process.

2.2.3.4 Virtual agent validation

In environments with dynamic obstacles, it is natural for the actual system behavior to differ over time from the prediction due to the asynchronous simulation and the different time step discretization. Our approach involves regular validations to ensure that the predicted trajectory of the current best agent remains consistent with the actual robot trajectory and avoids collisions with obstacles, for instance, from motions of dynamic obstacles. If deviations occur, the predicted trajectory is invalidated, and the best agent reward is reset to zero. After validation, a new agent is created with the current robot parameter set and state. This ensures continuous simulation of the best agent and maintains comparability of its reward with new agents. The planner also initiates re-planning when all agents have either reached the goal or have been deleted. This process also involves creating a predictive agent at the current robot state using the best parameter set. Setting the reward to zero may result in frequent changes of the best agent and could lead to undesired behavior, such as oscillations. To mitigate this, the comparison of rewards can be delayed until a minimum number of agents $n_{\text{a,min}}$ were evaluated. During this transition period, the real robot continues moving using the previous best parameters and using the safety-improving fallback, which is described in the following section. The whole process is depicted in Figure 7.

2.2.4 Safety-improving fallback

During motion planning, it may happen that none of the predicted trajectories match the current state of the environment, either because of unpredictable movements of the obstacles or because the robot cannot follow the predicted trajectories with sufficient accuracy. Additionally, this could lead to close distances between a control point and obstacles. In both cases, collision avoidance has absolute priority and the robot switches to a safety-improving fallback behavior. Here, the CF forces on the control points on the robot structure are replaced by repulsive forces that are defined similarly to the self-collision avoidance force in Equation 6. The force scaling from the sigmoidal function (cf. Equation 2) ensures that only obstacle points within a limited range around the robot generate a force. The maximum range can be adjusted by the factor γ_{d}, while γ_{sl} determines the rate at which its level of influence increases. The total force from all obstacle points on a control point is defined by

$$
{}^{k}\boldsymbol{f}_{\text{rpf}} = \sum_{j=0}^{n_{\text{o}}} \sum_{i=0}^{m_j} {}^{kj}_{i}\boldsymbol{f}_{\text{rpf}}.
$$

TABLE 1 Pre-planner comparison.

Pre-planner	Path length [%]			Success [%]	Path duration [%]		
	Mean	Max	Min		Mean	Max	Min
RRT LaValle (1998)	6.13	17.84	3.67	96.5	5.51	47.70	4.86
RRTConnect Kuffner and LaValle (2000)	4.42	19.52	1.93	98.0	4.33	23.09	5.06
RRT* Karaman and Frazzoli (2011)	4.28	**8.54**	3.66	96.5	**3.37**	**9.47**	6.67
TRRT Jaillet et al. (2010)	3.20	14.20	2.23	98.5	6.53	52.99	6.57
BiTRRT Devaurs et al. (2013)	4.93	22.44	3.44	96.5	7.90	54.90	8.44
LBT-RRT Salzman and Halperin (2016)	6.79	23.00	3.34	97.5	13.05	73.88	4.86
SBL Sánchez and Latombe (2003)	4.15	11.11	1.97	96.0	4.42	30.65	5.84
EST Hsu et al. (1997)	**2.20**	10.80	2.82	98.0	3.87	25.61	6.66
BiEST Hsu et al. (1997)	4.13	13.99	2.28	96.5	3.39	27.78	5.53
ProjEST Hsu et al. (1997)	4.14	11.81	1.55	98.0	4.70	32.28	5.60
KPIECE Şucan and Kavraki (2009)	3.18	10.81	2.23	97.5	5.40	44.85	7.13
BKPIECE Şucan and Kavraki (2009)	5.50	27.99	3.73	96.0	4.88	26.26	5.95
LBKPIECE Şucan and Kavraki (2009); Bohlin and Kavraki (2000)	3.76	10.72	3.22	96.5	6.05	53.49	7.37
PDST Ladd and Kavraki (2005)	4.61	18.92	2.50	97.5	5.82	46.03	6.77
STRIDE Gipson et al. (2013)	5.77	18.23	**0.91**	97.0	3.79	14.01	5.88
SPARS Dobson et al. (2013)	5.37	15.64	3.39	98.5	4.37	21.62	6.84
SPARS2 Dobson and Bekris (2013)	5.71	21.42	3.82	98.5	7.17	55.44	6.16
PRM Kavraki et al. (1996)	3.90	13.78	1.90	**100.0**	3.86	37.36	**3.94**
PRM* Karaman and Frazzoli (2011)	4.60	20.82	2.82	98.0	9.96	93.10	8.08
LazyPRM* Bohlin and Kavraki (2000); Karaman and Frazzoli (2011)	4.63	16.80	2.90	98.0	4.83	35.62	6.33

1. Bold values highlight the best results across all methods. 2. Red values indicate at least one occurrence of a collision during the respective run.

Moreover, we change the nullspace of the supplementary forces, i. e., the manipulability gradient, the joint limit avoidance and the damping velocity, whenever a control point is close to an obstacle. Instead of projecting the forces in the nullspace of the EE pose, the forces are projected into the augmented nullspace of the EE translation and the translation of the closest control point. Accordingly, the steering forces and the joint velocity control reference are calculated with.

$$\boldsymbol{f}_{s_{\text{ee}}} = \boldsymbol{f}_{\text{vlc}} + {}^{0}\boldsymbol{f}_{\text{rpf}},$$

$${}^{k}\boldsymbol{f}_{\text{cp}} = {}^{k}\boldsymbol{f}_{\text{rpf}}$$

$$\dot{\boldsymbol{q}}_{\text{cmd}} = \dot{\boldsymbol{q}} + \ddot{\boldsymbol{q}}_{\text{cmd}} T_{\text{c}} + \left(\boldsymbol{I} - \boldsymbol{J}^{\#}_{\text{safe}}(\boldsymbol{q})\boldsymbol{J}^{\top}_{\text{safe}}(\boldsymbol{q})\right)\left(\dot{\boldsymbol{q}}_{\text{m}} + \dot{\boldsymbol{q}}_{\text{jl}} + \dot{\boldsymbol{q}}_{\text{damp}}\right),$$

where $\boldsymbol{J}^{\top}_{\text{safe}} = \left[\boldsymbol{J}^{\top}_{\text{ee,t}}\ {}^{c}\boldsymbol{J}_{\text{cp,t}}{}^{\top}\right]$ is defined by the Jacobian of the EE position $\boldsymbol{J}_{\text{ee,t}} \in \mathbb{R}^{n_{\text{dof}}\times 3}$ and the Jacobian of the position of the control point that has the minimum distance to the obstacles ${}^{c}\boldsymbol{J}_{\text{cp,t}} \in \mathbb{R}^{n_{\text{dof}}\times 3}$. Note that this definition of the repulsive force inherits the same disadvantages as APF, notably the introduction of local minima. However, we want to highlight again, that this is a rare occurrence where collision avoidance takes priority over goal convergence. Furthermore, it should be noted that the safety improvement fallback mechanism is disabled once an agent discovers a feasible path to the goal pose.

3 Simulations and experiments

In this section, we evaluate the performance of the ICF planner across various static and dynamic environments. Given that the

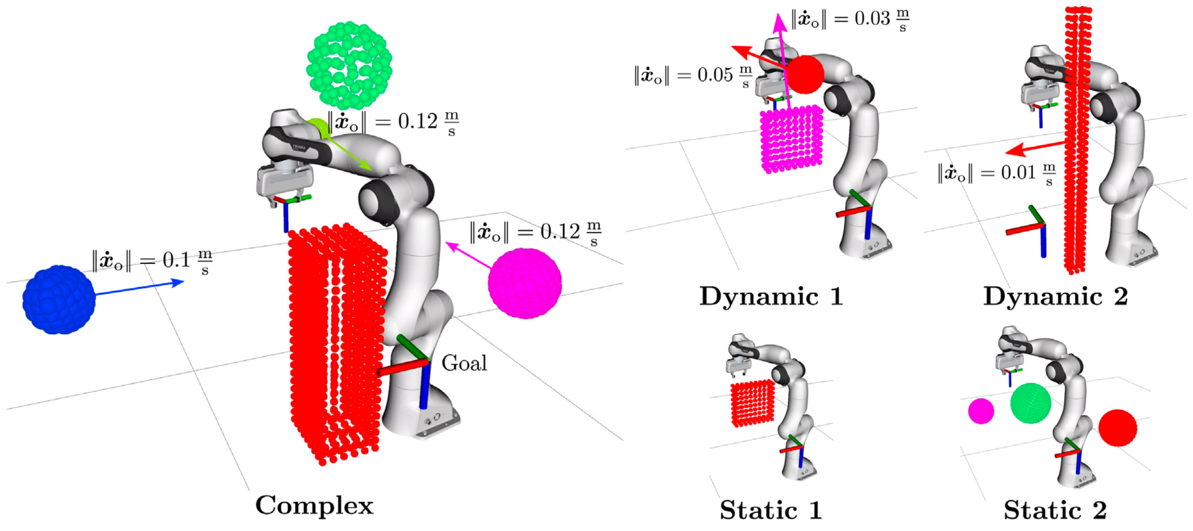
$\|\dot{x}_o\| = 0.03\ \frac{m}{s}$
$\|\dot{x}_o\| = 0.05\ \frac{m}{s}$
$\|\dot{x}_o\| = 0.12\ \frac{m}{s}$
$\|\dot{x}_o\| = 0.01\ \frac{m}{s}$
$\|\dot{x}_o\| = 0.12\ \frac{m}{s}$
$\|\dot{x}_o\| = 0.1\ \frac{m}{s}$
Dynamic 1
Dynamic 2
Goal
Complex
Static 1
Static 2

FIGURE 8
Simulation environments used for comparisons. Figure adapted from Becker et al. (2023a) (CC-BY-NC-ND).

ICF planning framework can be used in combination with arbitrary global motion planning approaches, we first conduct a comparative analysis to determine the most suitable pre-planner within our proposed framework (cf. Section 3.1). Subsequently, in Section 3.2, we compare the ICF planner in combination with the two most effective pre-planners against other state-of-the-art motion planners, encompassing sampling-based, optimization-based, and reactive planning approaches.

Finally, we demonstrate the efficacy of the motion planning framework in a dynamic real-world scenario, where a Franka Emika Research 3 robot has to reactively avoid colliding with a human in its workspace (cf. Section 3.3).

The simulative evaluations are conducted in a kinematic simulation environment without disturbances, wherein the current positions and velocities of the obstacles are known, but their future behaviour is not. We use a C++ implementation on a computer equipped with an AMD® Ryzen 9 5950X CPU with 16 cores, operating at 3.4 GHz and an NVIDIA GeForce RTX 2070 Super GPU.

The same computer is used for calculating the planning commands in the real world experiment. These commands are send to an Intel® Core i7-6700 CPU with 4 cores, 3.4 GHz running a real-time kernel, which is necessary for controlling the Franka Emika robot. For detecting and tracking the human, we use a Sterolabs ZED 2 camera and the corresponding ZED ROS wrapper[3] on a third computer with an Intel® Core i7-6700 CPU with 16 cores, 3.4 GHz and an NVIDIA GeForce RTX 4060 GPU.

Our simulations and experiments involve complex environments with multiple dynamic obstacles. Thus, we uploaded videos of the execution of all scenarios in our accompanying repository Becker et al. (2024). The repository includes comprehensive listings of the parameters utilized in all simulations and experiments.

3 https://github.com/stereolabs/zed-ros-wrapper

3.1 Performance of different global pre-planners

First, we evaluate the performance of the ICF planner using different global pre-planners. Towards this end, we employed 20 different sampling-based planners from the MoveIt! framework, executing them in five static and five dynamic environments. Each global motion planner was configured with a maximum planning time of $T_{\text{global}} = 0.2\text{s}$. To account for the stochastic nature of the sampling-based planners, each planner performed 20 runs in each scenario, resulting in a total of 4,000 simulations. We employed several performance metrics, including the average, minimum, and maximum path length of the EE and the path duration for successful runs. A run was deemed successful if the robot reached the goal within 60s without colliding with any obstacles.

The results are presented in Table 1, with the best results highlighted in bold. Success rates highlighted in red indicate at least one run resulting in a collision. To facilitate cross-scenario performance comparison, we utilized a relative metric. This metric compares the results of each pre-planner against the shortest path length and shortest path duration for each scenario, with a minimum of 0% representing the best value across all scenarios. For instance, a value of 100% in the metric of maximum path length indicates that, on average across all environments, the longest paths generated by the pre-planner were twice as long as the respective shortest paths.

The absolute values of the performance metrics for each individual scenario and planner and illustrations of all scenarios are presented in the accompanying repository Becker et al. (2024). As observed in Table 1, the performance variation among different pre-planners is not substantial. This can be attributed to the predictive agent simulation of the ICF planner, which filters out suboptimal

TABLE 2 Motion planner comparison.

Env.	Method	Length [m]	Duration [s]	Success	Iter. Time [ms]
Static 1	PRM	1.81	-	100%	190.0*
	RRTConnect	1.51	-	100%	74.6*
	RRT*	1.71	-	100%	500.0*
	STOMP	**1.34**	-	100%	120.1*
	Khatib	1.65	17.81	True	0.020
	Ataka	1.86	19.56	True	**0.018**
	CFP	1.49	13.00	True	0.019
	ICF-PRM	1.41	11.88	100%	**0.018**
	ICF-RRT*	1.49	**11.78**	100%	0.019
Static 2	PRM	2.02	-	100%	237.3*
	RRTConnect	2.01	-	100%	257.8*
	RRT*	2.00	-	62%	500.0*
	STOMP	**1.72**	-	43%	163.3*
	Khatib	-	-	False	**0.039**
	Ataka	-	60.00	False	0.045
	CFP	-	-	False	0.209
	ICF-PRM	1.88	30.45	100%	0.057
	ICF-RRT*	2.15	**26.51**	100%	0.053
Dynamic 1	Khatib	1.50	11.46	True	**0.029**
	Ataka	1.53	10.87	True	0.030
	CFP	-	-	False	0.053
	ICF-PRM	**1.31**	**12.55**	100%	0.035
	ICF-RRT*	1.48	13.13	91%	0.033
Dynamic 2	Khatib	1.68	33.81	True	0.020
	Ataka	-	60.00	False	0.021
	CFP	**1.47**	12.75	True	0.029
	ICF-PRM	1.51	**11.96**	100%	0.016
	ICF-RRT*	1.54	12.55	100%	**0.014**
Complex	Khatib	-	-	False	0.082
	Ataka	-	-	False	0.080
	CFP	-	-	False	0.126
	ICF-PRM	**2.12**	**27.42**	71%	**0.048**
	ICF-RRT*	2.20	28.46	62%	0.058

1. Bold values highlight the best results across all methods. 2. Red values indicate at least one occurrence of a collision during the respective run. 3. Iteration times marked with an asterisk (*) correspond to the planning time of global planners. Unmarked iteration times denote the computation time for generating a control reference in reactive planners.

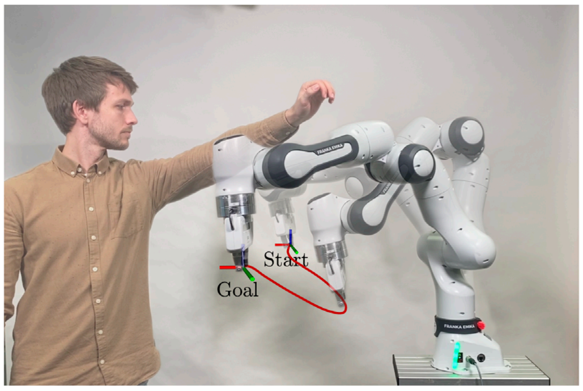

FIGURE 9
Excerpt from the experiment demonstrating the robot's ability to avoid humans, as shown by the red path of its avoidance movement.

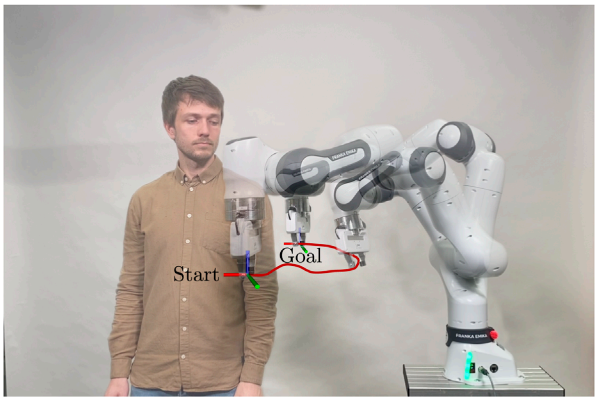

FIGURE 10
Excerpt from the experiment demonstrating the robot's ability to avoid humans, as shown by the red path of its avoidance movement.

choices for the magnetic field vector. For further evaluations and comparisons with other motion planners, we selected two well-performing planners: PRM, owing to its flawless success rate, and RRT*, which exhibited the most best values.

Note that the pre-planner comparisons presented here were performed using a slightly different calculation method described in detail in Becker et al. (2023a). However, we only conduct a relative comparison to identify suitable pre-planners. Therefore, the modifications introduced in this chapter do not affect the relative performances, which we also verified in empirical tests.

3.2 Comparisons to other motion planners

Next, we compare our ICF planner in combination with the chosen pre-planners against other reactive and global motion planning approaches in five additional challenging environments, which are illustrated in Figure 8.

The reactive approaches considered in the following comparative analysis include the APF approach from Khatib (1986), the CF approach from Ataka et al. (2018b) and our extension of the CFP framework for robotic manipulators introduced in Becker et al. (2021). In order to enable a fair comparison of the force command generation, our implementations of the locally reactive controllers integrate our safety-improving fallback method from Section 2.2.4, and our definition of the attractive force from Section 2.1.1. We also compare our planning approach against the global sampling-based planners PRM (Kavraki et al., 1996), RRT*(Karaman and Frazzoli, 2011) and RRTConnect (Kuffner and LaValle, 2000). For a comparison with optimization-based motion planers, we included the MoveIt! implementation of the stochastic trajectory optimization for motion planning (STOMP) planner from Kalakrishnan et al. (2011). The selection of these planners is based on their widespread use, good results in the previous pre-planner analysis, and availability in the MoveIt! framework.

The sampling-based planners and the STOMP planner exclusively consider static obstacles and are consequently tested only in the two static environments, in which a maximum planning time of 0.5s was allowed. If no solution was found in this time, the attempt was considered a failure. All non-deterministic planners were executed until ten successful runs were recorded for each planner in each environment. A path duration exceeding 60s is also considered a failure. Note that *iteration time* refers to the planning time for global planners (depicted with a * in Table 2) while it is used to specify the calculation time for a control reference for the reactive planners. Also note that the path duration depends highly on the allowed maximum velocities. While the EE velocity of the force-based methods can be specified via a Cartesian maximum velocity, the random-based planners used here operate exclusively in joint space. Specifying an EE velocity by means of constraints in the joint space is not straightforward, therefore, a measurement of the duration for the affected planners was omitted.

Overall, the results in Table 2 demonstrate that the ICF planner outperforms the other tested motion planners. Despite its path length in static environments being slightly larger than that of the optimizing STOMP planner, it remains within a comparable range while additionally being capable of reactively avoiding dynamic obstacles. The advantage of the ICF planner becomes particularly evident in the second static environment. Although the environment may not appear complicated at first, a relatively complex joint movement of the robot is required to avoid collisions with the obstacles. This complexity leads to challenges for the optimizing STOMP and RRT*, reflected in significantly lower success rates. The CFP planner also faced challenges in this environment due to the extensive number of predictive agents created, resulting in increased iteration times. Consequently, no suitable combination of avoidance directions could be found, and opposing forces on the control points led to a collision.

The *complex* environment (cf. Figure 8) requires simultaneous avoidance motions of the EE and the body, aiming to test the limits of the motion planners; our ICF planner was the only planner able to reach the goal successfully without a collision. However, as can be seen in Table 2, it did not succeed in all runs. The ICF planner failed when the pre-planner repeatedly could not find a path to the goal within the given planning time of 0.2s. In such cases, the ICF planner utilized the safety-improving fallback method, and the obstacles pushed the robot into configurations close to the joint

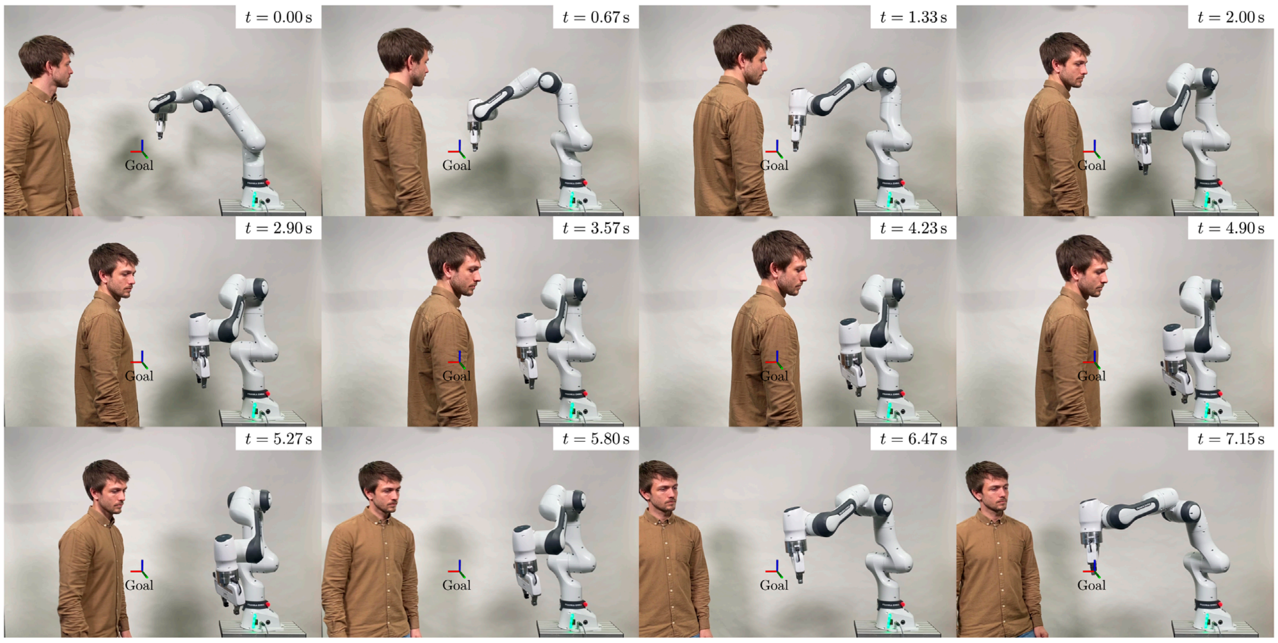

FIGURE 11
Excerpt from the experiment demonstrating the robot's ability to avoid humans. The figure displays a sequence of timeframes where the human obstructs all potential paths to the goal pose, causing the robot to deviate from its original path.

limits, where no avoidance was possible, leading to an inevitable collision. Similarly, in one execution in the environment *Dynamic 1*, the ICF-RRT* planner ended up in a configuration close to the joint limits and was unable to reach the goal afterwards.

3.3 Human-robot interaction experiments

In this section, we demonstrate the capabilities of the planning framework in a real-world experimental setup with a Franka Emika Research 3 robot. The robot is programmed to move continuously between two predefined goal poses with a maximum velocity of 0.65 ms^{-1}. The goal poses are updated when the robot is within a distance of 0.05 m of the current goal. During the execution of this task a human enters the workspace of the robot several times.

We use the body tracking feature of the Sterolabs ZED SDK[4], which enables the identification and tracking of 18 keypoints on the human body. For the purposes of this experiment, we use a subset of these keypoints, specifically those corresponding to the upper body. These keypoints are used to create cylindrical approximations of the human torso, head, and arms, providing a simplified yet accurate representation of the dynamic obstacle. These approximations are then published in point cloud format, which is directly processed by our motion planning framework to ensure real-time adaptability. The interaction between the tracked human body and the robot's movement can be observed in the accompanying video Becker et al. (2024), which shows how the cylindrical representations move in sync with the human collaborator.

The planner consistently demonstrates its ability to reactively avoid various dynamic motions of the human (cf. Figures 9, 10) even when the human obstructs all possible paths to the goal pose (cf. Figure 11).

4 Conclusion

In this paper, we introduced the ICF planning algorithm, which integrates reactive collision avoidance with global exploration by leveraging information about feasible avoidance directions from a global pre-planner. The ICF planner has been rigorously tested in a variety of simulations, demonstrating robust performance even in complex environments with multiple dynamic obstacles. Comparisons with other widely-used global and locally reactive motion planners demonstrate its superiority. Additionally, we validated the planning framework in a challenging real-world experiment, demonstrating its capability to react promptly and safely to fast movements of humans within its workspace. A promising extension could be to further exploit the high parallelizability of the planner by using multiple pre-planners simultaneously. In particular, a combination of fast global planners and optimizing planners with higher planning times is a promising extension. Future research should focus on improving the global information extraction module. Currently, it is unable to extract sufficient information when more complex obstacle avoidance motions are required.

4 https://www.stereolabs.com/docs/body-tracking

Data availability statement

The datasets presented in this study can be found in online repositories. The names of the repository/repositories and accession number(s) can be found in the article/supplementary material.

Ethics statement

Written informed consent was obtained from the individual(s) for the publication of any potentially identifiable images or data included in this article.

Author contributions

MB: Conceptualization, Data curation, Formal Analysis, Investigation, Methodology, Software, Validation, Visualization, Writing - original draft. PC: Investigation, Methodology, Software, Writing - review and editing. TL: Methodology, Supervision, Writing - review and editing. SH: Supervision, Writing - review and editing, MM: Conceptualization, Funding acquisition, Project administration, Resources, Supervision, Writing - review and editing.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. This work was partially funded by the Region Hannover in the project *roboterfabrik*.

Acknowledgments

The authors would like to thank Dwayne Steinke, Tom Hattendorf, Arian Pajaziti and Fabrice Zeug for the help with the implementation of the planner code, the vision system and the experiments. We would also like to thank Jonathan Vorndamme for his help with the joint impedance controller.

Conflict of interest

Author SH is a shareholder of Franka–Emika GmbH.

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

References

Adiyatov, O., and Varol, H. A. (2017). "A novel RRT-based algorithm for motion planning in Dynamic environments," in *Proc. IEEE int. Conf. Mechat. Autom*, 1416–1421. doi:10.1109/ICMA.2017.8016024

Ataka, A., Lam, H. K., and Althoefer, K. (2018a). "Reactive magnetic-field-inspired navigation for non-holonomic mobile robots in unknown environments," in Proc. IEEE Int. Conf. Robot. Autom, 6983–6988. doi:10.1109/icra.2018.8463203

Ataka, A., Lam, H.-K., and Althoefer, K. (2018b). Reactive magnetic-field-inspired navigation method for robots in unknown convex 3-d environments. *IEEE Robot. Autom. Lett.* 3, 3583–3590. doi:10.1109/lra.2018.2853801

Ataka, A., Lam, H.-K., and Althoefer, K. (2022). Magnetic-field-inspired navigation for robots in complex and unknown environments. *Front. Robotics AI* 9, 834177. doi:10.3389/frobt.2022.834177

Ataka, A., Shiva, A., Lam, H. K., and Althoefer, K. (2018c). "Magnetic-field-inspired navigation for soft continuum manipulator," in Proc. IEEE Int. Conf. Robot. Autom, 168–173. doi:10.1109/IROS.2018.8593592

Barbazza, L., Faccio, M., Oscari, F., and Rosati, G. (2017). Agility in assembly systems: a comparison model. *Assem. Autom.* 37, 411–421. doi:10.1108/aa-10-2016-128

Becker, M., Caspers, P., Hattendorf, T., Lilge, T., Haddadin, S., and Müller, M. A. (2023a). "Informed circular fields for global reactive obstacle avoidance of robotic manipulators," in *22nd international federation of automatic control world congress (IFAC WC 2023) (IFAC-PapersOnLine).*

Becker, M., Caspers, P., Lilge, T., Haddadin, S., and Müller, M. A. (2024). Informed circular fields for global reactive obstacle avoidance of robotic manipulators. doi:10.25835/sr51tdkd

Becker, M., Köhler, J., Haddadin, S., and Müller, M. A. (2023b). Motion planning using reactive circular fields: a 2D analysis of collision avoidance and goal convergence. *IEEE Trans. Automatic Control* 69, 1552–1567. Early access. doi:10.1109/TAC.2023.3303168

Becker, M., Lilge, T., Müller, M. A., and Haddadin, S. (2021). Circular fields and predictive multi-agents for online global trajectory planning. *IEEE Robot. Autom. Lett.* 6, 2618–2625. doi:10.1109/lra.2021.3061997

Bohlin, R., and Kavraki, L. E. (2000). Path planning using lazy PRM. *Proc. IEEE Int. Conf. Robot. Autom.* 1, 521–528 vol.1. doi:10.1109/robot.2000.844107

Brock, O., and Khatib, O. (2002). Elastic strips: a framework for motion generation in human environments. *Int. J. Robot. Res.* 21, 1031–1052. doi:10.1177/027836402128964189

Burget, F., Bennewitz, M., and Burgard, W. (2016). "BI2RRT*: an efficient sampling-based path planning framework for task-constrained mobile manipulation," in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 3714–3721. doi:10.1109/IROS.2016.7759547

Cefalo, M., Magrini, E., and Oriolo, G. (2017). "Parallel collision check for sensor based real-time motion planning," in Proc. IEEE Int. Conf. Robot. Autom, 1936–1943. doi:10.1109/icra.2017.7989225

Chen, J.-H., and Song, K.-T. (2018). "Collision-free motion planning for human-robot collaborative safety under cartesian constraint," in Proc. IEEE Int. Conf. Robot. Autom, 4348–4354. doi:10.1109/icra.2018.8460185

Coleman, D., Sucan, I., Chitta, S., and Correll, N. (2014). Reducing the barrier to entry of complex robotic software: a MoveIt! Case study. *J. Soft. Eng. Robot.* doi:10.6092/JOSER_2014_05_01_p3

Connell, D., and La, H. M. (2017). "Dynamic path planning and replanning for mobile robots using RRT," in Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC), 1429–1434. doi:10.1109/smc.2017.8122814

Connell, D., and La, H. M. (2018). Extended rapidly exploring random tree–based dynamic path planning and replanning for mobile robots. *Int. J. Adv. Robot. Syst.* 15. doi:10.1177/1729881418773874

Connolly, C. I., Burns, J. B., and Weiss, R. (1990). "Path planning using Laplace's equation," in Proc. IEEE Int. Conf. Robot. Autom, 2102–2106. doi:10.1109/ROBOT.1990.126315

Devaurs, D., Siméon, T., and Cortés, J. (2013). "Enhancing the transition-based rrt to deal with complex cost spaces," in Proc. IEEE Int. Conf. Robot. Autom, 4120–4125. doi:10.1109/ICRA.2013.6631158

Dobson, A., and Bekris, K. E. (2013). "Improving sparse roadmap spanners," in Proc. IEEE Int. Conf. Robot. Autom, 4106–4111.

Dobson, A., Krontiris, A., and Bekris, K. E. (2013). "Sparse roadmap spanners," in *Alg. Found. Robot. X*. Editors E. Frazzoli, T. Lozano-Perez, N. Roy, and D. Rus (Springer Berlin Heidelberg), 279–296.

Dufour, K., and Suleiman, W. (2020). On maximizing manipulability index while solving a kinematics task. *J. Intell. Robot. Sys.* 100, 3–13. doi:10.1007/s10846-020-01171-7

Elmokadem, T., and Savkin, A. V. (2021). A hybrid approach for autonomous collision-free uav navigation in 3d partially unknown dynamic environments. *Drones* 5, 57. doi:10.3390/drones5030057

El Zaatari, S., Marei, M., Li, W., and Usman, Z. (2019). Cobot programming for collaborative industrial tasks: an overview. *Robotics Aut. Syst.* 116, 162–180. doi:10.1016/j.robot.2019.03.003

Faccio, M., Bottin, M., and Rosati, G. (2019). Collaborative and traditional robotic assembly: a comparison model. *Int. J. Adv. Manuf. Technol.* 102, 1355–1372. doi:10.1007/s00170-018-03247-z

Fechter, M., Foith-Förster, P., Pfeiffer, M. S., and Bauernhansl, T. (2016). "Axiomatic design approach for human-robot collaboration in flexibly linked assembly layouts," in Procedia CIRP 26th CIRP Design Conference, 629–634. doi:10.1016/j.procir.2016.04.186

Flacco, F., Kröger, T., de Luca, A., and Khatib, O. (2012). "A depth space approach to human-robot collision avoidance," in Proc. IEEE Int. Conf. Robot. Autom, 338–345.

Gammell, J. D., Barfoot, T. D., and Srinivasa, S. S. (2020). Batch Informed Trees (BIT*): informed asymptotically optimal anytime search. *Int. J. Robot. Res.* 39, 543–567. doi:10.1177/0278364919890396

Gipson, B., Moll, M., and Kavraki, L. E. (2013). "Resolution independent density estimation for motion planning in high-dimensional spaces," in Proc. IEEE Int. Conf. Robot. Autom, 2437–2443. doi:10.1109/icra.2013.6630908

Grothe, F., Hartmann, V. N., Orthey, A., and Toussaint, M. (2022). "ST-RRT: asymptotically-optimal bidirectional motion planning through space-time," in *Proc. IEEE int. Conf. Robot. Autom. (IEEE)*, 3314–3320.

Haddadin, S., Belder, R., and Albu-Schäffer, A. (2011). Dynamic motion planning for robots in partially unknown environments. *IFAC Proc.* 44, 6842–6850. doi:10.3182/20110828-6-it-1002.02500

Haddadin, S., Urbanek, H., Parusel, S., Burschka, D., Roßmann, J., Albu-Schäffer, A., et al. (2010). "Real-time reactive motion generation based on variable attractor dynamics and shaped velocities," in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst, 3109–3116. doi:10.1109/IROS.2010.5650246

Halliday, D., Resnick, R., and Walker, J. (2013). *Fundamentals of physics*. John Wiley and Sons.

Hsu, D., Latombe, J.-C., and Motwani, R. (1997). "Path planning in expansive configuration spaces," in Proc. IEEE Int. Conf. Robot. Autom., 2719–2726. doi:10.1109/robot.1997.619371

Huang, S., Teo, R. S. H., and Tan, K. K. (2019). Collision avoidance of multi unmanned aerial vehicles: a review. *Annu. Rev. Control* 48, 147–164. doi:10.1016/j.arcontrol.2019.10.001

Jaillet, L., Cortés, J., and Siméon, T. (2010). Sampling-based path planning on configuration-space costmaps. *IEEE Trans. Robot.*, 635–646. doi:10.1109/TRO.2010.2049527

Kalakrishnan, M., Chitta, S., Theodorou, E., Pastor, P., and Schaal, S. (2011). "STOMP: stochastic trajectory optimization for motion planning," in Proc. IEEE Int. Conf. Robot. Autom, 4569–4574. doi:10.1109/ICRA.2011.5980280

Kappler, D., Meier, F., Issac, J., Mainprice, J., Cifuentes, C. G., Wüthrich, M., et al. (2018). Real-time perception meets reactive motion generation. *IEEE Robot. Autom. Lett.* 3, 1864–1871. doi:10.1109/LRA.2018.2795645

Karaman, S., and Frazzoli, E. (2011). Sampling-based algorithms for optimal motion planning. *Int. J. Robot. Res.* 30, 846–894. doi:10.1177/0278364911406761

Kavraki, L. E., Svestka, P., Latombe, J.-C., and Overmars, M. H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robot. Autom.* 12, 566–580. doi:10.1109/70.508439

Khansari-Zadeh, S. M., and Billard, A. (2012). A dynamical system approach to realtime obstacle avoidance. *Aut. Robots* 32, 433–454. doi:10.1007/s10514-012-9287-y

Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. *Int. J. Robot. Res.* 5, 90–98. doi:10.1177/027836498600500106

Kragic, D., Gustafson, J., Karaoguz, H., Jensfelt, P., and Krug, R. (2018). "Interactive, collaborative robots: challenges and opportunities," in Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI), 18–25. doi:10.24963/ijcai.2018/3

Krüger, J., Lien, T., and Verl, A. (2009). Cooperation of human and machines in assembly lines. *CIRP Ann.* 58, 628–646. doi:10.1016/j.cirp.2009.09.009

Kuffner, J. J., and LaValle, S. M. (2000). RRT-connect: an efficient approach to single-query path planning. In Proc. IEEE Int. Conf. Robot. Autom., 995–1001.

Ladd, A. M., and Kavraki, L. E. (2005). "Motion planning in the presence of drift, underactuation and discrete system changes," in *Robot: science and syst*, 1, 233–240.

LaValle, S. M. (1998). Rapidly-exploring random trees: a new tool for path planning. *Tech. Rep.*

Li, H., Wang, Z., and Ou, Y. (2019). "Obstacle avoidance of manipulators based on improved artificial potential field method," in *Proc. IEEE int. Conf. Robot. Biomim*, 564–569.

Li, S., and Dantam, N. T. (2023). "Sample-driven connectivity learning for motion planning in narrow passages," in Proc. IEEE Int. Conf. Robot. Autom. (IEEE), 5681–5687. doi:10.1109/icra48891.2023.10161339

Li, S., Han, K., Li, X., Zhang, S., Xiong, Y., and Xie, Z. (2021). Hybrid trajectory replanning-based dynamic obstacle avoidance for physical human-robot interaction. *J. Intell. and Robot. Syst.* 103, 41. doi:10.1007/s10846-021-01510-2

Liu, G., and Jiang, Y. (2018). "Research on dynamic trajectory planning of collaborative robots base on RRT-RV algorithm," in Proc. IEEE 4th Inf. Tech. and Mechatronics Eng. Conf. (ITOEC), 1020–1023. doi:10.1109/itoec.2018.8740744

Liu, H., Qu, D., Xu, F., Du, Z., Jia, K., and Liu, M. (2022). An efficient online trajectory generation method based on kinodynamic path search and trajectory optimization for human-robot interaction safety. *Entropy* 24, 653. doi:10.3390/e24050653

Luo, R. C., Ko, M.-C., Chung, Y.-T., and Chatila, R. (2014). "Repulsive reaction vector generator for whole-arm collision avoidance of 7-DoF redundant robot manipulator," in Proc. IEEE/ASME Int. Conf. on Adv. Intell. Mechat, 1036–1041. doi:10.1109/aim.2014.6878217

Matheson, E., Minto, R., Zampieri, E. G. G., Faccio, M., and Rosati, G. (2019). Human–robot collaboration in manufacturing applications: a review. *Robotics* 8, 100. doi:10.3390/robotics8040100

Meziane, R., Otis, M. J. D., and Ezzaidi, H. (2017). Human-robot collaboration while sharing production activities in dynamic environment: SPADER system. *Robotics Computer-Integrated Manuf.* 48, 243–253. doi:10.1016/j.rcim.2017.04.010

Nakanishi, J., Cory, R., Mistry, M., Peters, J., and Schaal, S. (2008). Operational space control: a theoretical and empirical comparison. *Int. J. Robot. Res.* 27, 737–757. doi:10.1177/0278364908091463

Niloy, M. A. K., Shama, A., Chakrabortty, R. K., Ryan, M. J., Badal, F. R., Tasneem, Z., et al. (2021). Critical design and control issues of indoor autonomous mobile robots: a review. *IEEE Access* 9, 35338–35370. doi:10.1109/ACCESS.2021.3062557

Ravankar, A. A., Ravankar, A., Emaru, T., and Kobayashi, Y. (2020). HPPRM: hybrid potential based probabilistic roadmap algorithm for improved dynamic path planning of mobile robots. *IEEE Access* 8, 221743–221766. doi:10.1109/access.2020.3043333

Reinhart, G., and Loy, M. (2010). Design of a modular feeder for optimal operating performance. *CIRP J. Manuf. Sci. Technol.* 3, 191–195. doi:10.1016/j.cirpj.2010.09.003

Rusu, R. B., and Cousins, S. (2011). "3D is here: point cloud library (PCL)," in Proc. IEEE Int. Conf. Robot. Autom.

Salzman, O., and Halperin, D. (2016). Asymptotically near-optimal rrt for fast, high-quality motion planning. *IEEE Trans. Robot.* 32, 473–483. doi:10.1109/TRO.2016.2539377

Sánchez, A., Cuautle, R., Zapata, R., and Osorio, M. (2006). "A reactive lazy prm approach for nonholonomic motion planning," in *Advances in artificial intelligence - IBERAMIA-SBIA 2006* (Springer), 542–551.

Sánchez, G., and Latombe, J.-C. (2003). "A single-query bi-directional probabilistic roadmap planner with lazy collision checking," in *Robot. Res.: Int. Symp.* (Springer), 403–417.

Schulman, J., Duan, Y., Ho, J., Lee, A., Awwal, I., Bradlow, H., et al. (2014). Motion planning with sequential convex optimization and convex collision checking. *Int. J. Robotics Res.* 33, 1251–1270. doi:10.1177/0278364914528132

Short, A., Pan, Z., Larkin, N., and van Duin, S. (2016). Recent progress on sampling based dynamic motion planning algorithms. 1305–1311. doi:10.1109/AIM.2016.7576950

Siciliano, B., and Khatib, O. (2008). *Springer handbook of robotics*. Springer. doi:10.1007/978-3-540-30301-5

Singh, L., Stephanou, H., and Wen, J. (1996). "Real-time robot motion control with circulatory fields," in Proc. IEEE Int. Conf. Robot. Autom., 2737–2742. doi:10.1109/robot.1996.506576

Singh, L., Wen, J., and Stephanou, H. (1997). "Motion planning and dynamic control of a linked manipulator using modified magnetic fields," in Proc. IEEE Int. Conf. Control Appl. (IEEE), 9–15. doi:10.1109/cca.1997.627455

Şucan, I. A., and Kavraki, L. E. (2009). "Kinodynamic motion planning by interior-exterior cell exploration," in *Algorithmic foundation of robotics VIII: selected contributions of the eight international workshop on the algorithmic foundations of robotics* (Springer), 449–464.

Wang, W., Zhu, M., Wang, X., He, S., He, J., and Xu, Z. (2018). An improved artificial potential field method of trajectory planning and obstacle avoidance for redundant manipulators. *Int. J. Adv. Robot. Syst.* 15, 172988141879956. doi:10.1177/1729881418799562

Yingqi, X., Wei, S., Wen, Z., Jingqiao, L., Qinhui, L., and Han, S. (2021). A real-time dynamic path planning method combining artificial potential field method and biased target RRT algorithm. *J. Phys. Conf. Ser.* 1905, 012015. doi:10.1088/1742-6596/1905/1/012015

Yoshikawa, T. (1985). Manipulability of robotic mechanisms. *Int. J. Robot. Res.* 4, 3–9. doi:10.1177/027836498500400201

Yuan, J. S. (1988). Closed-loop manipulator control using quaternion feedback. *IEEE J. Robot. Autom.* 4, 434–440. doi:10.1109/56.809