Dynamic Optimization Fabrics for Motion Generation

Max Spahn, Martijn Wisse, and Javier Alonso-Mora

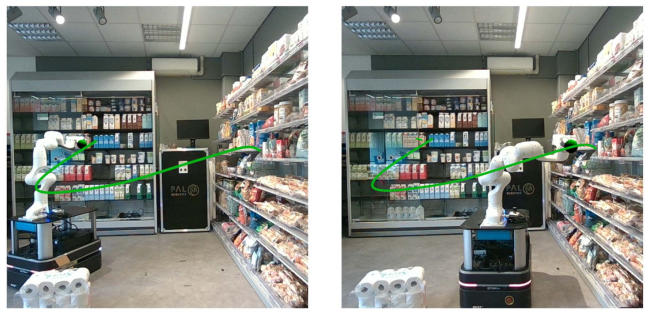
***Abstract*—Optimization fabrics are a geometric approach to real-time local motion generation, where motions are designed by the composition of several differential equations that exhibit a desired motion behavior. We generalize this framework to dynamic scenarios and nonholonomic robots and prove that fundamental properties can be conserved. We show that convergence to desired trajectories and avoidance of moving obstacles can be guaranteed using simple construction rules of the components. In addition, we present the first quantitative comparisons between optimization fabrics and model predictive control and show that optimization fabrics can generate similar trajectories with better scalability, and thus, much higher replanning frequency (up to 500 Hz with a 7 degrees of freedom robotic arm). Finally, we present empirical results on several robots, including a nonholonomic mobile manipulator with 10 degrees of freedom and avoidance of a moving human, supporting the theoretical findings.**

***Index Terms*—Geometric control, mobile manipulation, motion control of manipulators, nonholonomic motion planning.**

I. Introduction

ROBOTS increasingly populate dynamic environments. Imagine a robot operating alongside customers in a supermarket. It is requested to perform different tasks, such as cleaning the floor or picking a wide range of products. These different manipulation tasks may vary in their dimension and accuracy requirements, e.g., rotation around a suction gripper does not need to be specified while two-finger grippers require full poses. Thus, it is important for motion planning algorithms to support various goal definitions. Further, the robot is operating alongside humans, and it has to constantly react to the changing environment and consequently update an initial plan. As customers move fast, the adaptations must be computed in real time. Therefore, motion planning is often divided into global motion planning [1] and local motion planning, which we will refer to as motion generation in this article. A global planner generates a first feasible path that is used by a motion generator as global guidance. This article proposes a novel approach to motion generation that deals with a variety of different goal definitions.

Motion generation is often solved by formulating an optimization problem over a time horizon. The popularity of this approach is partly thanks to the guaranteed collision avoidance and, thus, safety [2], [3]. The optimization problem is then assembled from a scalar objective function, encoding the motion planning problem (e.g., the desired final position, path constraints, etc.), the transition function, defining the robot's dynamics, and several inequality constraints, such as integrating physical limits and obstacle avoidance. Despite abundant applications of such optimization-based approaches to mobile robots, the computational costs limit applicability when dealing with high-dimensional configuration spaces [4], [5]. Data-driven approaches to speed up the optimization process usually come with reduced generalization abilities, loss of formal guarantees [3], and require prior, often costly, data acquisition. Moreover, due to the scalar objective function, the user must carefully weigh up different parts of the objective function. As a consequence, optimization-based approaches are challenging to tune and inflexible to generic motion planning problems with variable goal objectives [6], [7].



(a) (b)

Fig. 1. DF for path (green) following with a nonholonomic mobile manipulator. DF control all actuators simultaneously to follow the end-effector path while keeping a given orientation and avoiding collision with the environment.

In the field of geometric control, namely Riemannian motion policies (RMPs) and optimization fabrics, all individual parts of the motion planning problem are formulated as differential equations of second order. Applying operations from differential geometry, the individual components are combined in the configuration space to define the resulting motion [8], [9]. This allows to iteratively *design* the motion of the robot while maintaining explainability over the resulting motion [6], [8], [9], [10].

These works on optimization fabrics [9], but also on predecessors, such as RMP [8] and RMP-Flow [11], have shown the power of designing reactive behavior as second-order differential equations. However, integration of dynamic features, such as moving obstacles and path following, have not been proposed nor have the framework been applied to nonholonomic systems. In this article, we exploit relative coordinate systems

Manuscript received 7 December 2022; accepted 13 February 2023. This work was supported by Ahold Delhaize. This paper was recommended for publication by Associate Editor O. Stasse and Editor F. Chaumette upon evaluation of the reviewers' comments. *(Corresponding author: Max Spahn.)*

The authors are with the Department of Cognitive Robotics, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: m.spahn@tudelft.nl; m.wisse@tudelft.nl; j.alonsomora@tudelft.nl).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TRO.2023.3255587.

Digital Object Identifier 10.1109/TRO.2023.3255587

in the framework of optimization fabrics by introducing the dynamic pullback operation (8). This generalization can then integrate moving obstacles and path following. We show that our generalization maintains guaranteed convergence for path-following tasks and improves collision avoidance with moving obstacles. Moreover, we propose a method to incorporate nonholonomic constraints. Finally, we compare a trajectory optimization formulation, namely a model predictive control (MPC) formulation, with optimization fabrics to provide the reader with a better understanding of key differences between the two approaches. We analyze computational costs and the quality of resulting trajectories for different robots. Several simulated results and real-world experiments show the practical implications of dynamic fabrics (DF). The contributions of this article can be summarized as follows.

1) We enable the usage of optimization fabrics for dynamic scenarios. Specifically, we propose time parameterized differential maps using up-to second-order predictor models. As a consequence, this enables the integration of moving obstacles and path-following tasks, see Fig. 3.
2) We extend the framework of optimization fabrics to nonholonomic robots.
3) We present a quantitative comparison between MPC and optimization fabrics. The results reveal that fabrics are an order of magnitude faster, more reliable, and easier to tune for goal-reaching tasks with a robotic manipulator in static environments.

All findings are supported by extensive experiments in both simulation and real world with a manipulator, a differential drive robot, and a mobile manipulator.

II. Related Work

In dynamic environments, global planning methods are not sufficient due to low planning frequencies. Thus, local motion generation methods, such as the one presented in this work, are employed. These methods typically require guidance to avoid local minima and, thus, effectively solve planning problems.

A. Task Constrained Global Motion Planning

Motion planning problems are usually defined by goals in arbitrary task spaces, such as the 3-D Euclidean space or end-effector poses. In this context, tasks can be regarded as constraints to the motion planning problem. Conventional approaches to motion planning rely on inverse kinematics to transform task constraints into sets of configurations. The resulting global motion planning problem is then often solved using sampling-based methods [12].

Sampling-based motion planners generate random configurations until a valid path between an initial configuration and a set of goal configurations is found [1]. Several methods have been proposed to directly integrate task constraints into the sampling phase. Stilman [13] proposed a method to iteratively push a random sample to the manifold adhering to the task constraint. The notion of task constraints was later extended to task space regions to define soft constraints for individual task components [14]. Kingston et al. [15] proposed scalar-valued functions to represent task constraints for sampling-based planning. As all of the abovementioned methods rely on implicitly constrained sampling in the joint space, they exhibit high computational time, which is especially harmful to real-world applications [16] and require local motion generation methods for path following and execution in dynamic environments [17]. In the next section, recent developments in local motion planning are summarized.

B. Receding-Horizon Trajectory Optimization

Methods formulating motion generation as an optimization problem with a finite discrete time horizon are known under the name of receding-horizon trajectory optimization. In line with most literature in robotics, we will refer to such methods as MPC. Generally, several objectives are encoded in the scalar cost function, dynamics are formulated as equality constraints, and inequality constraints ensure collision avoidance and joint limit avoidance. The dynamics for this problem can include the full dynamics model or simple integrating schemes [3]. By explicitly solving the constrained optimization problem, this approach yields formal guarantees on stability. Stability for model predictive control (MPC) is proven by formulating an appropriate Lyapunov function and showing that the finite time-horizon formulation with an appropriate terminal cost results in the same stability as the corresponding infinite time-horizon formulation [18], [19], [20]. MPC has been applied to various robotic systems in dynamic environments, such as drones [21], mobile robots [17], and mobile manipulators [22], [23]. Despite these results, formal stability guarantees in such environments are challenging as appropriate terminal cost functions are often not computable or too conservative. Besides, the computational costs scale with the degrees of freedom restricting real-time applicability to simple dynamics and environment models [24].

Some MPC formulations are nonlinear and can be analyzed using methods from nonlinear control. When analyzing nonlinear control system, Riemannian energies lead to more detailed stability results than Lyapunov functions. By investigating the variation around the generated trajectory and its contracting toward the desired trajectory, some control designs show exponential stabilizing properties [25]. These findings have been applied to tracking control problems [26].

C. RMPs and Fabrics

Based on the findings of contracting metrics for nonlinear control design [25], [26], geometric control approaches design the motion generation such that convergence is inherent to the problem formulation rather than imposing them on the solution process. Practically, individual constraints to the motion planning problem shape the optimization manifold so that the solution is accessible through the solution of simple differential equation. An example for shaping the optimization manifold is seen in Fig. 2.

Realizing this concept, RMPs represent a natural way of combining multiple policies into one joined policy. RMPs define individual subtasks of the motion planning as differential equations (*spectral semi sprays* or *specs* for short) of second order

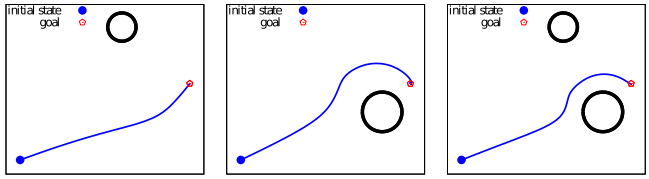


Fig. 2. Combining different avoidance behaviors using optimization fabrics. The components defining collision avoidance with single obstacles (a) and (b) are combined in (c). Obstacles are shown in black. Trajectories of the point robot are shown in blue.

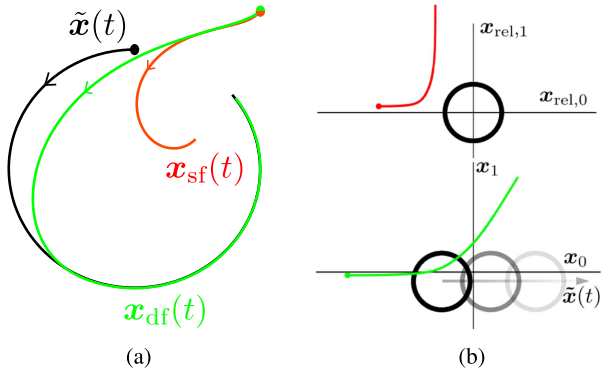


Fig. 3. Two implications of DF. (a) It can be seen that the trajectory obtained with DF (green) converges toward the reference trajectory (black) while the trajectory with SF (red) does not converge. (b) Top part visualizes collision avoidance, as suggested in [9]. Here, the trajectory and obstacle are expressed in a relative system $\boldsymbol{x}_{\mathrm{rel}}$. Using the dynamic pull, (8), this can be transformed into the static reference frame $\boldsymbol{x}$, bottom part. Together with dynamic energization, the framework of optimization fabrics is leveraged for dynamic environments. The motion of the obstacle $\boldsymbol{x}_{\mathrm{rel}}(t)$ is visualized with an arrow, and future positions of the obstacle are shown in lighter color. The resulting trajectory obtained with DF is shown in green. (a) Dynamic convergence. (b) Dynamic avoidance.

and propose the *pullback* and *summation* operators to combine multiple policies in the configuration space. As subtasks can be defined in arbitrary manifolds of the configuration space, and RMP generalizes operational space control [27]. The resulting behavior of RMP was reported to be intuitive while keeping computational costs low [28]. The concept of RMP was used in [8] and [11] to form RMP-Flow, a motion planning algorithm that is shown to be conditionally stable and invariant across robots. In RMP-Flow, individual tasks are represented as a pair of a motion policy and a corresponding metric defining the importance of individual directions. An RMP adaptation was proposed for nonholonomic robots in [29]. By incorporating the kinematic constraint into the root equation of the RMP, the computed policy is applicable to nonholonomic robots. Besides, that work proposed a neural net to learn the collision avoidance task components.

Although RMPs have proven to be a powerful tool for motion generation, it was reported to require intuition and experience during tuning [9]. Optimization fabrics with Finsler structures as metric generators simplify the motion design as the conditions for stability and convergence are inherent to the definition of Finsler structures [9], [30]. Opposed to RMPs, where the metric is typically user defined, fabrics derive Finsler metrics from artificial energies, similar to approaches from control design, [25], [26], using the Euler–Lagrange equation from geometric mechanics. Although fabrics generalize the concept of RMPs and make it accessible to a broader audience by decreasing the intuition and expertise required, they have not yet been applied to a wide range of robots.

The reason for this lack of application of fabrics is twofold. First, all the abovementioned methods are reactive and highly local methods, thus making them prone to local minima [31]. As RMPs and optimization fabrics do not incorporate path following, integration of global planning to overcome local minima is not possible to this date. Second, fabrics and RMP do not make use of velocity estimates of obstacles but rely purely on their high reactivity in dynamic environments. As for other trajectory optimization techniques, motion estimates could benefit fabrics (and RMP) to result in even smoother motion and allow applications in such environments.

In this article, we address these issues by proposing time parameterized differential maps to form dynamic fabrics. This generalization integrates path following and velocity estimates of moving obstacles. Together with the extension to nonholonomic robots, our method allows to deploy the promising theory of optimization fabrics to mobile manipulators, operating in dynamic environments.

III. Background

In the previous section, we have highlighted that optimization fabrics represent a powerful tool for reactive motion generation. Since we generalize this concept, this section aims at familiarizing the reader with key findings on optimization fabrics and recalling some of the basic notations known from differential geometry. We first give an overview on how optimization fabrics are used for motion generation and how the components are derived and composed. Then, the theoretical foundations are summarized from [9] and [11].

A. Motion Generation Using Optimization Fabrics

When using optimization fabrics for motion generation, all components, including constraints and goal attraction, are designed as second-order differential equations. If specific design rules for these equations are respected, all components can be combined to form a converging motion generator. Specifically, the following steps are performed.

1) Design path-consistent geometries in suited manifolds of the configuration space (7).
2) Design corresponding Finsler energies defining the importance metric in this manifold (see Section III-F).
3) Energize all geometries with the associated Finsler energies (see Section III-F).
4) Pull back the energized systems into the configuration space and sum them (see Section III-D).
5) Force the combined system with a differentiable potential. As a composition of optimization fabrics, the resulting trajectory converges toward the potential's minimum (see Section III-E).

In the following, we introduce the reader to the theory of optimization fabrics and recall important findings from [9].

B. Configurations and Task Variables

We denote $\boldsymbol{q} \in \mathcal{Q} \subset \mathbb{R}^n$ a configuration of the robot with n its degrees of freedom; $\mathcal{Q}$ is the configuration space of the generalized coordinates of the system. Generally, $\boldsymbol{q}(t)$ defines the robot's configuration at time t, so that $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$ define the instantaneous derivatives of the robot's configuration. Similarly, we assume that there is a set of task variables $\boldsymbol{x}_j \in \mathcal{X}_j \subset \mathbb{R}^{m_j}$ with variable dimension $m_j \leq n$. The task manifold $\mathcal{X}_j$ defines an arbitrary manifold of the configuration space $\mathcal{Q}$ in which a robotic task can be represented. Further, we assume that there is a differential map $\phi_j : \mathbb{R}^n \to \mathbb{R}^{m_j}$ that relates the configuration space to the jth task space. For example, when a task variable is defined as the end-effector position, then ϕ_j is the positional part of the forward kinematics. On the other hand, if a task variable is defined to be the joint position, then ϕ_j is the identity function. In the following, we drop the subscript j in most cases for readability when the context is clear.

In this work, we assume that ϕ is smooth and twice differentiable so that the Jacobian is defined as

$$\boldsymbol{J}_\phi = \frac{\partial \phi}{\partial \boldsymbol{q}} \in \mathcal{R}^{m \times n} \tag{1}$$

or $\boldsymbol{J}_\phi = \partial_{\boldsymbol{q}} \phi$ for short. Thus, we can write the total time derivatives of $\boldsymbol{x}$ as

$$\dot{\boldsymbol{x}} = \boldsymbol{J}_\phi \dot{\boldsymbol{q}} \tag{2}$$

$$\ddot{\boldsymbol{x}} = \boldsymbol{J}_\phi \ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}_\phi \dot{\boldsymbol{q}}. \tag{3}$$

C. Spectral Semisprays

Inspired by simple mechanics (e.g., the simple pendulum), the framework of optimization fabrics designs motion generation as second-order dynamical systems $\ddot{\boldsymbol{x}} = \pi(\boldsymbol{x}, \dot{\boldsymbol{x}})$ [9], [11]. While higher order systems seem feasible, their implementation on robots is much more challenging, as higher order configuration space derivatives would be required. The trajectory generator is defined by the differential equation $\boldsymbol{M}\ddot{\boldsymbol{x}} + \boldsymbol{f} = 0$, where $\boldsymbol{M}(\boldsymbol{x}, \dot{\boldsymbol{x}})$ and $\boldsymbol{f}(\boldsymbol{x}, \dot{\boldsymbol{x}})$ are functions of position and velocity. Besides, $\boldsymbol{M}$ is symmetric and invertible. Such systems $\mathcal{S} = (\boldsymbol{M}, \boldsymbol{f})_{\mathcal{X}}$ are known as *spectral semisprays*, or *specs* for short. When the space of the task variable is clear from the context, we drop the subscript. Then, the trajectory is computed as the solution to the system $\ddot{\boldsymbol{x}} = -\boldsymbol{M}^{-1}\boldsymbol{f}$.

D. Operations on Specs

Next, the two fundamental operations for specs, transformation between spaces and summation, are introduced.

Given a differential map $\phi : \mathcal{Q} \to \mathcal{X}$ and a spec $(\boldsymbol{M}, \boldsymbol{f})_{\mathcal{X}}$, the *pullback* is defined as

$$\text{pull}_\phi(\boldsymbol{M}, \boldsymbol{f})_{\mathcal{X}} = \left(\boldsymbol{J}_\phi^T \boldsymbol{M} \boldsymbol{J}_\phi, \boldsymbol{J}_\phi^T(\boldsymbol{f} + \boldsymbol{M}\dot{\boldsymbol{J}}_\phi \dot{\boldsymbol{q}})\right)_{\mathcal{Q}}. \tag{4}$$

The pullback allows converting between two distinct manifolds (e.g., a spec could be defined in the robot's workspace and being pulled into the robot's configuration space using the pullback with ϕ being the forward kinematics).

For two specs, $\mathcal{S}_1 = (\boldsymbol{M}_1, \boldsymbol{f}_1)_{\mathcal{X}}$ and $\mathcal{S}_2 = (\boldsymbol{M}_2, \boldsymbol{f}_2)_{\mathcal{X}}$, their *summation* is defined by

$$\mathcal{S}_1 + \mathcal{S}_2 = (\boldsymbol{M}_1 + \boldsymbol{M}_2, \boldsymbol{f}_1 + \boldsymbol{f}_2)_{\mathcal{X}}. \tag{5}$$

E. Optimization Fabrics

Optimization fabrics form a special class of specs, and thus they inherit their properties, specifically the previously defined operations of *summation* and *pullback*. First, let us introduce a finite and differentiable potential function $\psi(\boldsymbol{x})$ defined in a task manifold $\mathcal{X}$. Then, the modified spec $\mathcal{S}_\psi = (\boldsymbol{M}, \boldsymbol{f} + \partial_{\boldsymbol{x}}\psi)$ is called the *forced variant* of $\mathcal{S} = (\boldsymbol{M}, \boldsymbol{f})_{\mathcal{X}}$. Only if the trajectory $\boldsymbol{x}(t)$ generated by the forced spec converges to the minimum of ψ, the spec is said to form an *optimization fabric*. When the spec only converges to the minimum when equipped with a damping term, $(\boldsymbol{M}, \boldsymbol{f} + \partial_{\boldsymbol{x}}\psi + \boldsymbol{B}\dot{\boldsymbol{x}})$, it forms a *frictionless fabric* [9, Def. 4.4]. Note that the mechanical system of a pendulum forms a frictionless fabric, as it optimizes the potential function defined by gravity when being damped (i.e., it eventually comes to rest at the configuration with minimal potential energy). In the following, methods to construct optimization fabrics, or *fabrics* for short, are summarized: the definitions of conservative fabrics and energization are introduced.

F. Conservative Fabrics and Energization

While the previous section defined what criteria are required for a spec to form an optimization fabric, the theory on conservative fabrics and energization offers a simple way of generating such special specs. As a full summary of the theory on optimization fabrics and their construction is out of scope here, this section only provides an outline of the theory and the reader is referred to [10] and [30] for detailed derivations.

In the context of fabrics, the term *energy* describes a scalar quantity that changes as the system evolves over time. Although this quantity has a physical meaning in natural systems (e.g., kinetic energy), it can be arbitrarily defined for motion generation. Generally, specs and optimization fabrics do not conserve an energy, but when they do, we call them *conservative specs*. A stationary Lagrangian [9, Def. 4.11] is one definition for an energy for which the corresponding spec, known as the Lagrangian spec $\mathcal{S}_{\mathcal{L}_e} = (\boldsymbol{M}_{\mathcal{L}_e}, \boldsymbol{f}_{\mathcal{L}_e})$, is obtained by applying the Euler–Lagrange equations. Importantly, Lagrangian specs conserve energy and do, thus, belong to the class of *conservative specs*. It was proven that an unbiased (see [9, Def. 4.11]) Lagrangian spec forms a frictionless fabric [9, Prop. 4.18]. Such fabrics are analogously called *conservative fabrics*. There are two classes of conservative fabrics: Lagrangian fabrics (i.e., the defining energy is a Lagrangian) and the more specific subclass of Finsler fabrics (i.e., the defining energy is a Finsler structure [9, Def. 5.4]).

The operation of *energization* transforms a given differential equation into a conservative spec. Specifically, given an unbiased energy Lagrangian $\mathcal{L}_e$ with boundary conforming $\boldsymbol{M}_{\mathcal{L}_e}$ [9, Def. 4.6] and lower bounded energy $\mathcal{H}_e$, an unbiased spec of form $\mathcal{S}_{\boldsymbol{h}} = (\boldsymbol{I}, \boldsymbol{h})$ is transformed into a frictionless fabric using

energization as

$$\begin{aligned} S_{\boldsymbol{h}}^{\mathcal{L}_e} &= \text{energize}_{\mathcal{L}_e}\{S_{\boldsymbol{h}}\} \\ &= (\boldsymbol{M}_{\mathcal{L}_e}, \boldsymbol{f}_{\mathcal{L}_e} + \boldsymbol{P}_{\mathcal{L}_e}[\boldsymbol{M}_{\mathcal{L}_e}\boldsymbol{h} - \boldsymbol{f}_{\mathcal{L}_e}]) \end{aligned} \tag{6}$$

where $\boldsymbol{P}_{\mathcal{L}_e} = \boldsymbol{M}_{\mathcal{L}_e}\left(\boldsymbol{M}_{\mathcal{L}_e}^{-1} - \frac{\dot{\boldsymbol{x}}\dot{\boldsymbol{x}}^T}{\dot{\boldsymbol{x}}^T\boldsymbol{M}_{\mathcal{L}_e}\dot{\boldsymbol{x}}}\right)$ is an orthogonal projector. Energized specs maintain the energy of the Lagrangian and generally change the trajectory of the underlying spec $\mathcal{S}_{\boldsymbol{h}}$. However, if

1) $\mathcal{S}_{\boldsymbol{h}} = (\boldsymbol{I}, \boldsymbol{h})$ is homogeneous of degree 2

$$\boldsymbol{h}(\boldsymbol{x}, \alpha\dot{\boldsymbol{x}}) = \alpha^2\boldsymbol{h}(\boldsymbol{x}, \dot{\boldsymbol{x}}) \tag{7}$$

and

2) the energizing Lagrangian is a Finsler structure,

the resulting energized spec forms a frictionless fabric for which the trajectory matches the original trajectory of $\mathcal{S}_{\boldsymbol{h}}$. We refer to energized fabrics with that property as *geometric fabrics*. Geometric fabrics form the building blocks for motion generation with optimization fabrics. Practically, energization equips the individual components of the planning problem with a metric when being combined with other components.

G. Experimental Results Fabrics

The theory explained previously was tested on several simple kinematic chains in [9] and [30]. As fabrics design motion as a summation of several differential equations, each representing a specific constraint to the motion, it is possible to sequentially design motion [9]. This procedure allows to carefully tune individual components without harming the others. The application to a planar arm in a goal-reaching setup was successfully tested in [9]. Here, the authors illustrated how the resulting motion can be modified arbitrarily by the user by adding additional constraints or preferences.

Although important concepts and findings on optimization fabrics were summarized in this section, we refer to [9] for a more in-depth presentation of optimization fabrics. In the following, we generalize the framework of optimization fabrics to dynamic settings.

IV. Derivation of DF

We extend the framework of optimization fabrics to DF. including dynamic environments and path-following tasks. We prove that DF converge to moving goals and can be combined with previous approaches in geometric control. This section first introduces the notion of reference trajectory, dynamic Lagrangians, and the dynamic pullback. These notations allow then to formulate DF. As DF generalize the concept of optimization fabrics to dynamic scenarios, we refer to the non-DF as static fabrics (SF) to explicitly distinguish between the work presented in [9] and our work.

A. Motion Design Using DF

The method explained in this article generalizes the concept of static fabrics (SF) from [9] and can then be extended from the procedure outlined in Section III-A.

1) Design path-consistent geometries in a suited, *time-parameterized* (see Definition 4.2) manifold of the configuration.
2) Design corresponding Finsler energies defining the importance metric in this manifold.
3) Energize all geometries with the associated Finsler energies.
4) *If necessary, pull back the energized system from the time-parameterized manifold into the corresponding fixed manifold (8).*
5) Pull back the energized system into the configuration space and combine it with all components using summation.
6) Force the system with a *time-parameterized* potential. As a composition of DF, the resulting trajectory converges toward the potential's minimum (see Lemma 4.11).

In the following, we explain our proposed changes to the framework of SF so that it remains valid in dynamic environments.

B. Reference Trajectories

To enable the definition of dynamic convergence and dynamic energy, we introduce a reference trajectory that remains inside a domain $\mathcal{X}$ as *boundary conforming*. This term is chosen in accordance to [9, Def. 4.6].

Definition 4.1: A reference trajectory $\tilde{\boldsymbol{x}}(t)$, with its corresponding time derivatives $\dot{\tilde{\boldsymbol{x}}}$ and $\ddot{\tilde{\boldsymbol{x}}}$, is boundary conforming on the manifold $\mathcal{X}$ if $\tilde{\boldsymbol{x}}(t) \in \mathcal{X}\ \forall\, t$.

In the following, the reference trajectory will be used to define dynamic Lagrangians and DF. In this context, the word "dynamic" can often be read as "relative to the reference trajectory". With the notion of reference trajectories, we formulate a mapping to the relative coordinate system.

Definition 4.2: Given a reference trajectory $\tilde{\boldsymbol{x}}$ on $\mathcal{X}$, the dynamic mapping $\phi_d : \mathcal{X} \times \mathcal{X} \to \mathcal{X}_{\text{rel}}$ represents the relative coordinate system $\boldsymbol{x}_{\text{rel}} = \boldsymbol{x} - \tilde{\boldsymbol{x}}$.

C. Dynamic Pullback

The theory of optimization fabrics also applies to relative coordinates $\boldsymbol{x}_{\text{rel}}$, specifically, specs and potentials can be formulated in moving coordinates. However, there is no theory to combine specs defined in relative coordinates with specs in fixed coordinates. In most cases, individual components of the behavior design are not formulated in the same relative coordinates. Specifically, the configuration space is always static, so we introduce a transformation of a relative spec into the static space $\mathcal{X}$. We call this operation *dynamic pullback*

$$\text{pull}_{\phi_d}(\boldsymbol{M}_d, \boldsymbol{f}_d)_{\mathcal{X}_{\text{rel}}} = (\boldsymbol{M}_d, \boldsymbol{f}_d - \boldsymbol{M}_d\ddot{\tilde{\boldsymbol{x}}})_{\mathcal{X}}. \tag{8}$$

Two specs $\mathcal{S}_{\mathcal{X}_{\text{rel},1}}$ and $\mathcal{S}_{\mathcal{X}_{\text{rel},2}}$ defined in two different relative coordinate systems are then combined by first applying the dynamic pullback to both individually and then applying the summation operation for specs. The dynamic pullback is the natural extension to optimization fabrics for relative coordinate systems. It cannot directly be integrated into the framework of optimization fabrics as it breaks the algebra. In the following,

we derive several generalizations so that the theory remains valid even in the presence of reference trajectories for individual components, such as moving obstacles or reference trajectories.

D. Dynamic Lagrangians

Next, we show that energy conservation commutes with the dynamic pullback. This allows us to transfer findings on conservative fabrics to DF. We call a Lagrangian that is defined using relative coordinates a *dynamic Lagrangian* and write $\mathcal{L}_d(\boldsymbol{x}_{\text{rel}}, \dot{\boldsymbol{x}}_{\text{rel}})$. In this relative coordinate system, the dynamic Lagrangian has the same properties as the Lagrangian defined in [9], specifically it induces the Lagrangian spec through the Euler–Lagrange equation, $\partial^2_{\dot{\boldsymbol{x}}_{\text{rel}}\dot{\boldsymbol{x}}_{\text{rel}}}\mathcal{L}_d\ddot{\boldsymbol{x}}_{\text{rel}} + \partial^2_{\dot{\boldsymbol{x}}_{\text{rel}}\boldsymbol{x}_{\text{rel}}}\mathcal{L}_d\dot{\boldsymbol{x}}_{\text{rel}} - \partial_{\boldsymbol{x}_{\text{rel}}}\mathcal{L}_d$, as $(\boldsymbol{M}_{de}, \boldsymbol{f}_{de})$. The system's Hamiltonian $\mathcal{H}_d = \partial_{\dot{\boldsymbol{x}}_{\text{rel}}}\mathcal{L}_d^T\dot{\boldsymbol{x}}_{\text{rel}} - \mathcal{L}_d$ is conserved by the equation of motion, as proven in [9].

Applying the dynamic pullback to the dynamic Lagrangian, we obtain the transformed Lagrangian $\mathcal{L}_d(\boldsymbol{x}, \dot{\boldsymbol{x}}, \tilde{\boldsymbol{x}}, \dot{\tilde{\boldsymbol{x}}})$ in the static coordinate system.

Theorem 4.3: Let $\mathcal{L}_d(\boldsymbol{x}_{\text{rel}}, \dot{\boldsymbol{x}}_{\text{rel}})$ be a dynamic Lagrangian and let ϕ_d be the dynamic mapping to $\boldsymbol{x}_{\text{rel}}$. Then, the application of the Euler–Lagrange equation commutes with the dynamic pullback.

Proof: We will show the equivalence by calculation. As abovementioned, the induced spec is defined in the relative system as $(\boldsymbol{M}_{de}, \boldsymbol{f}_{de})$. It can be dynamically pulled to form

$$\text{pull}_{\phi_d}(\boldsymbol{M}_{de}, \boldsymbol{f}_{de})_{\mathcal{X}_{\text{rel}}} = (\boldsymbol{M}_{de}, \boldsymbol{f}_{de} - \boldsymbol{M}_{de}\ddot{\tilde{\boldsymbol{x}}})_{\mathcal{X}}. \tag{9}$$

We can dynamically pull the Lagrangian $\mathcal{L}_d(\boldsymbol{x}_{\text{rel}}, \dot{\boldsymbol{x}}_{\text{rel}})$ to form $\mathcal{L}_d(\boldsymbol{x}, \dot{\boldsymbol{x}}, \tilde{\boldsymbol{x}}, \dot{\tilde{\boldsymbol{x}}})$, where only the first two variables are system variables. Using the generalized Euler–Lagrange equation, the equations of motion of the pulled Lagrangian are obtained as

$$\begin{aligned}
0 &= \frac{d}{dt}\frac{\partial \mathcal{L}_d}{\partial \dot{\boldsymbol{x}}} - \frac{\partial \mathcal{L}_d}{\partial \boldsymbol{x}} \\
&= \frac{\partial^2 \mathcal{L}_d}{\partial \dot{\boldsymbol{x}}\partial \dot{\boldsymbol{x}}}\ddot{\boldsymbol{x}} + \frac{\partial^2 \mathcal{L}_d}{\partial \dot{\boldsymbol{x}}\partial \boldsymbol{x}}\dot{\boldsymbol{x}} + \frac{\partial^2 \mathcal{L}_d}{\partial \dot{\boldsymbol{x}}\partial \dot{\tilde{\boldsymbol{x}}}}\ddot{\tilde{\boldsymbol{x}}} + \frac{\partial^2 \mathcal{L}_d}{\partial \dot{\boldsymbol{x}}\partial \tilde{\boldsymbol{x}}}\dot{\tilde{\boldsymbol{x}}} - \frac{\partial \mathcal{L}_d}{\partial \boldsymbol{x}} \\
&= \frac{\partial^2 \mathcal{L}_d}{\partial \dot{\boldsymbol{x}}_{\text{rel}}\partial \dot{\boldsymbol{x}}_{\text{rel}}}\frac{\partial \dot{\boldsymbol{x}}_{\text{rel}}}{\partial \dot{\boldsymbol{x}}}\frac{\partial \dot{\boldsymbol{x}}_{\text{rel}}}{\partial \dot{\boldsymbol{x}}}\ddot{\boldsymbol{x}} + \frac{\partial^2 \mathcal{L}_d}{\partial \dot{\boldsymbol{x}}_{\text{rel}}\partial \boldsymbol{x}_{\text{rel}}}\frac{\partial \dot{\boldsymbol{x}}_{\text{rel}}}{\partial \dot{\boldsymbol{x}}}\frac{\partial \boldsymbol{x}_{\text{rel}}}{\partial \boldsymbol{x}}\dot{\boldsymbol{x}} \\
&\quad + \frac{\partial^2 \mathcal{L}_d}{\partial \dot{\boldsymbol{x}}_{\text{rel}}\partial \dot{\boldsymbol{x}}_{\text{rel}}}\frac{\partial \dot{\boldsymbol{x}}_{\text{rel}}}{\partial \dot{\boldsymbol{x}}}\frac{\partial \dot{\boldsymbol{x}}_{\text{rel}}}{\partial \dot{\tilde{\boldsymbol{x}}}}\ddot{\tilde{\boldsymbol{x}}} + \frac{\partial^2 \mathcal{L}_d}{\partial \dot{\boldsymbol{x}}_{\text{rel}}\partial \boldsymbol{x}_{\text{rel}}}\frac{\partial \dot{\boldsymbol{x}}_{\text{rel}}}{\partial \dot{\boldsymbol{x}}}\frac{\partial \boldsymbol{x}_{\text{rel}}}{\partial \tilde{\boldsymbol{x}}}\dot{\tilde{\boldsymbol{x}}} \\
&\quad - \frac{\partial \mathcal{L}_d}{\partial \boldsymbol{x}_{\text{rel}}}\frac{\partial \boldsymbol{x}_{\text{rel}}}{\partial \boldsymbol{x}} \\
&= \frac{\partial^2 \mathcal{L}_d}{\partial \dot{\boldsymbol{x}}_{\text{rel}}\partial \dot{\boldsymbol{x}}_{\text{rel}}}\ddot{\boldsymbol{x}} + \frac{\partial^2 \mathcal{L}_d}{\partial \dot{\boldsymbol{x}}_{\text{rel}}\partial \boldsymbol{x}_{\text{rel}}}\dot{\boldsymbol{x}} - \frac{\partial^2 \mathcal{L}_d}{\partial \dot{\boldsymbol{x}}_{\text{rel}}\partial \dot{\boldsymbol{x}}_{\text{rel}}}\ddot{\tilde{\boldsymbol{x}}} \\
&\quad - \frac{\partial^2 \mathcal{L}_d}{\partial \dot{\boldsymbol{x}}_{\text{rel}}\partial \boldsymbol{x}_{\text{rel}}}\dot{\tilde{\boldsymbol{x}}} - \frac{\partial \mathcal{L}_d}{\partial \boldsymbol{x}_{\text{rel}}} \\
&= \frac{\partial^2 \mathcal{L}_d}{\partial \dot{\boldsymbol{x}}_{\text{rel}}\partial \dot{\boldsymbol{x}}_{\text{rel}}}(\ddot{\boldsymbol{x}} - \ddot{\tilde{\boldsymbol{x}}}) + \frac{\partial^2 \mathcal{L}_d}{\partial \dot{\boldsymbol{x}}_{\text{rel}}\partial \boldsymbol{x}_{\text{rel}}}(\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}}) - \frac{\partial \mathcal{L}_d}{\partial \boldsymbol{x}_{\text{rel}}} \\
&= \frac{\partial^2 \mathcal{L}_d}{\partial \dot{\boldsymbol{x}}_{\text{rel}}\partial \dot{\boldsymbol{x}}_{\text{rel}}}(\ddot{\boldsymbol{x}} - \ddot{\tilde{\boldsymbol{x}}}) + \frac{\partial^2 \mathcal{L}_d}{\partial \dot{\boldsymbol{x}}_{\text{rel}}\partial \boldsymbol{x}_{\text{rel}}}(\dot{\boldsymbol{x}}_{\text{rel}}) - \partial_{\boldsymbol{x}_{\text{rel}}}\mathcal{L}_d \\
&= \boldsymbol{M}_{de}\ddot{\boldsymbol{x}} + \boldsymbol{f}_{de} - \boldsymbol{M}_{de}\ddot{\tilde{\boldsymbol{x}}}.
\end{aligned}$$

The obtained equations of motion match the one obtained by applying the dynamic pullback, see (9). ■

Hence, independently of the coordinates, the system conserves the energy $\mathcal{H}_d$ computed with the Hamiltonion in relative coordinates. Next, we adapt the operation of energization to dynamic Lagrangians. Dynamic Lagrangians are a necessary step to allow for collision avoidance with dynamic obstacles in the framework of optimization fabrics. Specifically, the metric for a moving obstacle is computed using the Euler–Lagrange equation in the relative coordinate system. Importantly, in this system, the same energies as with SF can be employed. Using the dynamic pullback, the energy defining the metric for the moving obstacle is then maintained according to Theorem IV.3. Concretely, this means that collision avoidance can be achieved in a similar manner as with SF with the added advantage of integrated motion estimates of obstacles.

Proposition 4.4 (Dynamic energization): Let $\ddot{\boldsymbol{x}} + \boldsymbol{h}(\boldsymbol{x}, \dot{\boldsymbol{x}}) = \boldsymbol{0}$ be a differential equation, and suppose $\mathcal{L}_d$ is a dynamic Lagrangian with the induced spec $(\boldsymbol{M}_{de}, \boldsymbol{f}_{de})$ and dynamic energy $\mathcal{H}_d$. Then, the dynamically energized system $\ddot{\boldsymbol{x}} + \boldsymbol{h}(\boldsymbol{x}, \dot{\boldsymbol{x}}) + \alpha_{\mathcal{H}_d}\dot{\boldsymbol{x}}_{\text{rel}} = \boldsymbol{0}$ with

$$\alpha_{\mathcal{H}_d} = -(\dot{\boldsymbol{x}}_{\text{rel}}^T\boldsymbol{M}_{de}\dot{\boldsymbol{x}}_{\text{rel}})^{-1}\dot{\boldsymbol{x}}_{\text{rel}}^T(\boldsymbol{M}_{de}(\boldsymbol{h} + \ddot{\tilde{\boldsymbol{x}}}) - \boldsymbol{f}_{de})$$

conserves the dynamic energy $\mathcal{H}_d$.

Proof: From the derivations in [9], we can compute the rate of change of the dynamic energy as $\dot{\mathcal{H}}_d = \dot{\boldsymbol{x}}_{\text{rel}}^T(\boldsymbol{M}_{de}\ddot{\boldsymbol{x}}_{\text{rel}} + \boldsymbol{f}_{de})$. The equations of motion can be plugged in through the definition of the reference trajectory Definition 4.1, $\ddot{\boldsymbol{x}}_{\text{rel}} = \ddot{\boldsymbol{x}} - \ddot{\tilde{\boldsymbol{x}}}$ to obtain

$$\begin{aligned}
\dot{\mathcal{H}}_d &= \dot{\boldsymbol{x}}_{\text{rel}}^T(\boldsymbol{M}_{de}(-\boldsymbol{h} - \alpha_{\mathcal{H}_d}\dot{\boldsymbol{x}}_{\text{rel}} - \ddot{\tilde{\boldsymbol{x}}}) + \boldsymbol{f}_{de}) \\
&= \dot{\boldsymbol{x}}_{\text{rel}}^T(-\boldsymbol{M}_{de}\boldsymbol{h} - \boldsymbol{M}_{de}\dot{\boldsymbol{x}}_{\text{rel}}\alpha_{\mathcal{H}_d} - \boldsymbol{M}_{de}\ddot{\tilde{\boldsymbol{x}}} + \boldsymbol{f}_{de}) \\
&= \dot{\boldsymbol{x}}_{\text{rel}}^T(-\boldsymbol{M}_{de}\boldsymbol{h} \\
&\quad + \boldsymbol{M}_{de}\dot{\boldsymbol{x}}_{\text{rel}}(\dot{\boldsymbol{x}}_{\text{rel}}^T\boldsymbol{M}_{de}\dot{\boldsymbol{x}}_{\text{rel}})^{-1}\dot{\boldsymbol{x}}_{\text{rel}}^T(\boldsymbol{M}_{de}(\boldsymbol{h} + \ddot{\tilde{\boldsymbol{x}}}) - \boldsymbol{f}_{de}) \\
&\quad - \boldsymbol{M}_{de}\ddot{\tilde{\boldsymbol{x}}} + \boldsymbol{f}_{de}) \\
&= -\dot{\boldsymbol{x}}_{\text{rel}}^T\boldsymbol{M}_{de}\boldsymbol{h} + \dot{\boldsymbol{x}}_{\text{rel}}^T(\boldsymbol{M}_{de}(\boldsymbol{h} + \ddot{\tilde{\boldsymbol{x}}}) - \boldsymbol{f}_{de}) \\
&\quad - \dot{\boldsymbol{x}}_{\text{rel}}^T\boldsymbol{M}_{de}\ddot{\tilde{\boldsymbol{x}}} + \dot{\boldsymbol{x}}_{\text{rel}}^T\boldsymbol{f}_{de} \\
&= 0.
\end{aligned}$$

The energized system conserves the dynamic energy. ■

Proposition 4.4 allows to combine dynamic components of the motion generator with static components. Effectively, the dynamic component *bends* the underlying geometry according to the motion of the dynamic components (e.g., a moving obstacle).

While dynamic Lagrangians and the corresponding energization operation are similar to the methods described in [9], the operation of the standard pull to the dynamically energized system must be slightly modified. Specifically, the reference velocity must be pulled. We show that dynamic energization also commutes with the standard pullback.

Theorem 4.5: Let $\mathcal{L}_d$ be a dynamic Lagrangian to the reference trajectory $\tilde{\boldsymbol{x}}$, and let $\ddot{\boldsymbol{x}} + \boldsymbol{h}(\boldsymbol{x}, \dot{\boldsymbol{x}}) = \boldsymbol{0}$ be a second-order differential equation with a metric $\boldsymbol{M}_d$ such that $\boldsymbol{J}_\phi^T\boldsymbol{M}_d\boldsymbol{J}_\phi$ has full rank that can be written as spec $(\boldsymbol{M}_d, \boldsymbol{M}_d\boldsymbol{h})$. Suppose

$x = \phi(\boldsymbol{q})$ is a differential map with $\boldsymbol{J}_\phi$ its Jacobian. Then,

$$\text{energize}_{\text{pull}_\phi \mathcal{L}_d}\big(\text{pull}_\phi(\boldsymbol{I}, \boldsymbol{h})\big) = \text{pull}_\phi\big(\text{energize}_{\mathcal{L}_d}(\boldsymbol{I}, \boldsymbol{h})\big) \quad (10)$$

when the reference velocity is being pulled as $\dot{\tilde{q}} = \boldsymbol{J}_\phi^\dagger \dot{\tilde{\boldsymbol{x}}}$. $\boldsymbol{J}_\phi^\dagger$ denotes the pseudoinverse of $\boldsymbol{J}_\phi$. We say that the dynamic energization operation commutes with the pullback transform.

Proof: The commutation can be proven by calculation. First, we compute the right side of the equivalence. According to Proposition 4.4, the energized system (that maintains the dynamic energy $\mathcal{H}_d$) writes as

$$\boldsymbol{M}_d \ddot{\boldsymbol{x}} + \boldsymbol{M}_d \boldsymbol{h} + \alpha_{\mathcal{H}_d}(\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}}) = 0$$

with $\alpha_{\mathcal{H}_d}$ as defined in Proposition 4.4. Applying the pull-operation, we obtain

$$\begin{aligned} &\boldsymbol{J}_\phi^T \boldsymbol{M}_d \boldsymbol{J}_\phi \ddot{\boldsymbol{q}} + \boldsymbol{J}_\phi^T \boldsymbol{M}_d \boldsymbol{h} + \boldsymbol{J}_\phi^T \boldsymbol{M}_d \dot{\boldsymbol{J}}_\phi \dot{\boldsymbol{q}} \\ &\quad + \boldsymbol{J}_\phi^T \boldsymbol{M}_d \alpha_{\mathcal{H}_d}(\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}}) = 0. \end{aligned} \quad (11)$$

As the equation expressed in $\mathcal{X}$, this equation in $\mathcal{Q}$ maintains the energy $\mathcal{H}_d$. Next, we compute the left-hand side. The equation of motion of the pulled dynamic Lagrangian $\mathcal{L}_d$ computes as

$$\begin{aligned} \text{pull}_\phi(\boldsymbol{M}_d, \boldsymbol{f}_d) &= \boldsymbol{J}_\phi^T \left(\boldsymbol{M}_d \boldsymbol{J}_\phi \ddot{\boldsymbol{q}} + \boldsymbol{f}_d + \boldsymbol{M}_d \dot{\boldsymbol{J}}_\phi \dot{\boldsymbol{q}} - \boldsymbol{M}_d \ddot{\tilde{\boldsymbol{x}}} \right) \\ &= \tilde{\boldsymbol{M}}_d \ddot{\boldsymbol{q}} + \tilde{\boldsymbol{f}}_d - \boldsymbol{J}_\phi^T \boldsymbol{M}_d \ddot{\tilde{\boldsymbol{x}}}. \end{aligned}$$

The original spec is pulled accordingly

$$\begin{aligned} \text{pull}_\phi(\boldsymbol{M}_d, \boldsymbol{M}_d \boldsymbol{h}) &= \boldsymbol{J}_\phi^T \boldsymbol{M}_d \boldsymbol{J}_\phi \ddot{\boldsymbol{q}} + \boldsymbol{J}_\phi^T \boldsymbol{M}_d \boldsymbol{h} + \boldsymbol{J}_\phi^T \boldsymbol{M}_d \dot{\boldsymbol{J}}_\phi \dot{\boldsymbol{q}} \\ &= \tilde{\boldsymbol{M}}_d \ddot{\boldsymbol{q}} + \tilde{\boldsymbol{M}}_d \tilde{\boldsymbol{h}}. \end{aligned}$$

We energize the pulled system according to Proposition IV.4

$$\begin{aligned} &\boldsymbol{J}_\phi^T \boldsymbol{M}_d \boldsymbol{J}_\phi \ddot{\boldsymbol{q}} + \boldsymbol{J}_\phi^T \boldsymbol{M}_d \boldsymbol{h} + \boldsymbol{J}_\phi^T \boldsymbol{M}_d \dot{\boldsymbol{J}}_\phi \dot{\boldsymbol{q}} \\ &\quad + \boldsymbol{J}_\phi^T \boldsymbol{M}_d \boldsymbol{J}_\phi \alpha_{\text{pull}_\phi \mathcal{H}_d}(\dot{\boldsymbol{q}} - \boldsymbol{J}_\phi^\dagger \dot{\boldsymbol{x}}) = 0 \end{aligned} \quad (12)$$

with

$$\begin{aligned} \alpha_{\text{pull}_\phi \mathcal{H}_d} &= -\Big((\dot{\boldsymbol{q}} - \boldsymbol{J}_\phi^\dagger \dot{\tilde{\boldsymbol{x}}})^T \boldsymbol{J}_\phi^T \boldsymbol{M}_d \boldsymbol{J}_\phi (\dot{\boldsymbol{q}} - \boldsymbol{J}_\phi^\dagger \dot{\tilde{\boldsymbol{x}}}) \Big)^{-1} \\ &\quad (\dot{\boldsymbol{q}} - \boldsymbol{J}_\phi^\dagger \dot{\tilde{\boldsymbol{x}}})^T \Big(\boldsymbol{J}_\phi^T \boldsymbol{M}_d \boldsymbol{J}_\phi (\tilde{\boldsymbol{h}} + \ddot{\tilde{\boldsymbol{x}}}) \\ &\quad - \boldsymbol{J}_\phi^T \boldsymbol{f}_d - \boldsymbol{J}_\phi^T \boldsymbol{M}_d \dot{\boldsymbol{J}}_\phi \dot{\boldsymbol{q}} \Big) \\ &= -\Big((\boldsymbol{J}_\phi \dot{\boldsymbol{q}} - \boldsymbol{J}_\phi \boldsymbol{J}_\phi^\dagger \dot{\tilde{\boldsymbol{x}}})^T \boldsymbol{M}_d (\boldsymbol{J}_\phi \dot{\boldsymbol{q}} - \boldsymbol{J}_\phi \boldsymbol{J}_\phi^\dagger \dot{\tilde{\boldsymbol{x}}}) \Big)^{-1} \\ &\quad (\dot{\boldsymbol{q}} - \boldsymbol{J}_\phi^\dagger \dot{\tilde{\boldsymbol{x}}})^T \Big(\boldsymbol{J}_\phi^T \boldsymbol{M}_d \boldsymbol{J}_\phi \tilde{\boldsymbol{h}} + \boldsymbol{J}_\phi^T \boldsymbol{M}_d \ddot{\tilde{\boldsymbol{x}}} \\ &\quad - \boldsymbol{J}_\phi^T \boldsymbol{f}_d - \boldsymbol{J}_\phi^T \boldsymbol{M}_d \dot{\boldsymbol{J}}_\phi \dot{\boldsymbol{q}} \Big) \\ &= -\Big((\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}})^T \boldsymbol{M}_d (\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}}) \Big)^{-1} \\ &\quad (\dot{\boldsymbol{q}} - \boldsymbol{J}_\phi^\dagger \dot{\tilde{\boldsymbol{x}}})^T \Big(\boldsymbol{J}_\phi^T \boldsymbol{M}_d \boldsymbol{h} + \boldsymbol{J}_\phi^T \boldsymbol{M}_d \dot{\boldsymbol{J}}_\phi \dot{\boldsymbol{q}} + \boldsymbol{J}_\phi^T \boldsymbol{M}_d \ddot{\tilde{\boldsymbol{x}}} \\ &\quad - \boldsymbol{J}_\phi^T \boldsymbol{f}_d - \boldsymbol{J}_\phi^T \boldsymbol{M}_d \dot{\boldsymbol{J}}_\phi \dot{\boldsymbol{q}} \Big) \\ &= -\Big((\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}})^T \boldsymbol{M}_d (\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}}) \Big)^{-1} \\ &\quad (\dot{\boldsymbol{q}} - \boldsymbol{J}_\phi^\dagger \dot{\tilde{\boldsymbol{x}}})^T \Big(\boldsymbol{J}_\phi^T \boldsymbol{M}_d \boldsymbol{h} + \boldsymbol{J}_\phi^T \boldsymbol{M}_d \ddot{\tilde{\boldsymbol{x}}} - \boldsymbol{J}_\phi^T \boldsymbol{f}_d \Big) \\ &= -\Big((\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}})^T \boldsymbol{M}_d (\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}}) \Big)^{-1} \\ &\quad (\boldsymbol{J}_\phi \dot{\boldsymbol{q}} - \boldsymbol{J}_\phi \boldsymbol{J}_\phi^\dagger \dot{\tilde{\boldsymbol{x}}})^T \Big(\boldsymbol{M}_d \boldsymbol{h} + \boldsymbol{M}_d \ddot{\tilde{\boldsymbol{x}}} - \boldsymbol{f}_d \Big) \\ &= -\Big((\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}})^T \boldsymbol{M}_d (\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}}) \Big)^{-1} (\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}})^T \\ &\quad \Big(\boldsymbol{M}_d \boldsymbol{h} + \boldsymbol{M}_d \ddot{\tilde{\boldsymbol{x}}} - \boldsymbol{f}_d \Big) \\ &= \alpha_{\mathcal{H}_d}. \end{aligned}$$

Thus, we have shown equivalence between $\alpha_{\mathcal{H}_d}$ and $\alpha_{\text{pull}_\phi \mathcal{H}_d}$. As α is scalar, we can rewrite the energization term in (12) as

$$\begin{aligned} &\boldsymbol{J}_\phi^T \boldsymbol{M}_d \boldsymbol{J}_\phi \alpha_{\text{pull}_\phi \mathcal{H}_d}(\dot{\boldsymbol{q}} - \boldsymbol{J}_\phi^\dagger \dot{\boldsymbol{x}}) \\ &= \boldsymbol{J}_\phi^T \boldsymbol{M}_d \alpha_{\text{pull}_\phi \mathcal{H}_d}(\boldsymbol{J}_\phi \dot{\boldsymbol{q}} - \boldsymbol{J}_\phi \boldsymbol{J}_\phi^\dagger \dot{\boldsymbol{x}}) \\ &= \boldsymbol{J}_\phi^T \boldsymbol{M}_d \alpha_{\text{pull}_\phi \mathcal{H}_d}(\dot{\boldsymbol{x}} - \dot{\boldsymbol{x}}) \\ &= \boldsymbol{J}_\phi^T \boldsymbol{M}_d \alpha_{\mathcal{H}_d}(\dot{\boldsymbol{x}} - \dot{\boldsymbol{x}}). \end{aligned}$$

With the equivalence of the energization terms, we conclude the proof that dynamic energization commutes with the standard pullback. ■

E. Dynamic Fabrics

With the previous results, we formulate a new class of fabrics that converge to a reference trajectory. We call this class of fabrics DF. First, some notations are introduced to eventually show that dynamically energized specs form DF. Analogously to unbiased specs, we define dynamically unbiased specs (i.e., specs whose solutions do not diverge from the reference $\tilde{\boldsymbol{x}}$ when starting on the reference).

Definition 4.6: A spec is said to be *dynamically unbiased* with respect to $\tilde{\boldsymbol{x}}(t)$ if $\boldsymbol{f}(\boldsymbol{x}, \dot{\boldsymbol{x}}) = -\boldsymbol{M}\ddot{\tilde{\boldsymbol{x}}}$, for $\boldsymbol{x}(t) = \tilde{\boldsymbol{x}}(t)$ and $\dot{\boldsymbol{x}}(t) = \dot{\tilde{\boldsymbol{x}}}(t)$.

Besides being dynamically unbiased, some specs will converge to the reference trajectory independently from their initial conditions.

Definition 4.7: A spec is *dynamically rough* with respect to $\tilde{\boldsymbol{x}}(t)$ if all its integral curves $\boldsymbol{x}(t)$ converge dynamically with respect to $\tilde{\boldsymbol{x}}(t)$.

As for SF, DF can be formed by specs when they are being forced by a potential function ψ. Such a forcing potential is generally a function of $\boldsymbol{x}$ and $\tilde{\boldsymbol{x}}$ and has at least one minimum. A spec that converges to a minimum of the forcing potential then forms a DF.

Definition 4.8: A spec forms a *dynamically rough fabric* if it is dynamically rough with respect to $\tilde{\boldsymbol{x}}(t)$ when forced by a dynamic potential, and $\exists t_1 > 0$ such that $\forall t > t_1, \boldsymbol{x}(t)$ satisfies the Karush–Kuhn–Tucker (KKT) conditions for the optimization problem $\min_{\boldsymbol{x} \in \mathcal{X}} \psi(\boldsymbol{x}, \tilde{\boldsymbol{x}}(t))$. If a spec does not form a dynamically rough fabric but all its damped variants do, it forms a *dynamically frictionless fabric*.

Theorem 4.9 (DF): Suppose $S = (\boldsymbol{M}, \boldsymbol{f})_{\mathcal{X}}$ is a spec. Then, S forms a dynamically rough fabric with respect to $\tilde{\boldsymbol{x}}$ if and only if it is dynamically unbiased with respect to $\tilde{\boldsymbol{x}}$ and it converges

dynamically when being forced by a dynamic potential $\psi(\boldsymbol{x}, \tilde{\boldsymbol{x}})$ with $\|\partial_{\boldsymbol{x}}\psi\| < \infty$ on $\mathcal{X}$.

Proof: We can write the corresponding differential equation

$$\boldsymbol{M}\ddot{\boldsymbol{x}} + \boldsymbol{f} = -\partial_{\boldsymbol{x}}\psi. \tag{13}$$

Assume that S is dynamically unbiased. Since the spec converges with respect to $\tilde{\boldsymbol{x}}(t)$, we have $\dot{\boldsymbol{x}} \to \dot{\tilde{\boldsymbol{x}}}, \boldsymbol{x} \to \tilde{\boldsymbol{x}}$. Because it is dynamically unbiased, we also have $\boldsymbol{f} \to -\boldsymbol{M}\ddot{\tilde{\boldsymbol{x}}}$. Thus, the left-hand side of (13) approaches $\mathbf{0}$. Consequently, the right-hand side must also approach $\mathbf{0}$, and hence, $\partial_{\boldsymbol{x}}\psi \to \mathbf{0}$. The last satisfies the KKT conditions of ψ.

To prove the converse, assume $\boldsymbol{f}$ dynamically biased. That implies that

$$\exists t > 0, \boldsymbol{f} = \boldsymbol{M}\ddot{\tilde{\boldsymbol{x}}} + \boldsymbol{a}(\tilde{\boldsymbol{x}}, \dot{\tilde{\boldsymbol{x}}}), \boldsymbol{a}(\tilde{\boldsymbol{x}}, \dot{\tilde{\boldsymbol{x}}}) \neq \mathbf{0}.$$

Hence, there exist a $t > 0$ for which the left-hand side does not vanish. As ψ satisfies the KKT conditions at $\boldsymbol{x} = \tilde{\boldsymbol{x}}$, its derivative equals zero at $\boldsymbol{x} = \tilde{\boldsymbol{x}}$, which contradicts (13) with $\boldsymbol{M}\boldsymbol{a}(\tilde{\boldsymbol{x}}, \dot{\tilde{\boldsymbol{x}}}) = \mathbf{0}$. ■

Hence, the spec is required to be unbiased and convergent when forced. While the former can be verified using straightforward computation, convergence is difficult to verify in the general case.

Lemma 4.10 (Dynamically energized fabrics): Suppose S is a dynamically unbiased energized spec. Then, S forms a dynamically frictionless fabric if $\partial_{\boldsymbol{x}}\psi = -\partial_{\tilde{\boldsymbol{x}}}\psi$.

Proof: The equation of motion for the energized, forced, and damped system writes as

$$\ddot{\boldsymbol{x}} + \boldsymbol{h} + \alpha_{\mathcal{H}_d}\dot{\boldsymbol{x}}_{\text{rel}} + \boldsymbol{B}\dot{\boldsymbol{x}}_{\text{rel}} + \partial_{\boldsymbol{x}}\psi = 0. \tag{14}$$

The systems energy (dynamic Hamiltonian) is used as a Lyapunov function to show convergence. The rate of change is computed as

$$\begin{aligned}\dot{\mathcal{H}}_d^{\psi} &= \dot{\boldsymbol{x}}_{\text{rel}}^T(\boldsymbol{M}_{de}(-\boldsymbol{h} - \alpha_{\mathcal{H}_d}\dot{\boldsymbol{x}}_{\text{rel}} - \boldsymbol{B}\dot{\boldsymbol{x}}_{\text{rel}} - \partial_{\boldsymbol{x}}\psi - \ddot{\tilde{\boldsymbol{x}}}) \\ &\quad + \boldsymbol{f}_{de}) + \dot{\psi} \\ &= -\dot{\boldsymbol{x}}_{\text{rel}}^T\boldsymbol{B}\dot{\boldsymbol{x}}_{\text{rel}} - \dot{\boldsymbol{x}}_{\text{rel}}^T\partial_{\boldsymbol{x}}\psi + \dot{\boldsymbol{x}}^T\partial_{\boldsymbol{x}}\psi + \dot{\tilde{\boldsymbol{x}}}^T\partial_{\tilde{\boldsymbol{x}}}\psi \\ &= -\dot{\boldsymbol{x}}_{\text{rel}}^T\boldsymbol{B}\dot{\boldsymbol{x}}_{\text{rel}}.\end{aligned}$$

As the system energy is lower bounded with $\mathcal{H}_d + \psi \geq 0$ and $\dot{\mathcal{H}}_d^{\psi} \leq 0$, when $\boldsymbol{B}$ strictly positive definite, we must have $\dot{\mathcal{H}}_d^{\psi} \to 0$. Thus, $\dot{\boldsymbol{x}}_{\text{rel}}$ goes to zero. We can conclude that the system is dynamically converging. As it also said to be dynamically unbiased, the damped energized system forms a dynamic fabric by Theorem 4.9. ■

Lemma 4.11 (Dynamic Lagrangian fabrics): An unbiased, dynamic Lagrangian spec forms a dynamically frictionless fabric if $\partial_{\boldsymbol{x}}\psi = -\partial_{\tilde{\boldsymbol{x}}}\psi$ holds for the forcing term.

Proof: The equations of motion induced by the dynamic Lagrangian including damping and forcing are defined by the spec and can be written explicitly as

$$\begin{aligned}\boldsymbol{M}_{\mathcal{L}_d}\ddot{\boldsymbol{x}}_{\text{rel}} + \boldsymbol{f}_{\mathcal{L}_d} + \boldsymbol{B}\dot{\boldsymbol{x}}_{\text{rel}} + \partial_{\boldsymbol{x}}\psi &= 0 \\ \boldsymbol{M}_{\mathcal{L}_d}(\ddot{\boldsymbol{x}} - \ddot{\tilde{\boldsymbol{x}}}) + \boldsymbol{f}_{\mathcal{L}_d} + \boldsymbol{B}(\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}}) + \partial_{\boldsymbol{x}}\psi &= 0 \\ \boldsymbol{M}_{\mathcal{L}_d}\ddot{\boldsymbol{x}} - \boldsymbol{M}_{\mathcal{L}_d}\ddot{\tilde{\boldsymbol{x}}} + \boldsymbol{f}_{\mathcal{L}_d} + \boldsymbol{B}\dot{\boldsymbol{x}} - \boldsymbol{B}\dot{\tilde{\boldsymbol{x}}} + \partial_{\boldsymbol{x}}\psi &= 0.\end{aligned} \tag{15}$$

In the following, we use the Hamiltonian and the potential function as Lyapunov function to show convergence of the damped spec:

$$\begin{aligned}\mathcal{H}_d^{\psi}(\boldsymbol{x}) &= \mathcal{H}_d + \psi \\ &= \partial_{\dot{\boldsymbol{x}}_{\text{rel}}}\mathcal{L}_d^T\dot{\boldsymbol{x}}_{\text{rel}} - \mathcal{L}_d + \psi.\end{aligned}$$

The time derivative is composed of the time derivative of the Hamiltonian, $\dot{\mathcal{H}}_e = \dot{\boldsymbol{x}}_{\text{rel}}^T(\boldsymbol{M}_{\mathcal{L}_d}\ddot{\boldsymbol{x}}_{\text{rel}} + \boldsymbol{f}_{\mathcal{L}_d})$, and the time derivative of the forcing potential, $\dot{\psi} = \dot{\boldsymbol{x}}^T\partial_{\boldsymbol{x}}\psi + \dot{\tilde{\boldsymbol{x}}}^T\partial_{\tilde{\boldsymbol{x}}}\psi$. Thus, the system's total energy varies over time

$$\dot{\mathcal{H}}_d{}^{\psi}(\boldsymbol{x}) = (\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}})^T(\boldsymbol{M}_{\mathcal{L}_d}(\ddot{\boldsymbol{x}} - \ddot{\tilde{\boldsymbol{x}}}) + \boldsymbol{f}_{\mathcal{L}_d}) + \dot{\boldsymbol{x}}^T\partial_{\boldsymbol{x}}\psi + \dot{\tilde{\boldsymbol{x}}}^T\partial_{\tilde{\boldsymbol{x}}}\psi.$$

Plugging in the equations of motion (15) gives

$$\begin{aligned}\dot{\mathcal{H}}_d{}^{\psi}(\boldsymbol{x}) &= (\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}})^T(-\boldsymbol{f}_{\mathcal{L}_d} - \boldsymbol{B}(\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}}) - \partial_{\boldsymbol{x}}\psi + \boldsymbol{f}_{\mathcal{L}_d}) \\ &\quad + \dot{\boldsymbol{x}}^T\partial_{\boldsymbol{x}}\psi + \dot{\tilde{\boldsymbol{x}}}^T\partial_{\tilde{\boldsymbol{x}}}\psi \\ &= -(\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}})^T\boldsymbol{B}(\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}}) - (\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}})^T\partial_{\boldsymbol{x}}\psi \\ &\quad + \dot{\boldsymbol{x}}^T\partial_{\boldsymbol{x}}\psi + \dot{\tilde{\boldsymbol{x}}}^T\partial_{\tilde{\boldsymbol{x}}}\psi \\ &= -(\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}})^T\boldsymbol{B}(\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}}) + \dot{\tilde{\boldsymbol{x}}}^T(\partial_{\boldsymbol{x}}\psi + \partial_{\tilde{\boldsymbol{x}}}\psi).\end{aligned}$$

For $\partial_{\boldsymbol{x}}\psi = -\partial_{\tilde{\boldsymbol{x}}}\psi$ and $\boldsymbol{B}$ strictly positive definite, $\dot{\mathcal{H}}_d$ is strictly negative for $(\dot{\boldsymbol{x}} - \dot{\tilde{\boldsymbol{x}}}) \neq 0$. Since $\mathcal{H}_d^{\psi}$ is lower bounded as composition of lower bounded function and $\dot{\mathcal{H}}_d{}^{\psi} \leq 0$, $\dot{\mathcal{H}}_d{}^{\psi} \to 0$, and thus, $\dot{\boldsymbol{x}} \to \dot{\tilde{\boldsymbol{x}}}$ and $\boldsymbol{x} \to \tilde{\boldsymbol{x}}$. Hence, the spec converges dynamically with respect to $\tilde{\boldsymbol{x}}$. As the spec is further said to be dynamically unbiased, the damped spec forms a dynamic fabric by Theorem 4.9. ■

Concretely, Lemma 4.11 allows for trajectory following with guaranteed convergence with DF. For example, a reference trajectory for the robot's end-effector is defined as $\tilde{\boldsymbol{x}}(t)$. Then, the potential can be designed as $\psi = \tilde{\boldsymbol{x}}(t) - \boldsymbol{x}$ (respecting the construction rule required for Lemma 4.11). In contrast to SF, where the static potential function is simply updated at every time step, DF makes use of the dynamics of the reference trajectory through the dynamic pullback.

F. Construction Procedure

From the high-level procedure explained in Section IV-A, we can derive the algorithm using the formal findings in this section, see Algorithm 1.

Methods to design the individual components, such as geometry and Finsler structures, are introduced in [9]. As these design patterns do not vary for DF, they are not repeated here. In the result section, we show some experimental examples highlighting the comparative advantage of optimization fabrics over model predictive schemes and the advantage of DF over SF for dynamic environments.

V. Extension to Nonholonomic Constraints

Mobile manipulators are often equipped with a nonholonomic base (e.g., a differential drive mobile robot). In contrast to revolute joints for manipulators, nonholonomic bases imply

Algorithm 1: Motion design with DF.

Define basic inertia as spec $\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{M}\boldsymbol{h} = 0$
for *avoidance in avoidances* **do**
  Define differential map between $\mathcal{Q}$ and $\mathcal{X}_i$: ϕ or ϕ_t
  Design geometry on $\mathcal{X}_i$: $\ddot{\boldsymbol{x}}_i + \boldsymbol{h}_{2,i} = \boldsymbol{0}$
  Design Finsler energy for behavior on $\mathcal{X}_i$: $\mathcal{L}_i$
  Energize geometry with Finsler energy IV.4
  if ϕ *is time-parameterized* **then**
    Apply dynamic pullback to energized system
  end
  Apply standard pullback
  Add pulled avoidance component to root fabric
end
Force root system with (time-parameterized) potential

nonholonomic constraints. Based on ideas presented in [29], we propose a method to integrate such constraints in optimization fabrics, including DF.

We assume that the nonholonomic constraint at hand can be expressed as an equality of form

$$\dot{\boldsymbol{x}} = \boldsymbol{J}_{\text{nh}}\dot{\boldsymbol{q}} \tag{16}$$

where $\boldsymbol{J}_{\text{nh}}$ is the Jacobian of the constraint, $\dot{\boldsymbol{q}}$ is the velocity of the controlled joints of the system, and $\dot{\boldsymbol{x}}$ is the root velocity of the fabric. For a differential drive, $\dot{\boldsymbol{x}}$ is the velocity of the system in the Cartesian plane $(\dot{x}, \dot{y}, \dot{\theta})$ and $\dot{\boldsymbol{q}}$ is the velocity of the actuated wheels $(u_{\text{left}}, u_{\text{right}})$. Moreover, we assume that (16) is smooth and differentiable so that we can write

$$\ddot{\boldsymbol{x}} = \dot{\boldsymbol{J}}_{\text{nh}}\dot{\boldsymbol{q}} + \boldsymbol{J}_{\text{nh}}\ddot{\boldsymbol{q}}. \tag{17}$$

The theory of optimization fabrics allows to pull a tree of fabrics back into one fabric expressed in its root coordinates of form $\boldsymbol{M}\ddot{\boldsymbol{x}} + \boldsymbol{f} = 0$ with its solution as

$$\ddot{\boldsymbol{x}} = -\boldsymbol{M}^{-1}\boldsymbol{f}. \tag{18}$$

Plugging (17) into the root fabric, we obtain the nonholonomic fabric of form

$$\begin{aligned}\boldsymbol{M}\boldsymbol{J}_{\text{nh}}\ddot{\boldsymbol{q}} + \boldsymbol{M}\dot{\boldsymbol{J}}_{\text{nh}}\dot{\boldsymbol{q}} + \boldsymbol{f} &= 0\\ \boldsymbol{M}_{\text{nh}}\ddot{\boldsymbol{q}} + \boldsymbol{f}_{\text{nh}} &= 0.\end{aligned}$$

Note that $\boldsymbol{M}_{\text{nh}}$ is not necessarily a square matrix and, thus, not invertible as it was in the original fabric. To find the best actuation for the wheels, we formulate motion generation with fabrics as an unconstrained optimization problem

$$\ddot{\boldsymbol{q}}^* = \min_{\ddot{\boldsymbol{q}}} \|\boldsymbol{M}_{\text{nh}}\ddot{\boldsymbol{q}} + \boldsymbol{f}_{\text{nh}}\|_2^2. \tag{19}$$

In this approach, we minimize the error of the final equation. We could equally derive (19) with the objective of minimizing the error between $\ddot{\boldsymbol{x}} = \boldsymbol{J}_{\text{nh}}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}_{\text{nh}}\dot{\boldsymbol{q}}$ and the original fabric's solution $\ddot{\boldsymbol{x}} = -\boldsymbol{M}\boldsymbol{f}$. The minimization of the difference leads to similar result. This optimization problem replaces (18) and is solved by

$$\ddot{\boldsymbol{q}}^* = \boldsymbol{M}_{\text{nh}}^{\dagger}\boldsymbol{f}_{\text{nh}}. \tag{20}$$

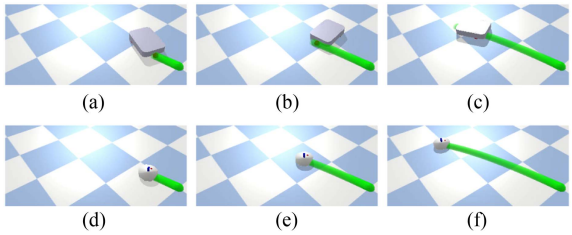


Fig. 4. Path (green) following with a holonomic and a nonholonomic robot using DF with the extension to nonholonomic robots. (a) 7 s. (b) 11 s. (c) 18 s. (d) 7 s. (e) 11 s. (f) 18 s.

The solutions to this problem makes optimization fabrics, and thus DF, applicable to nonholonomic robots, such as differential drive robots or cars. A qualitative comparison between a trajectory generated for a holonomic and a nonholonomic robot is shown in Fig. 4.

The theory of optimization fabrics is built upon energy conservation of artificial energies that design the motion. Equation (19) does not solve the resulting spec exactly, but minimizes the deviation according to the least square objective function. For many kinematic systems, e.g., differential drive model, bicycle model, the nonholonomic constraint additionally reduces the number of degrees of freedom, $\dim q < \dim x$. As a consequence, the least square solution has a nonzero residuum. Then, some fundamental properties of optimization fabrics, such as energy conservation and convergence, can no longer be guaranteed. Despite this theoretical shortcoming, we show that this approach leads to good performance in practical applications.

VI. Experimental Results

In this section, the performance of optimization fabrics is assessed on various robotic platforms. Although [9] suggested performance benefits over optimization-based methods to local motion planning, no quantitative comparisons have been presented to this date. The scenarios that we have chosen here (especially in the first two experiments) are intentionally simple to identify the specific differences. In the real-world experiments, we show the differences on more dynamic scenarios, where the limited frequency of a global planning method, such as RRT, justifies the need for a local planning method. To give a general idea of the performance differences between SF and receding-horizon trajectory optimization, we compare the performance of an MPC formulation, adapted from [24], with SF, as proposed in [9]. The second experiment compares performance between SF and DF for trajectory-following tasks. In the third experiment, moving obstacles are added to the scene to form a dynamic environment. Our extension to nonholonomic systems is tested in the fourth experiment. Then, everything is combined in an experiment with a differential drive mobile manipulator. Finally, we present a possible application of a robot sharing the environment with a human. The experiments described here are supported by videos accompanying this article.

A. Settings and Performance Metrics

We present a detailed analysis of the experimental results for two commonly used setups, namely the *Franka Emika Panda*, a *Clearpath Boxer*, and a mobile manipulator composed of both components see [24]. Note that these robots are representative of commonly used robots in dynamic environments. The Franka Emika Panda is a 7 degree-of-freedom robot with joint torque sensors, comparable to the Kuka Iiwa. Mobile manipulators equipped with differential drives are widely used by other manufacturers, see Pal Robotics Tiago or the Fetch Robotics Mobile Manipulator. Compared with [8], we propose a more extensive list of metrics. With regards to safety, we measure the *Clearance*, the minimum distance between the robot and any obstacle along the path. For static goals, solver planner performance is measured in terms of *Path Length*, Euclidean length of the end-effector trajectory, and *Time-to-Goal*, time to reach the goal. For trajectory-following tasks, this measure is replaced by *Summed Error*, the normed sum of deviation from the desired trajectory. Computational costs are measured by the average *Solver Time* in each time step. Most important, binary success is measured by the *Success Rate*, where failure indicates that either the goal was not reached or a collision occurred during execution. Performance metrics are only evaluated if the concerned motion generator succeeded. More information on the testbed can be found in [32].

In static, industrial environments, the time to reach the goal can be considered the one single most important metric, but we argue that dynamic environments require a more nuanced performance evaluation and, thus, a set of metrics. Intentionally, we do not give general weights to the individual metrics, as their corresponding importance highly depends on the application. As a consequence, we tuned the compared planners in such a way that they reach the goal in a similar time. Note that the general speed for all planners compared in this article can be adjusted by choosing a different parameter setup.

As this work does not focus on obstacle detection, we simplify obstacles to spheres. Thus, we assume that an operational perception pipeline detects obstacles and constructs englobing spheres. The experiments are randomized in either the location of the obstacles, the location of the goal, the initial configuration, or in a combination of all three aspects. For every experiment, the type and level of randomization are stated.

B. Experiment 1: SF Versus MPC

In the first experiment, we compare the performance of an MPC formulation with SF [9], [10]. Compared with the formulation used in [24], we use a workspace goal rather than a configuration space goal, and apply a second-order integration scheme so that the control outputs are accelerations instead of velocities. We clarify that the formulation deployed for the following tests is geometric as the model used is a second-order integrator and does not include the dynamics of the robots. The main reason lies in the reduced computational costs and the inaccessibility of a highly accurate model [33].

1) Parameters: The low-level controller of the robot runs at 1 kHz. The fabrics are running at 100 Hz and the MPC at 10 Hz. The time horizon for the MPC planner was set to $T = 3$ s spread equally over $H = 30$ stages. Based on the findings in [24], we are confident that the MPC planner is close to its optimal settings. Moreover, we used the implementations by [34] and [35], which are reported to have improved performance over open-source libraries, such as *acado*.

2) Simulation: A series of $N = 50$ runs was evaluated with the panda robot in simulation. Randomized end-effector positions were set for every run, whereas the initial configuration remained unchanged. One to five spherical obstacles of radius $r = 0.15$ m were placed in the workspace at random. An example setup is shown in Fig. 6(a). The results are summarized in Fig. 7. Solver times with fabrics averaged at 1 ms while the MPC solver took around 50 ms in every time step. Although the path length is similar with both solvers, the minimum clearance from obstacles is increased with SF (0.183 m) compared with MPC (0.138 m). This means that the trajectories are safer and, thus, more suitable for dynamic environments. Both motion generation methods fail in six cases. However, the SF produce only one collision while MPC creates five collisions. The remaining failures are deadlocks. For both methods, deadlocks result from local minima, highlighting the need for supportive global plans. Collisions are caused by numerical inaccuracies, which are generally higher with MPC due to the lower frequency.

3) Real World: For the experiments with the real robot, we limited the number of test runs to $N = 20$. In contrast to the simulated results, MPC has significantly more collisions than SF, Fig. 8(b). This is likely to be caused by the lower frequency at which the MPC is running. While in simulation, the model matches the actual behavior perfectly and the time interval between two computations can be accurately predicted, and more uncertainty in the model is present in the real world. This leads to prediction errors that cause collisions. For the collision free cases, the real-world experiments confirm that optimization fabrics tend to be more conservative with respect to obstacles, see *Clearance* in Fig. 8(a). Similar to the simulated results, the solving time is reduced by a factor of around 50 with fabrics. This allows to run the planner at a higher frequency and, thus, generating smoother motions.

4) Discussion: The difference in performance (except for solver time) is likely caused by the different objective metrics. The objective function in the MPC formulation is mainly governed by the Euclidean distance to the goal while control inputs and velocity magnitude are given a relative small weight. Avoidance behaviors, such as joint limit avoidance and obstacle avoidance, are respected through inequality constraints. In contrast, SF design the objective in a purely geometric manner including all avoidance behaviors. Thus, the manifold for the motion is directly altered by the avoidance behaviors, i.e., the manifold is *bent* [9] so that the notion of shortest path changes with the addition of obstacles. This shaping of the manifold leads to improved convergence compared with the combination of Euclidean distance objective function and inequality constraints used with MPC.

C. Experiment 2: SF Versus DF

In motion planning for dynamic environments, global and local planning methods work together to achieve efficient and

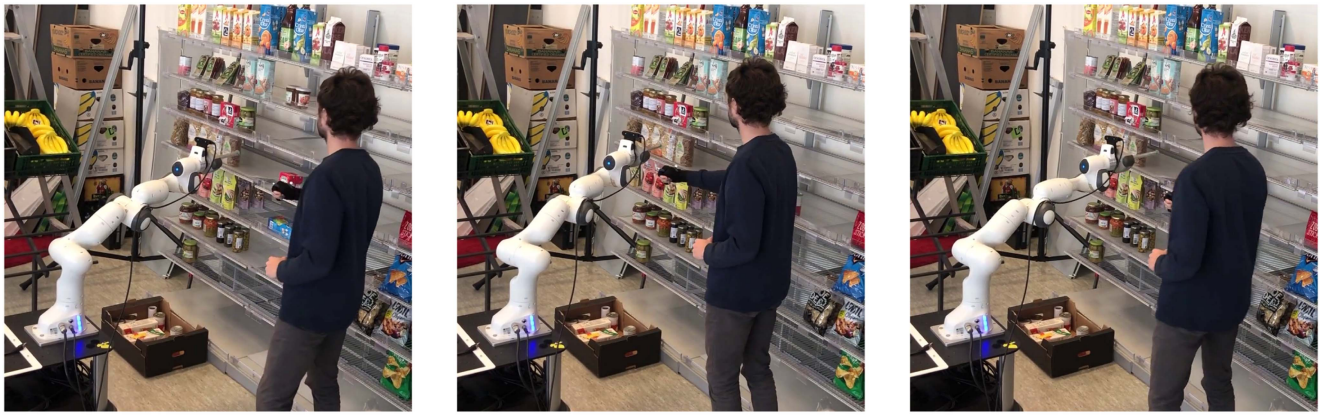


Fig. 5. DF in the presence of a human. The human hand's state is estimated with a motion capture system. The robot smoothly and in advance avoids the human operator and allows for safe coexistence. (a) $t = 0$ s. (b) $t = 2$ s. (c) $t = 3$ s.

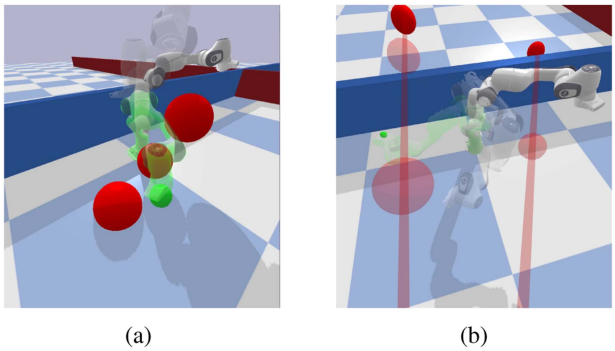


Fig. 6. Examples for simulation setups with panda robot. Initial configurations are shown in white and final configurations are shown in light green. Obstacles are visualized in red. (a) Only static obstacles are considered. (b) Trajectories of two moving obstacles are visualized in light red.

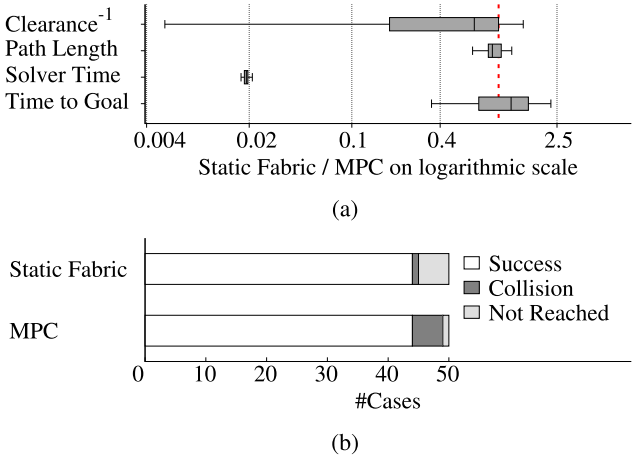


Fig. 7. Results for randomized motion planning problems with the panda robot in simulation. Lower values represent an improved performance of SF over MPC. (a) Metrics evaluation for successful experiments. (b) Success results.

safe motion of the robot. However, SF are not designed to follow global paths. Path following can only be achieved using a pseudodynamic approach where the forcing potential is shifted in every time step without considering the dynamics

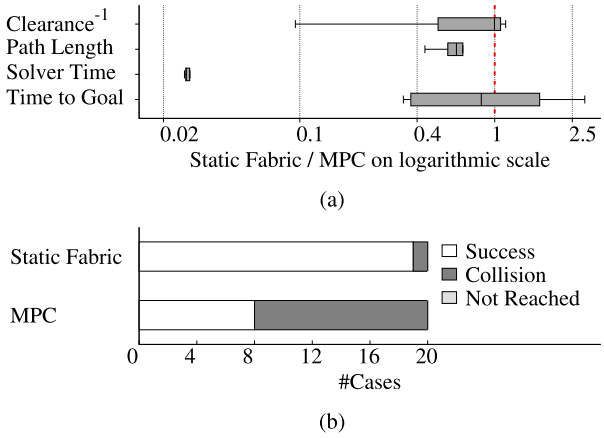


Fig. 8. Results for randomized motion planning problems with the real panda robot. SF are more conservative around obstacles, improving on safety, while reducing the computational cost by a factor of ≈ 50. As a result of the increased clearance, collisions are more reliably avoided with SF. (a) Metrics evaluation for successful experiments. (b) Success results.

of the trajectory. Therefore, we propose DF to allow smoother path-following tasks, where the speed of the goal is also considered during execution. In this second experiment, we investigate how DF compare to SF for path-following tasks. Specifically, we show that DF outperform SF when following a path generated by a global planner.

1) Simulation: We evaluated DF on the panda robot in simulation with an analytic, time-parameterized curve and a path generated by a global planner, namely RRT (see Fig. 9). In the case of the analytic trajectory, the three obstacles were randomized across all runs. For the experiment with the global planner, the goal position and the obstacles were randomized across all runs. A total of $N = 50$ experiments were executed for this experiment. The reduced summed error for DF verifies the theoretical finding that DF can follow paths more closely. The average error over all runs with the analytic trajectory is 0.0792 m (DF) and 0.136 m (SF), see Fig. 10(a) for the comparison. For the spline path generated with RRT, the average

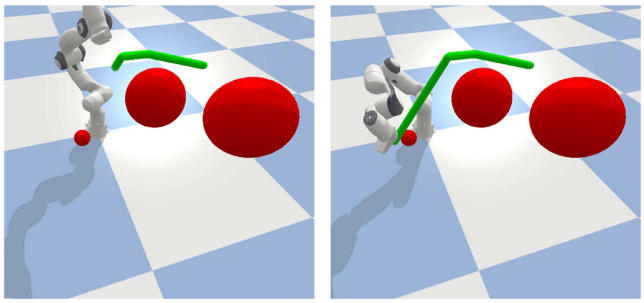


Fig. 9. Path generated with RRT from OMPL.

Fig. 10. Comparison between SF and DF for trajectory-following tasks in simulation. Lower values in a metric indicate that DF performed better than SF. (a) Analytic, user-specified global path. (b) Global path generated by RRT using OMPL.

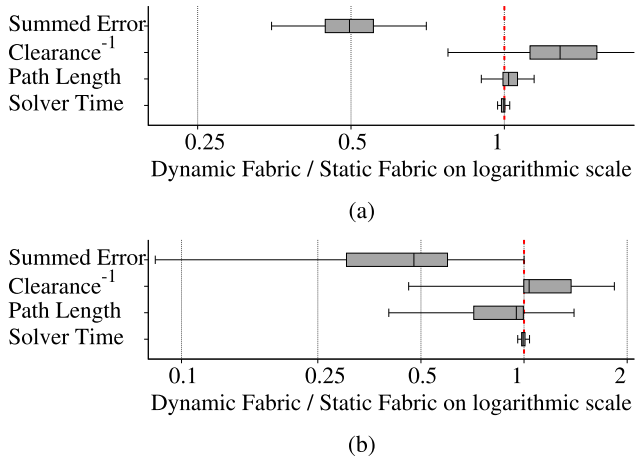


Fig. 11. Trajectory-following tasks with DF. (a) Trajectory is a time-parameterized analytic curve. (b) Trajectory is described by a spline.

error over all runs is 0.145 (DF) and 0.240 m (SF), see Fig. 10(b) for the comparison.

2) Real World: Path following was also assessed with the real panda robot in similar settings. Quantitative results are only presented for $N = 20$ different paths with splines where up to three obstacles were added to the workspace, see Fig. 11. The results in real-world confirm the findings from the simulation. By exploiting the velocity information of the trajectory, the integration error can be effectively reduced, Fig. 12. In contrast to the simulation we see a higher fluctuation in solver times, which

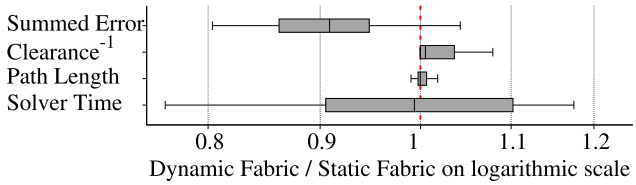


Fig. 12. Comparison between SF and DF when following a path defined by a basic spline in the real world. The splines and the obstacles are different for the $N = 20$ case. DF achieve lower deviation errors that SF.

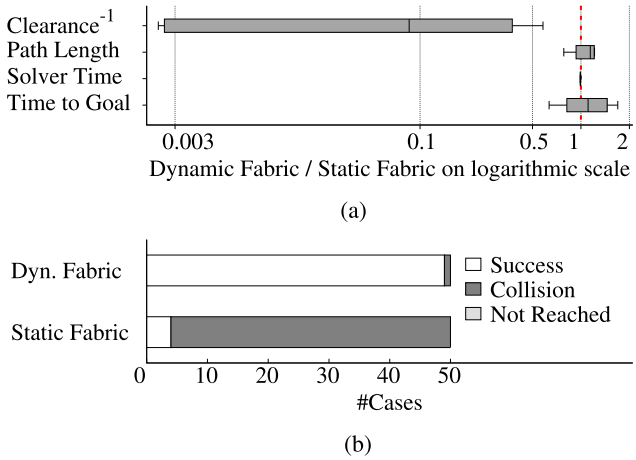


Fig. 13. Comparison between SF and DF for scenarios with dynamic obstacles. While path length and solver time is not increased, clearance is increased and the time to reach the goal is reduced with DF compared with SF. (a) Metrics evaluation for successful experiments. (b) Success results.

can be caused by a generally lower capacity of the computing unit on the robot.

D. Experiment 3: Moving Obstacles

Next, we compare the different methods in the presence of dynamic obstacles. All experiments in this section consist of at least one moving obstacle that follows either an analytic trajectory or a spline. Here, we use stationary goals to isolate the results from the behavior investigated in the previous section.

1) Simulation: For this series with the simulated panda robot, only the goal position was randomized. The initial configuration

$$\boldsymbol{q}_0 = [1.0, 0.0, 0.0, -1.5, 0.0, 1.8675]^T$$

and the two moving obstacles with the trajectories

$$\tilde{\boldsymbol{x}}_{\text{obst1}} = [-1.0 + 0.1t, -0.4, 0.7]^T$$

$$\tilde{\boldsymbol{x}}_{\text{obst2}} = [-1.0 + 0.2t, 1.0 - 0.1t, 0.3]^T$$

were kept constant throughout all experiments. The environment is visualized in Fig. 6(b). The comparison between SF and DF shows that DF are more conservative in terms of collision avoidance with dynamic obstacles. Specifically, the distance between the robot and the obstacles is increased [see Fig. 13(a)]. The success rate with DF compared with SF is significantly improved, see Fig. 13(b). Thus, showing the need for using DF in dynamic environments.

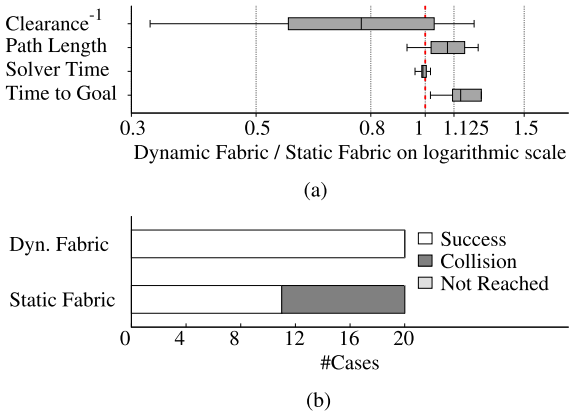


Fig. 14. Comparison between SF and DF for real-world scenarios with dynamic obstacles. (a) Metrics evaluation for successful experiments. (b) Success results.

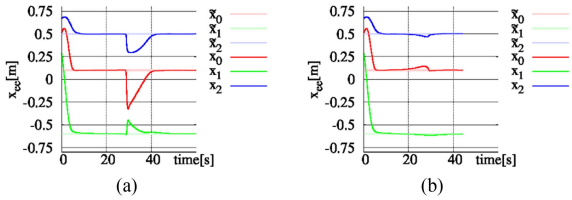


Fig. 15. Trajectories for real panda robot in the presence of a dynamic obstacle. DF show a smoother and in-advance reaction to the approaching obstacle while SF can only react in sudden motion. (a) SF. (b) DF.

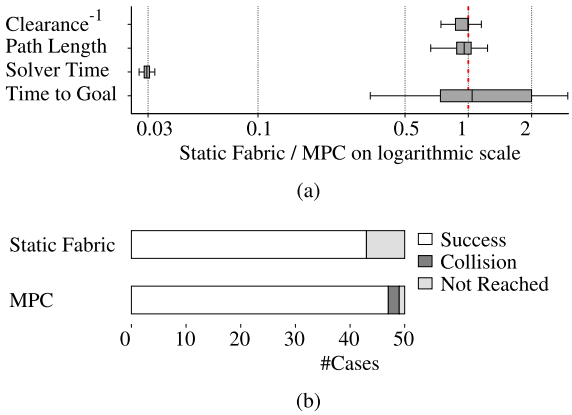


Fig. 16. Results for randomized cased with the Clearpath Boxer robot. Similar performance in terms of safety and goal reaching can be combined with very fast computation using optimization fabrics. (a) Metrics evaluation for successful experiments. (b) Success results.

2) Real World: In a series of $N = 20$ experiments, performance on the real panda arm was assessed. The same trend for more conservative behavior with DF compared with SF can be observed, see Fig. 14. However, DF take longer on average to reach the goal as they keep larger clearance from obstacles. Note that collisions are effectively eliminated with DF compared with SF.

By investigating one example out of the series, see trajectories in Fig. 15, the reason for the large number of collisions with SF can be explained. Both methods initially drive the end-effector to the goal position. As the moving obstacle is approaching the

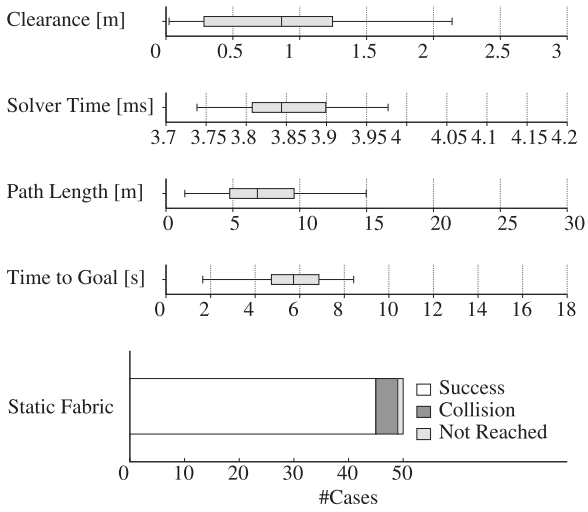


Fig. 17. Quantitative results with SF for a nonholonomic mobile manipulator in simulation. Fabrics solve planning problems in randomized environments in low planning time. This allows whole-body control and highly reactive behavior.

robot, the DF are starting to react while SF are not changing its behavior resulting in a very sudden motion at around $t = 30$ s. SF treat moving obstacles as pseudostatic (i.e., the position of the obstacle is updated at every time step, but the information on its velocity is discarded). As a result, the relative velocity between obstacle and robot is only a function of the velocity of the robot. Geometries and energies for collision avoidance with fabrics are, by design, a function of this velocity and, therefore, fail to avoid moving obstacles when the robot moves slowly or not at all. This behavior is most visible when the goal has already been reached but an obstacle is approaching. DF on the other hand take the velocity of the moving obstacles into account and can, therefore, avoid them.

E. Experiment 4: Nonholonomic Robots

1) Simulation: This experiment assesses the performance of the proposed method to compute trajectories for nonholonomic robots with fabrics. Specifically, we run experiments for a *Clearpath Boxer* for position. As for the first experiment, we compare the performance with MPC. In this experiment, the initial position, the goal location, and the position of five obstacles were randomized. The results reveal that our extension of optimization fabrics to nonholonomic robots maintains similar results as with a robotic arm. Specifically, computational time can be reduced to optimization-based methods while maintaining good performance in terms of safety and goal-reaching, see Fig. 16. We can also observe that success rate with SF is lower compare with MPC due to a high number of unreached goals.

F. Experiment 5: Mobile Manipulators

In the final experiment, we assess the applicability of SF and DF to a nonholonomic mobile manipulator. In an environment that is densely occluded by obstacles, the motion planning

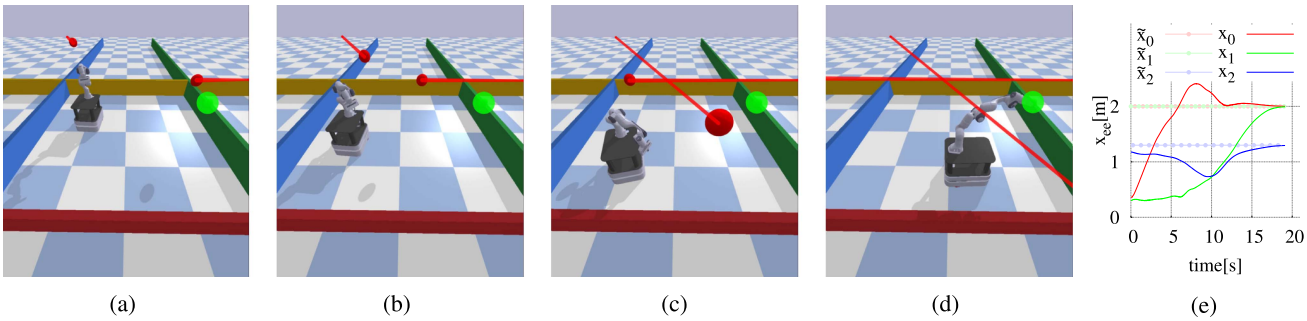


Fig. 18. Sequence of trajectory computed with DF for a mobile manipulator in simulation with moving obstacles (red sphere with line indicating the past trajectory) and one end-effector goal (green). The trajectory of the end-effector are visualized in (e) as $\boldsymbol{x}$ and the desired end-effector position as $\tilde{\boldsymbol{x}}$. (a) $t = 0$ s. (b) $t = 7$ s. (c) $t = 14$ s. (d) $t = 20$ s.

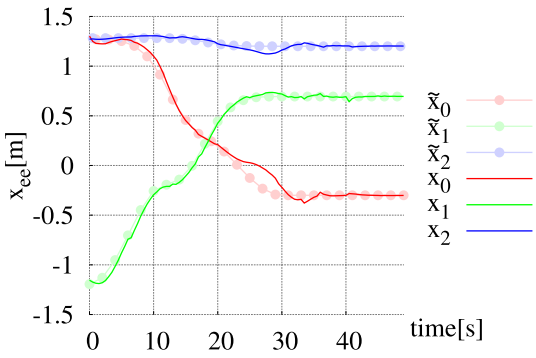


Fig. 19. Real-world experiment for path following with a mobile manipulator. The global path can be tracked accurately by DF, including the extension to nonholonomic robots. The scene is visualized in Fig. 1.

problem is defined by a desired end-effector position and additional path constraints (e.g., desired orientation of the end-effector).

1) Simulation: In simulation, we evaluate the performance of our extension to nonholonomic mobile manipulators with SF. In this series, the positions of eight obstacles are randomized for $N = 50$ cases. The workspace was limited to a 7 m $\times$ 7 m square, so that random obstacles are ensured to be actually hindering the motion planner. The results reveal that properties shown in the previous experiments transfer to more complex systems without loss of the computational benefit, see Fig. 17. In this series, there were one unreached goals and four collisions that are, similar to the previous experiments, caused by local minima. Local minima are more likely for mobile manipulators as their workspace is larger. Combining our contributions, DF and the extension to nonholonomic robots, we achieve reactive and safe behavior in dynamic environments. Moving obstacles are avoided in a natural way using our method, see Fig. 18.

2) Real World: We present qualitative results for a nonholonomic mobile manipulator using DF. In Fig. 1, the robot follows a trajectory defined by a basic spline, while additionally respecting an orientation constraints on its end-effector and avoiding the shelves and an obstacle on the ground. The end-effector trajectory is plotted in Fig. 19.

G. Experiment 6: DF in Supermarkets

In this experiment, we show qualitatively how DF could be used in collaborative environments where humans and robots coexist. For this experiment, we give the robot a static goal pose similar to a pickup setup. The same environment is shared with a coworker who restocks a shelf. The right-hand side of the human is tracked with a motion capture system. The hand is then avoided by the robot using DF, see Fig. 5. In this experiment, the minimum distance between the robot and the hand was 0.062 m. This real-world experiment showcases potential applications of the proposed method.

VII. Conclusion

In this article, we have generalized optimization fabrics to dynamic environments. We have proven that our proposed DF are convergent to reference paths and can, thus, compute motion for path-following tasks (see Lemma IV.10). Besides, we have proposed an extension to optimization fabrics (and, thus, also DF) for nonholonomic robots. This allows the application of this framework to a wider range of robotic applications and ultimately allows the deployment to many mobile manipulators in dynamic environments.

These theoretical findings were confirmed in various experiments. First, the quantitative comparisons showed that SF outperforms MPC in terms of solver time while maintaining similar performance in terms of goal reaching and success rate. The improved performance with optimization fabrics might be caused by the different metric for goal reaching compared with MPC. An integration of non-Riemannian metrics into an MPC formulation should be further investigated in the future.

Verifying our theoretical derivations for DF, the experiments showed that the deviation error for path-following tasks is decreased compared with SF. Similarly, environments with moving obstacles and humans showed increased clearance while maintaining low computational costs and execution times. Thus, DF overcome an important drawback of SF [9], [10], where collision avoidance with moving obstacle is solved purely by the high frequency at which optimization fabrics can be computed. Moreover, the generalization did not increase the solving time compared with SF. Unlike the original work on optimization

fabrics, this generalization allows the deployment to dynamic environments where velocity estimates of moving obstacles are available.

Direct sensor integration in optimization fabrics might be feasible in future works to overcome the shortcomings of perception pipelines for collision avoidance. For the trajectory path tasks in this article, we used a simple global path generated in workspace. As DF integrate global path in arbitrary manifolds, improving the global planning phase could be further investigated. We expect this to be beneficial when robotics tasks are constantly changing and task planning is required.

ACKNOWLEDGMENT

All content represents the opinion of the author(s), which is not necessarily shared or endorsed by their respective employers and/or sponsors.

REFERENCES

[1] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robot. Res.*, vol. 30, no. 7, pp. 846–894, 2011. [Online]. Available: https://journals.sagepub.com/doi/10.1177/0278364911406761

[2] D. Hrovat, S. Di Cairano, H. Tseng, and I. Kolmanovsky, "The development of model predictive control in automotive industry: A survey," in *Proc. IEEE Int. Conf. Control Appl.*, 2012, pp. 295–302.

[3] M. Bujarbaruah, X. Zhang, F. Borrelli, and H. Eric Tseng, "Learning-based model predictive control: Toward safe learning in control," *Annu. Rev. Control, Robot., Auton. Syst.*, vol. 3, pp. 269–296, 2018. [Online]. Available: https://doi.org/10.1146/annurev-control-090419-

[4] M. Bednarczyk, H. Omran, and B. Bayle, "Model predictive impedance control," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 4702–4708.

[5] S. Richter, S. Mariéthoz, and M. Morari, "High-speed online MPC based on a fast gradient method applied to power converter control," in *Proc. Amer. Control Conf.*, 2010, pp. 4737–4743.

[6] M. Xie et al., "Geometric fabrics for the acceleration-based design of robotic motion," Oct. 2020. [Online]. Available: https://arxiv.org/abs/2010.14750

[7] W. Edwards, G. Tang, G. Mamakoukas, T. Murphey, and K. Hauser, "Automatic tuning for data-driven model predictive control," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 7379–7385.

[8] C. A. Cheng et al., "RMPflow: A computational graph for automatic motion policy generation," 2018.

[9] N. D. Ratliff, K. Van Wyk, M. Xie, A. Li, and M. A. Rana, "Optimization fabrics," 2020, *arXiv:2008.02399*.

[10] K. Van Wyk et al., "Geometric fabrics: Generalizing classical mechanics to capture the physics of behavior," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 3202–3209, Apr. 2022.

[11] C.-A. Cheng et al., "RMPflow: A geometric framework for generation of multi-task motion policies," *IEEE Trans. Autom. Sci. Eng.*, vol. 18, no. 3, pp. 968–987, Jul. 2021.

[12] M. Rickert, A. Sieverling, and O. Brock, "Balancing exploration and exploitation in sampling-based motion planning," *IEEE Trans. Robot.*, vol. 30, no. 6, pp. 1305–1317, Dec. 2014.

[13] M. Stilman, "Global manipulation planning in robot joint space with task constraints," *IEEE Trans. Robot.*, vol. 26, no. 3, pp. 576–584, Jun. 2010.

[14] D. Berenson, S. Srinivasa, and J. Kuffner, "Task space regions: A framework for pose-constrained manipulation planning," *Int. J. Robot. Res.*, vol. 30, no. 12, pp. 1435–1460, Oct. 2011.

[15] Z. Kingston, M. Moll, and L. E. Kavraki, "Exploring implicit spaces for constrained sampling-based planning," *Int. J. Robot. Res.*, vol. 38, no. 10-11, pp. 1151–1178, 2019.

[16] A. H. Qureshi, J. Dong, A. Baig, and M. C. Yip, "Constrained motion planning networks X," *IEEE Trans. Robot.*, vol. 8, no. 2, pp. 868–886, 2021.

[17] B. Brito, B. Floor, L. Ferranti, and J. Alonso-Mora, "Model predictive contouring control for collision avoidance in unstructured dynamic environments," *IEEE Robot. Automat. Lett.*, vol. 4, no. 4, pp. 4459–4466, Jul. 2019.

[18] D. Mayne, J. Rawlings, C. Rao, and P. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, Jun. 2000. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0005109899002149

[19] A. W. Winkler, C. D. Bellicoso, M. Hutter, and J. Buchli, "Gait and trajectory optimization for legged systems through phase-based end-effector parameterization," *IEEE Robot. Automat. Lett.*, vol. 3, no. 3, pp. 1560–1567, Jul. 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8283570/

[20] S. S. Keerthi and E. G. Gilbert, "Optimal infinite-horizon feedback laws for a general class of constrained discrete-time systems: Stability and moving-horizon approximations," *J. Optim. theory Appl.*, vol. 57, no. 2, pp. 265–293, 1988.

[21] J. Tordesillas, B. T. Lopez, and J. P. How, "FASTER: Fast and safe trajectory planner for flights in unknown environments," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2019, pp. 1934–1940. [Online]. Available: https://github.com/jtorde

[22] G. B. Avanzini, A. M. Zanchettin, and P. Rocco, "Constraint-based model predictive control for holonomic mobile manipulators," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2015, vol. 2015, pp. 1473–1479.

[23] G. Buizza Avanzini, A. M. Zanchettin, and P. Rocco, "Constrained model predictive control for mobile robotic manipulators," Robotica, vol. 36, no. 1, pp. 19–38, 2018. [Online]. Available: https://doi.org/10.1017/S0263574717000133

[24] M. Spahn, B. Brito, and J. Alonso-Mora, "Coupled mobile manipulation via trajectory optimization with free space decomposition," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 12759–12765.

[25] I. R. Manchester and J.-J. E. Slotine, "Control contraction metrics: Convex and intrinsic criteria for nonlinear feedback design," *IEEE Trans. Autom. Control*, vol. 62, no. 6, pp. 3046–3053, Jun. 2017. [Online]. Available: https://ieeexplore.ieee.org/document/7852456/

[26] B. Yi, R. Wang, and I. R. Manchester, "On necessary conditions of tracking control for nonlinear systems via contraction analysis," Nov. 2020, *arXiv:2009.08662*. [Online]. Available: https://arxiv.org/abs/2009.08662

[27] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE J. Robot. Autom.*, vol. 3, no. 1, pp. 43–53, Feb. 1987.

[28] N. D. Ratliff, J. Issac, D. Kappler, S. Birchfield, and D. Fox, "Riemannian motion policies," Jan. 2018. [Online]. Available: https://arxiv.org/abs/1801.02854

[29] X. Meng, N. Ratliff, Y. Xiang, and D. Fox, "Neural autonomous navigation with Riemannian motion policy," in *Proc. - IEEE Int. Conf. Robot. Automat.*, 2019, vol. 2019, pp. 8860–8866.

[30] N. D. Ratliff, K. Van Wyk, M. Xie, A. Li, and M. A. Rana, "Generalized nonlinear and Finsler geometry for robotics," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 10206–10212.

[31] M. Bhardwaj et al., "STORM: An integrated framework for fast joint-space model-predictive control for reactive manipulation," in *Proc. Conf. Robot Learn.*, 2022, pp. 750–759.

[32] M. Spahn, C. Salmi, and J. Alonso-Mora, "Local planner bench: Benchmarking for local motion planning," 2022, *arXiv:2210.06033*.

[33] C. Meo, G. Franzese, C. Pezzato, M. Spahn, and P. Lanillos, "Adaptation through prediction: Multisensory active inference torque control," 2021, *arXiv:2112.06752*.

[34] A. Domahidi and J. Jerez, "Forces professional," Embotech AG, Zurich, Switzerland, 2014–2019. [Online]. Available: https://embotech.com/FORCES-Pro

[35] A. Zanelli, A. Domahidi, J. Jerez, and M. Morari, "FORCES NLP: An efficient implementation of interior-point... methods for multistage nonlinear nonconvex programs," *Int. J. Control*, vol. 93, no. 1, pp. 13–29, 2017.

Max Spahn received the B.Sc. degree in mechanical engineering and the M.Sc. degree in general mechanical engineering from the RWTH Aachen, Aachen, Germany, in 2018 and 2019, respectively, and the Diplôme d'ingénieur des Arts et Manufactures degree from Centrale Supelec, Gif-sur-Yvette, France, in 2020 as part of a double degree program. He is currently working towards the Ph.D. degree in robotics.

He is part of AIRLab, the AI for Retail Lab, Delft, The Netherlands. His research interests include geometric approaches to motion planning and trajectory generation. He is passionate about open-sourcing scientific code, as he believes it will improve scientific outcome in robotics.

Martijn Wisse received the M.Sc. and Ph.D. degrees in mechanical engineering from the Delft University of Technology, Delft, The Netherlands, in 2000 and 2004, respectively.

He is currently a Professor with the Delft University of Technology. His previous research focused on passive dynamic walking robots and passive stability in the field of robot manipulators. He worked on underactuated grasping, open-loop stable manipulator control, design of robotic systems, and the creation of startups in this field. His research focuses on the neuroscientific principle of active inference and its application and advancements in robotics.

Javier Alonso-Mora received the Ph.D. degree in robotics on cooperative motion planning from ETH Zurich, Zurich, Switzerland, in partnership with Disney Research Studios, Zurich, Switzerland.

He is currently an Associate Professor with the Delft University of Technology, Delft, The Netherlands, where he leads the Autonomous Multi-robots Lab. He was a Postdoctoral Associate with Computer Science and Artificial Intelligence Lab (CSAIL), Massachusetts Institute of Technology, Cambridge, MA, USA. His research interests include navigation, motion planning, and control of autonomous mobile robots, with a special emphasis on multirobot systems, mobile manipulation, on-demand transportation, and robots that interact with other robots and humans in dynamic and uncertain environments.

Dr. Alonso-Mora is an Associate Editor for IEEE TRANSACTIONS ON ROBOTICSand for *Springer Autonomous Robots*. He was the recipient of a talent scheme VENI Award from the Netherlands Organization for Scientific Research (2017), ICRA Best Paper Award on Multi-robot Systems (2019), and an ERC Starting Grant (2021).