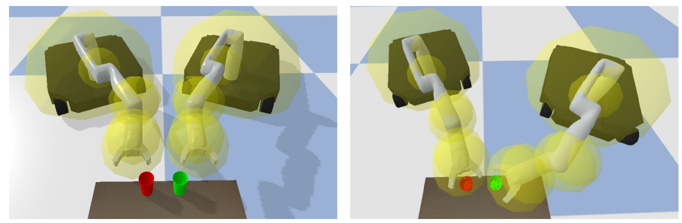
Reactive grasp and motion planning for adaptive mobile manipulation among obstacles

Tomas Merva[1,2], Saray Bakker[2], Max Spahn[2], Ivan Virgala[1] and Javier Alonso-Mora[2]
[1] Department of Industrial Automation and Mechatronics, Technical University of Kosice
[2] Department of Cognitive Robotics, Delft University of Technology
Email: Tomas.Merva@tuke.sk, S.Bakker-7@tudelft.nl

***Abstract*—Mobile manipulators are susceptible to situations in which the precomputed grasp pose is not reachable as the result of conflicts between collision avoidance behaviour and the manipulation task. In this work, we address this issue by combining real-time grasp planning with geometric motion planning for decentralized multi-agent systems, referred to as Reactive Grasp Fabrics (RGF). We optimize the precomputed grasp pose candidate to account for obstacles and the robot's kinematics. By leveraging a reactive geometric motion planner, specifically geometric fabrics, the grasp optimization problem can be simplified, resulting in a fast, adaptive framework that can resolve deadlock situations in pick-and-place tasks. We demonstrate the robustness of this approach by controlling a mobile manipulator in both simulation and real-world experiments in dynamic environments.**



(a) Vanilla geometric fabrics (b) RGF (ours)

Fig. 1: Mobile manipulators are grasping objects in close proximity. Without adapting the grasp poses, Fig. 1a, the robots cannot resolve the deadlock caused by the potential collision. Via the proposed RGF, both robots grasp the object while avoiding collisions, Fig. 1b.

I. Introduction and related works

One of the most common applications of robotic manipulators is the pick-and-place task. A grasp pose is selected based on a detected object's pose, which is used for trajectory planning and execution. A top-down grasp is usually sufficient for a single manipulator in a highly controlled environment. However, in the domain of mobile manipulation or multiple manipulator workspaces, the presence of other agents and dynamic obstacles can easily render the precomputed top-down grasp invalid. Moreover, the manipulation task may conflict with collision avoidance, resulting in deadlocks where neither of the manipulators can continue its task, Fig. 1. As a consequence of the dynamic nature of the environment, computing trajectories and grasp poses must be done in real time. In this paper, we aim for online grasp and motion planning that is especially robust in scenarios where robots operate in close proximity.

Multi-agent motion planning can be broadly divided into two categories: centralized and decentralized. As centralized approaches compute a collaborative plan for all agents to achieve their goals while ensuring collision avoidance, an optimal solution can be obtained. However, the high computational time is inevitable, and the necessary communication between agents is constraining [13]. On the other hand, decentralized methods do not have these limitations, but the solution is not jointly optimal since there is no global understanding of the problem. Our approach can adapt the grasp poses in a decentralized way by considering other robots as obstacles.

In the context of motion generation, repeated replanning is commonly done using the Model Predictive Control (MPC) framework. However, the required computation is a major bottleneck for high-dimensional systems [14]. Data-based methods represent an attractive alternative since they can drastically reduce computational time. Nonetheless, these approaches require a lengthy process of data collection and subsequent training. Moreover, they lack guarantees to satisfy all physical constraints [4]. Another class of methods is built on the concept of geometry control, such as Riemannian Motion Policies [11] and Geometric Fabrics [12, 15]. Fabrics can encode the desired behaviour of the system, including joint limits and collision avoidance, into geometries represented by second-order differential equations. The resulting geometry, obtained by combining all behaviours, shapes a manifold of the configuration space in such a way that avoidances are achieved in a smooth manner. The solution to the differential equation can be symbolically precomputed offline, thereby achieving a high replanning frequency.

The popular perception approach for pick-and-place tasks involves data-driven models trained on generated synthetic point clouds and using parallel-jaw grippers [3]. By processing sensory inputs from RGB-D cameras, these models provide top-down grasp candidates for the given objects, usually in the form of a grasp axis [7, 9, 8]. The drawback of these methods is that they do not consider the current state of obstacles, often rendering the path from the robot's initial state to the grasp candidate unreachable. This issue is especially common for mobile manipulators operating in dynamic environments. Ichnowski et al. [5] demonstrated that the grasp axis can be exploited by creating an additional degree of freedom around it to compute a more feasible grasp pose and trajectory that enables faster pick-and-place cycle time. However, simultaneous grasp and motion optimization introduces additional nonlinear constraints resulting in higher computation times [6]. Our approach is related to the work in [6] but achieves run-time performance via geometric fabrics.

We propose the Reactive Grasp and Motion Generation framework, denoted as RGF, for adaptive mobile manipulators operating in close proximity. The grasp planner is formulated as a nonlinear programming (NLP) problem that exploits the degrees of freedom around the grasp axis, determined by vision-based grasp analysis methods. Additionally, the NLP formulation allows us to integrate the robot's mechanical limits and collision avoidance constraints. By building on reactive geometric fabrics, we simplify grasp planning to account for the grasp pose and sparse waypoints with minimal constraints. The framework is capable of real-time adaptive grasp selection and motion planning in decentralized multi-agent settings. This paper provides the following contributions:

1) A framework for reactive grasp selection based on the current state of the environment guiding geometric fabrics for efficient motion planning.
2) Efficient decentralized multi-agent task execution in environments shared with humans and mobile manipulators, avoiding deadlock scenarios.
3) Experimental evaluation in simulation and real-world experiments.

II. Preliminaries

A. Geometric fabrics

Geometric fabrics define the desired behaviour of a system using second-order differential equations of the form $\ddot{\boldsymbol{x}} = \boldsymbol{h}(\boldsymbol{x}, \dot{\boldsymbol{x}})$ [12, 10]. The desired motions are described using an artificial dynamical system for each task, such as avoiding a collision between the end-effector of the robot and an obstacle. Each system is defined in a task space $\mathcal{X}_j$ with task variable $\boldsymbol{x}_j$ given $j \in [M]$, where M denotes the number of task spaces, and $[M] = \{j \in \mathbb{N} : j \leq M\}$. To ensure that trajectories generated by the dynamical systems are converging when forced, the system is *energized* using a Finsler energy. The resulting system $\ddot{\boldsymbol{x}} = \tilde{\boldsymbol{h}}(\boldsymbol{x}, \dot{\boldsymbol{x}})$ then forms a geometric fabric [10]. To ensure path consistency, e.g. energization only changes the speed along the path but not the path itself, the function $\boldsymbol{h}(\boldsymbol{x}, \dot{\boldsymbol{x}})$ is designed to be homogenous of order 2, $\boldsymbol{h}(\boldsymbol{x}, \alpha\dot{\boldsymbol{x}}) = \alpha^2 \boldsymbol{h}(\boldsymbol{x}, \dot{\boldsymbol{x}}),\ \forall \alpha \geq 0$.

To combine all behaviours, all task-dependent dynamical systems are pulled to the configuration space $\mathcal{C}$ and summed. The *pullback operation*, $\text{pull}_{\phi_j} : \mathcal{X}_j \rightarrow \mathcal{C}$, maps the energy-conserving fabric to the configuration space using a twice-differential map $\phi_j : \mathcal{C} \rightarrow \mathcal{X}_j$ [12]. For example, if the task variable is the end-effector position, the differential map is given by the forward kinematics, $\text{FK}(\boldsymbol{q})$. The pulled fabrics are summed in configuration space, where the resulting dynamical system $\ddot{\boldsymbol{q}} = \tilde{\boldsymbol{h}}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is a fabric as well since the pullback and summation operations are closed under algebra. The resulting energy-conserving fabric can be forced by a navigation policy $\boldsymbol{f}$ to the minimum of a potential function $\psi(\boldsymbol{q})$ when damped,

$$\ddot{\boldsymbol{q}} = \tilde{\boldsymbol{h}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{f}(\boldsymbol{q}, \dot{\boldsymbol{q}}). \tag{1}$$

Eq. (1) is therefore a combination of all avoidance behaviours defined as energy-conserving fabrics $\tilde{\boldsymbol{h}}$ and a policy forcing the system to a desired goal. As geometric fabrics remain a local motion planning method, it is dependent on global guidance to avoid deadlock scenarios and local minima. In the context of grasp and motion planning, we ensure this global guidance via a reactive grasp and motion generation framework.

III. Reactive Grasp and Motion Generation

Building on work by Ichnowski et al. [5], we assume that the superior perception system produces a grasp pose candidate denoted as $\boldsymbol{p}^-$. In the case of parallel-jaw grippers, the grasp pose defines the axis that connects the two grasp points, around which we can formulate an additional degree of freedom (DoF). Moreover, this can be expressed as rotation matrix $R_{\text{DoF}}(\theta)$ providing a continuous interval $\mathcal{P}$ around the original constant grasp pose,

$$\mathcal{P} = \{\boldsymbol{p}_i | \boldsymbol{p}_i = R_{\text{DoF}}(\theta)\boldsymbol{p}^-\}, \tag{2}$$

which can be formulated as a constraint in a nonlinear mathematical program. Here, θ specifies the angle of rotation, whose range depends on a given task. The decision variables of the optimization problem are formulated as waypoints $\boldsymbol{q}_t \in \mathcal{C}^n$, where n is the number of DoF. These waypoints are expressed in the joint configuration space, where the final waypoint $\boldsymbol{q}_T$ represents the joint configuration for reaching the grasp pose. The resulting NLP formulation also enables the incorporation of mechanical limits, collision constraints, and task constraints as follows,

$$\min_{\boldsymbol{q}_{0:T}} \quad f(\tau) \tag{3a}$$

$$\text{s.t.} \quad \text{FK}(\boldsymbol{q}_T) \in \mathcal{P} \tag{3b}$$

$$g(\boldsymbol{q}(t)) \leq 0, \qquad \forall t \in [0, T] \tag{3c}$$

$$h(\boldsymbol{q}(t)) = 0, \qquad \forall t \in [0, T] \tag{3d}$$

where the goal is to find a sequence of the robot's joint configurations $\tau = (\boldsymbol{q}_0, \boldsymbol{q}_1, ..., \boldsymbol{q}_T)$, which minimizes the control cost $f(\tau)$ while satisfying the constraints. Equation (3b) imposes that the result of the forward kinematics $\text{FK}(\boldsymbol{q}_T)$ for the last waypoint $\boldsymbol{q}_T$ needs to be in the grasp pose interval defined in Eq. (2). The robot's motion is obtained under the inequality and equality constraints in Eq. (3c)-(3d) respectively, satisfying joint limits and collision avoidance on the discrete path τ.

Solving the full grasp and trajectory optimization problem from Eq. (3) in a receding horizon fashion is costly, especially for systems with high DoF. However, by leveraging fabrics as a local motion planner, we can embed the majority of constraints within the designed geometry. Consequently, the NLP can be simplified in terms of constraints and sparse waypoints, allowing it to be used effectively as a sparse global planner that considers a full manipulation horizon, while still achieving real-time performance.

A. Objective function

The objective function in Eq. (3a) is the quadratic control cost function that penalizes the sum of squared accelerations along the trajectory expressed by joints' positions.

$$f(\tau) = \frac{1}{2}\bar{\boldsymbol{q}}^T \boldsymbol{Q} \bar{\boldsymbol{q}}, \tag{4}$$

where $\bar{\boldsymbol{q}}$ is a one-dimensional stacked vector of waypoints within τ. $\boldsymbol{Q}$ is a positive semidefinite matrix expressed as $\boldsymbol{Q} = \boldsymbol{A}^T\boldsymbol{A}$, with $\boldsymbol{A}$ being a finite differencing matrix used to compute joint accelerations $\ddot{\boldsymbol{q}} = \boldsymbol{A}\boldsymbol{q}$. This objective function enforces smooth motions to achieve feasible trajectories, with the joint limits for the first waypoint $\boldsymbol{q}_0$ set to the current state of the robot.

B. Collision avoidance

In order to estimate the feasibility of the grasp pose and avoid deadlocks caused by the multi-agent environment, collision avoidance is embedded into the optimization problem. Since fabrics facilitate whole-body collision avoidance, it is sufficient only to consider the mobile base link and the wrist link of the manipulator for the NLP. Collision avoidance is expressed as inequality constraints, where these links and obstacles are approximated by spheres in the form,

$$||\boldsymbol{s}_{\text{link}} - \boldsymbol{s}_{\text{obst}}||_2 \geq r_{\text{link}} + r_{\text{obst}}, \tag{5}$$

where $\boldsymbol{s}_{\text{link}}$ and $\boldsymbol{s}_{\text{obst}}$ are the center position of spheres of the links of the robot and obstacles respectively and r_{link} and r_{obst} are their radii [2].

C. Grasp constraint

The grasp constraint in Eq. (3b) is expressed as enforcing the angle θ between two normalized vectors $\boldsymbol{a}_{\text{unit}}$ and $\boldsymbol{b}_{\text{unit}}$ to be within a given interval. Let's assume that vector $\boldsymbol{a}$ and $\boldsymbol{b}$ are expressed in the same frame, then the constraint has the form:

$$\cos(\theta_{\text{upper}}) \leq \boldsymbol{a}_{\text{unit}}^T \boldsymbol{b}_{\text{unit}} \leq \cos(\theta_{\text{lower}}). \tag{6}$$

To leverage the interval specified by $\mathcal{P}$ in Eq. (2), we constrain two pairs of normalized rotational axes. Firstly, one of the end-effector axes has to align with the axis of rotation defined by a grasp candidate. Secondly, the angle between the other end-effector axis and the grasp candidate must be within the interval specified by θ. The selection of specific vectors depends on the given task. For cylindrical objects, such as cups, or when using suction grippers, we can formulate the problem to only require alignment of the axis in the first step.

D. The RGF framework

The solution to the grasp planning problem, formulated by the aforementioned objective and constraint functions and expressed in the end-effector frame, is shown in Fig. 2. Even though the grasp planner computes the sequence τ, currently only the final configuration $\boldsymbol{q}_T$ is used for fabrics, as reference tracking will be addressed in future work. Nevertheless, since the objective function binds these waypoints together, considering collision avoidance for these sparse waypoints helps

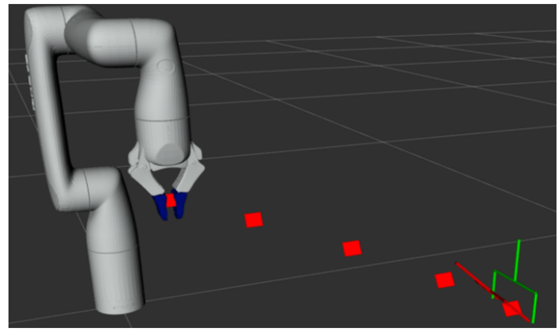


Fig. 2: Grasp planning optimizes the grasp candidate (green), which is initially unreachable, by leveraging the DoF around the grasp candidate. This results in an optimized grasp pose (red) and a sparse trajectory (red squares).

to compute a better grasp pose. The final waypoint $\boldsymbol{q}_T$ of the sequence τ is mapped to its corresponding end-effector pose $\boldsymbol{x}_{\text{ee}}^{\text{ref}}$ via the forward kinematics and integrated into the potential function $\psi(\boldsymbol{x})$ where $\boldsymbol{x} \in \mathcal{X}$ is the distance between the current end-effector pose $\boldsymbol{x}_{\text{ee}}$ and the subgoal $\boldsymbol{x}_{\text{ee}}^{\text{ref}}$, within its respective task space $\mathcal{X}$.

IV. Experiments

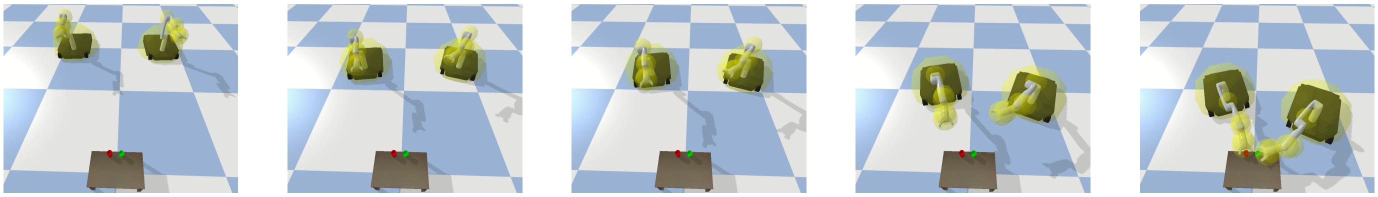
A. Experimental setup

The presented RGF framework is applied both in simulation and with real hardware. In both settings, we used a 9-DoF mobile manipulator consisting of a Clearpath Dingo holonomic mobile base and a Kinova Gen3 Lite 6-DoF manipulator. The grasp planning optimization problem was solved using the nonlinear optimizer IPOPT [16] within the symbolic framework CasADi [1]. In all experiments, the grasp planner operates at a frequency of 10 Hz, whereas geometric fabrics operate at 25 Hz. To detect the poses of obstacles, objects, and the mobile base, we employ the Vicon motion capture system. However, this can be replaced by any grasp analysis method, such as Mahler et al. [7]. A laptop with an Intel Core i7 processor and 16 GB of RAM is used to run the grasp planner and simulations, while in real-world experiments, fabrics and low-level controllers are executed on the NVIDIA AGX Orin controller within the Dingo.

B. Avoiding deadlocks in multi-agent settings

In the simulation experiments, we focus on a task where multiple robots work in close proximity to pick objects from a table as illustrated in Fig. 3. We compare the performance of vanilla fabrics to our method RGF where fabrics are enhanced with a reactive grasp planner. Both algorithms are implemented in a decentralized manner, meaning each robot operates without knowledge of the intentions of the others. Consequently, each robot treats the others as obstacles.

For collision avoidance, fabrics are responsible for managing collisions for all links on the robots, while the grasp planner only considers collision spheres on the chassis links of the mobile bases and the end-effector links, using larger radii than fabrics. At the beginning of the manoeuvre, the



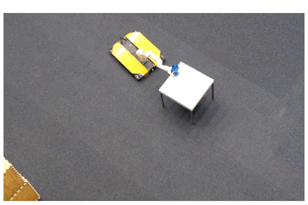
(a) (b) (c) (d) (e)

Fig. 3: Selected time frames for resolving deadlocks in a multi-robot scenario. As the robots approach each other, the grasp planners of both robots independently adapt their respective grasp poses, in contrast to vanilla multi-agent fabrics as in Fig. 1.

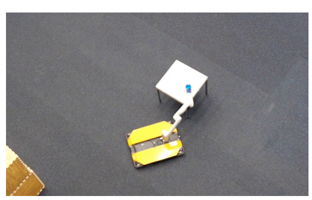
grasp pose for both robots is set identical to the initial constant grasp pose. As the robots approach each other, the considered collision avoidance within the reactive grasp planner modifies the grasp pose, leveraging the additional DoF expressed by the grasp constraint. Without the reactive grasp planner, the robots approach the objects but are not capable of grasping the objects because of the deadlock as shown in Fig. 1a. The proposed RGF is able to resolve this deadlock, Fig. 1b, and allows for efficient navigation in decentralized multi-agent scenarios.

C. Real world experiments

In the experiments with real hardware, we use RGF in a single-robot setup for two distinct scenarios: an environment with static obstacles and an environment with a dynamic obstacle, specifically a human. In the first experiment, Fig. 4, the task is to pick the object from the table without colliding with the table. The grasp planner is capable of adapting the desired end-effector pose based on the location of the object relative to the table, since collision avoidance is included (Section III-C). If the object is placed on the opposite side of the table, the grasp planner guides the fabrics to grasp the object from that side. If the environment includes more static obstacles, such as a standing human, additional waypoints within the optimization formulation help modify the optimized grasp pose (Section III-D). Obstacle avoidance along the path is ensured by fabrics.



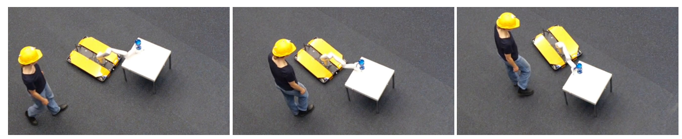
(a) Reachable grasp candidate



(b) Optimized grasp pose

Fig. 4: Adapting the grasp pose in the presence of static obstacles: Whole-body optimization considers potential collisions with the table, depending on the object's pose. In Fig. 4a, the solver prefers a grasp pose that is closer to the robot's current state, whereas in Fig. 4b, the grasp pose is modified to avoid a collision with the table.

The aim of the second experiment is to test the adaptability of the proposed RGF framework in the presence of a human moving around the table. Using only vanilla geometric fabrics, the robot adjusts its current joint configuration to avoid the human while attempting to reach the stationary grasp candidate. However, this adjustment results in collisions with pickable objects, thereby violating the manipulation objective. RGF resolves this issue by adapting the desired $\boldsymbol{x}_{\text{ee}}^{\text{ref}}$ for fabrics, as shown in Fig. 5.



(a) (b) (c)

Fig. 5: Selected time frames of adapting the grasp pose in the presence of a dynamic obstacle. In the first frame (5a), the solver detects a human and immediately modifies its grasp pose. Throughout the human's movement (5b), the robot continues to adjust its grasp pose. Finally, when the human stops (5c), the robot stabilizes its grasp.

V. Conclusion and future work

This work presents a framework for reactive grasp and motion generation that adapts the grasp pose in real time to avoid potential collisions and deadlocks caused by the presence of other agents and obstacles. By leveraging geometric fabrics, the grasp planner is capable of optimizing the grasp candidate in real-time. This capability was demonstrated on a 9-DoF mobile manipulator. The fast reactivity of the RGF framework allows it to execute the grasping task efficiently and resolve deadlock situations for multi-agent systems in a decentralized manner. The adaptivity of RGF was tested in both simulation and real-world experiments. However, the success rate of the grasp planner heavily depends on a correct initial guess, which could be improved by data-driven methods in the future. In subsequent work, we will consider more diverse objects in terms of shape, replace the motion capture system with grasp analysis models, and test multi-agent scenarios using real hardware.

Acknowledgments

This project has received funding from the European Union through ERC, INTERACT, under Grant 101041863, and from the Slovak Grant Agency VEGA 1/0436/22 and 07/TUKE/2024. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

References

[1] Joel A E Andersson, Joris Gillis, Greg Horn, James B Rawlings, and Moritz Diehl. CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1): 1–36, 2019. doi: 10.1007/s12532-018-0139-4.

[2] Simon Duenser, James M. Bern, Roi Poranne, and Stelian Coros. Interactive Robotic Manipulation of Elastic Objects. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3476–3481, 2018. doi: 10.1109/IROS.2018.8594291. ISSN: 2153-0866.

[3] Clemens Eppner, Arsalan Mousavian, and Dieter Fox. Acronym: A large-scale grasp dataset based on simulation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6222–6227. IEEE, 2021.

[4] Jeffrey Ichnowski, Yahav Avigal, Vishal Satish, and Ken Goldberg. Deep learning can accelerate grasp-optimized motion planning. *Science Robotics*, 5(48):eabd7710, 2020.

[5] Jeffrey Ichnowski, Michael Danielczuk, Jingyi Xu, Vishal Satish, and Ken Goldberg. GOMP: Grasp-Optimized Motion Planning for Bin Picking. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5270–5277, May 2020. doi: 10.1109/ICRA40945.2020.9197548. ISSN: 2577-087X.

[6] Jeffrey Ichnowski, Yahav Avigal, Yi Liu, and Ken Goldberg. GOMP-FIT: Grasp-Optimized Motion Planning for Fast Inertial Transport. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 5255–5261, May 2022. doi: 10.1109/ICRA46639.2022.9812387.

[7] Jeffrey Mahler, Matthew Matl, Vishal Satish, Michael Danielczuk, Bill DeRose, Stephen McKinley, and Ken Goldberg. Learning ambidextrous robot grasping policies. *Science Robotics*, 4(26):eaau4984, January 2019.

[8] Douglas Morrison, Peter Corke, and Jürgen Leitner. Learning robust, real-time, reactive robotic grasping. *The International journal of robotics research*, 39(2-3):183–201, 2020.

[9] Rhys Newbury, Morris Gu, Lachlan Chumbley, Arsalan Mousavian, Clemens Eppner, Jürgen Leitner, Jeannette Bohg, Antonio Morales, Tamim Asfour, Danica Kragic, Dieter Fox, and Akansel Cosgun. Deep learning approaches to grasp synthesis: A review. *IEEE Transactions on Robotics*, 39(5):3994–4015, 2023. doi: 10.1109/TRO.2023.3280597.

[10] Nathan Ratliff and Karl Van Wyk. Fabrics: A foundationally stable medium for encoding prior experience. *arXiv preprint:2309.07368*, 2023.

[11] Nathan D. Ratliff, Jan Issac, and Daniel Kappler. Riemannian motion policies. *CoRR*, abs/1801.02854, 2018. URL http://arxiv.org/abs/1801.02854.

[12] Nathan D. Ratliff, Karl Van Wyk, Mandy Xie, Anqi Li, and Muhammad Asif Rana. Optimization Fabrics. August 2020. doi: 10.48550/arXiv.2008.02399.

[13] Samaneh Hosseini Semnani, Hugh Liu, Michael Everett, Anton De Ruiter, and Jonathan P How. Multi-agent motion planning for dense and dynamic environments via deep reinforcement learning. *IEEE Robotics and Automation Letters*, 5(2):3221–3226, 2020.

[14] Max Spahn, Bruno Brito, and Javier Alonso-Mora. Coupled mobile manipulation via trajectory optimization with free space decomposition. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12759–12765, 2021. doi: 10.1109/ICRA48506.2021.9561821.

[15] Max Spahn, Martijn Wisse, and Javier Alonso-Mora. Dynamic Optimization Fabrics for Motion Generation. *IEEE Transactions on Robotics*, 39(4):2684–2699, August 2023. ISSN 1941-0468. doi: 10.1109/TRO.2023.3255587.

[16] Andreas Wächter and Lorenz T Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106:25–57, 2006.