Honda Research Institute EU
Honda Research Institute Europe GmbH
https://www.honda-ri.de/

Partner Adaptive Dyadic collaborative Manipulation through Informed Hybrid Bilevel Optimization

Theodoros Stouraitis, Michael Gienger, Sethu Vijayakumar

2020

Preprint:

This is an accepted article published in IEEE Transactions on Robotics. The final authenticated version is available online at: https://doi.org/10.1109/TRO.2020.2992987 Copyright 2020 IEEE

Online Hybrid Motion Planning for Dyadic Collaborative Manipulation via Bilevel Optimization

Theodoros Stouraitis[⋆,•], *Student Member, IEEE,* Iordanis Chatzinikolaidis[⋆], *Student Member, IEEE,* Michael Gienger[•], *Member, IEEE,* and Sethu Vijayakumar[⋆]

***Abstract*—Effective collaboration is based on online adaptation of one's own actions to the actions of their partner. This article provides a principled formalism to address online adaptation in joint planning problems such as Dyadic collaborative Manipulation (DcM) scenarios. We propose an efficient bilevel formulation which combines graph search methods with trajectory optimization, enabling robotic agents to adapt their policy on-the-fly in accordance to changes of the dyadic task. This method is the first to empower agents with the ability to plan online in hybrid spaces; optimizing over discrete contact locations, contact sequence patterns, continuous trajectories, and force profiles for co-manipulation tasks. This is particularly important in large object co-manipulation that requires changes of grasp-holds and plan adaptation. We demonstrate in simulation and with robot experiments the efficacy of the bilevel optimization by investigating the effect of robot policy changes in response to real-time alterations of the dyadic goals, eminent grasp switches, as well as optimal dyadic interactions to realize the joint task.**

***Index Terms*—Physical Human-Robot Interaction, Optimization and Optimal Control, Manipulation Planning, Dual Arm Manipulation**

I. Introduction

With Dyadic collaborative Manipulation (DcM) we refer to a set of two individuals jointly manipulating an object, as shown in Fig. 1. The two individuals partner together to form a distributed system, augmenting their manipulation abilities. Such individuals can be either humans or robots. In scenarios where both individuals are humans, the collaboration is natural as we humans are adept at co-manipulation. One key element is our ability to understand our partner's intentions and adapt our actions accordingly. A second central skill is our ability to generate sequential manipulation plans. Nevertheless, our understanding of the mechanisms of joint action [1] and sequential decision making [2] are still subject of research.

Early work by Sheridan [3] identified eight core challenges of human-robot communication, with two of them being: (i) the need to acquaint both humans and robots with models of their partners, and (ii) the need to regulate the interaction of distributed decision-making systems, typically referred as *mixed initiative systems*. Ajoudani *et al.* [4] summarized the strategies used to equip robots with interaction capabilities and pinpointed that research on human-robot interaction models is still at its infancy. In this work we focus on how a robot policy can be partner aware and flexible towards complying with the requirements of DcM scenarios.

Authors with ⋆ are with the School of Informatics, University of Edinburgh, Edinburgh EH8 9AB, U.K. E-mail: theodoros.stouraitis@ed.ac.uk, i.chatzinikolaidis@ed.ac.uk, sethu.vijayakumar@ed.ac.uk

Authors with • are with the Honda Research Institute Europe (HRI-EU), Carl-Legien-Straße 30, 63073 Offenbach am Main, Germany. E-mail: michael.gienger@honda-ri.de

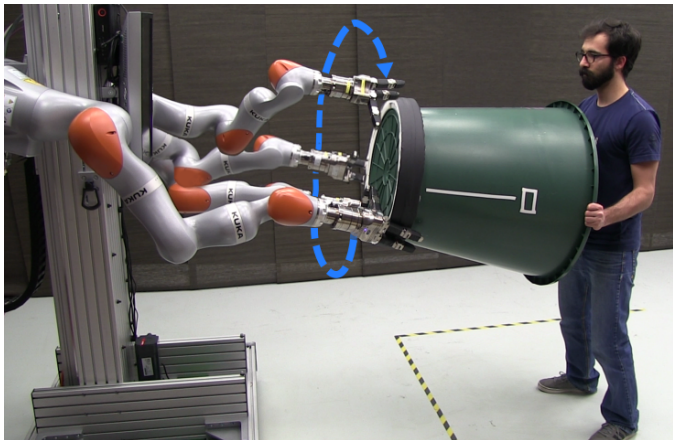


Fig. 1. DcM scenario which requires contact change with the left end-effector.

DcM scenarios demand a broad range of manipulation skills from both participants. The secret behind humans' remarkable manipulation skills, is our competence in control and prediction of *contact events* [5]. In this article, we address co-manipulation scenarios that involve multiple changes of contact, which is the crux of sequential manipulation [6].

As a typical DcM example, consider a robot transporting a large object with a human as shown in Fig. 1. During the task execution shown in Fig. 2, there are instances in which the current contact configuration is not sufficient for the continuation of the task, *e.g.* rotating the object upside-down (Fig. 2A to Fig. 2G). To avoid such deadlocks, both the human and the robot should anticipate the object's future state and change their contact locations accordingly, as shown in Fig. 2B and Fig. 2D. As illustrated in Fig. 2E and Fig. 2G, contact adjustments are crucial, as they result in failure or success of the task. Further, actions like contact changes must comply with the partner's actions to jointly balance the object.

Joint planning in DcM scenarios is extremely challenging and requires solution of the following four complex problems:

1) *Partner's intention estimation*: An agent can only contribute to the performance of the dyad, if an estimation of the partner's intention can be obtained.
2) *Joint action space planning*: As the two individuals act upon the same object their actions need to be coordinated with respect to the critical aspects of the task, *e.g.* balancing the object in collaboration with the partner.

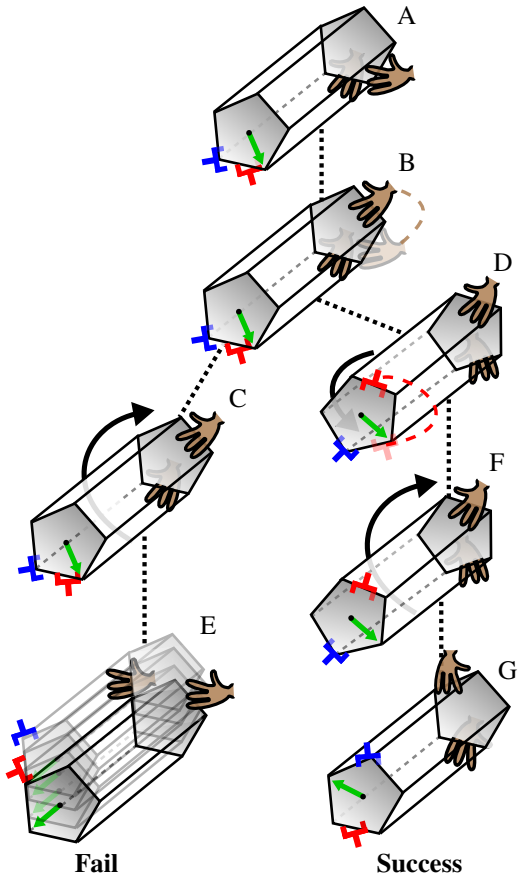


Fig. 2. Different action paths to collaboratively manipulate the object from the initial pose (A) to the goal pose (G). The tree illustration has two branches. The left branch (A-B-C-E), that does not involve any change of grasp from the robot side and results in an object drop. The right branch (A-B-D-F-G), where the robot changes grasp-hold at (D) after the human partner changed grasp-hold at (B), to successfully jointly rotate the object to the goal (G).

3) *Online motion plan generation*: Since the human partner's behavior is changing—*i.e.* non-stationary—a collaborative agent needs to update its own action plans on-the-fly according to the current goal and state of the interaction.
4) *Agent's hybrid policy*: The agent's repertoire of actions needs to be sufficiently rich to participate in DcM tasks. Such actions belong to a hybrid space of both continuous and discrete quantities, like forces and contact changes.

In this article we address the last three points with a novel bilevel optimization formulation—where a continuous optimization problem is embedded into a discrete one. We couple informed-graph-search methods and trajectory optimization to efficiently compute online hybrid motions with high fidelity. The resulting motion plans can be updated on-the-fly and incorporate both geometrical and physical couplings between the individuals of the dyad. Partner's intentions are represented as task space goals and here we assume that the intentions can be predicted. The partner's policy is abstracted as task space wrenches, which enables us to model joint manipulation. The proposed method enables robots to generate on-the-fly joint action policies that are partner-aware and can benefit from the breadth of the hybrid action space.

The contributions of this article can be summarized as:

1) ***Partner-aware dyadic planning formalism****:* We extend the joint planning formalism—introduced in our previous work [7]—to non-stationary partner behaviors. Using this, the problem of finding the appropriate actions to co-manipulate the object can be addressed given an estimate of the partner's variable intentions. This formalism serves as a principled basis for the development of partner-aware joint-action method.

2) ***Bilevel computational formulation****:* Our bilevel optimization formulation enables the combination of graph-search methods with trajectory optimization methods in one framework. The former provides a coarse solution which is refined by the latter. This combination allows us to efficiently explore the discrete modes of the problem, *e.g.* the contact state of an end-effector, and holistically reason about geometric and dynamic properties, *e.g.* contact locations, forces and timings.

3) ***Hybrid optimal control for multi-contact planning****:* We present a holistic model-based optimization method that allows robotic agents to treat concurrently (i) forces, (ii) contact locations, (iii) actions timings, (iv) object's trajectory, and (v) contact sequence pattern, towards obtaining an optimal solution for multi-contact tasks. To do so, we introduce a set of hybrid motion primitives that enable our method to generate hybrid plans without a pre-specified contact pattern as in [7].

4) ***Online dyadic planning****:* By combining the above three with the partner model introduced in our work [7] and an informed search planner, we realize a computationally efficient optimization method for DcM setups. The method generates hybrid motion plans online that are capable of adapting on-the-fly to changes during the task, like dyadic goal changes.

This article is organized as follows: Section II reviews related work on DcM setups and hybrid motion generation. Section III presents the problem formulation. Brief background on methods is provided in Section IV, while the methods' details are given in Section V, Section VI and Section VII. Section VIII presents the evaluation of the method and the experimental results. Finally, Sections IX and X discuss promising research directions and conclude this article. Overall, our work addresses the problem of hybrid action generation towards complying with the joint planning requirements in DcM scenarios. We demonstrate that such actions can be computed efficiently, enabling online motion re-planning with respect to dyadic goals and interactions.

II. Related work

A. Human-Robot Collaboration (HRC) components

1) ***Partner's policy prediction****:* In [8], [9] a confidence measure of the human's goal prediction was used to alter between reactive and proactive robot behaviors. Multi-class classifiers were utilized in [10], [11] to recognise human partner's commands through force interaction in co-manipulation tasks. Further, conditional random fields were adopted in [12] to infer the human's intended goal during box co-pushing tasks and in [13] to anticipate object related human activities. In [14] learning-based human's occupancy workspace was predicted to generate robot collision free motion. A comprehensive literature on modelling other agents in multi-agent environments

can be found in [15]. *These methods predict the partner's intent, however the adaptation of the agent's policy is only realised as a selection from a set of discrete behaviors or as a modification of few continuous variables*.

2) ***Dyadic interaction models****:* Interaction models are separated in three prevailing schools of thought, described below.

Control focused: Agravante *et al.* [16] used impedance control to accommodate partner's actions and collaboratively carry a table with a human. A load sharing framework with predefined sharing modes was presented in [17]. *In both cases, the partner is treated as an external disturbance to the system.*

Coupled-policies focused: Maeda *et al.* [18] proposed a method to transfer adaptive hand-overs to robots from kinesthetic demonstrations. Similarly, in [19] force and vision information was employed to commence the appropriate learned impedance behaviors depending on the task's phase. Data-driven extraction of interaction constraints during hand-over tasks was proposed in [20]. The extracted constraints were then used to form online robot responses. *These methods couple together the policies of the agent, the partner, and the task evolution, to learn a direct mapping towards generating online adaptive robot responses. Thus, their generalization capabilities are limited to the demonstration set.*

Partner-model focused: Maeda *et al.* [21] used a polynomial model to predict human motion and update the robot's goal. In [22], [23] a task model was learned offline, to guide the interaction at the reproduction phase. Evidence of humans utilizing a model-based prediction of the partner's goal to update their own task's goal and consequently their own policy have been presented in [24]. *Such methods are elegant, as each entity (partner, agent, task) of the interaction is modelled separately and actions of the two individuals can be appropriately reasoned upon. Our work follows the same principle to obtain generalizable robot behaviors.*

3) ***Agent's policy generation****:* Another central aspect to HRC are motion attributes that the robot can regulate to fulfill the task in collaboration with the human. In [25] task space attributes, *e.g.* the object's trajectory, were optimized to facilitate human ergonomics. In [26] adaptation during co-manipulation was realized through turn-taking collaboration. Further, a number of methods focus on the dynamic properties of the interaction. Inverse dynamics approaches concentrated on the torque and force regularization [27], [28], while others adapted the impedance characteristics of the robot online to accommodate for partner's actions [22], [29]. However, a central aspect of manipulation is the selection of the appropriate contact locations on the object [30]. Accordingly, the exploitation of the contact space of the object by the individuals is vital in DcM scenarios. *To the best of our knowledge, contact adaptation within collaborative manipulation scenarios has not been addressed yet, although it is crucial towards enabling robot perform complex DcM tasks jointly with a partner*.

B. Hybrid Motion with contact changes

Motivated to enhance robots with multi-contact manipulation capabilities, here we review methods used to generate hybrid motions.

1) ***Multi-contact planar manipulation****:* Mason introduced the problem of planar non-prehensile manipulation, the *motion cone* concept, and the voting theorem [31]. The *limit surface* concept was introduced in [32] and used in [33] to model the dynamics of planar pushing. These concepts map contact point's motion to object's motion, and have been used [34]–[36] to address planning and control for planar pushing. Also, recently they were generalized to a broader set of planar tasks [37]. *Yet the different contact modes are typically explored with offline sampling, and the quasi-static environment assumption limits their applicability to 2D tabletop pushing*.

2) ***Hybrid planning and control****:* According to an important duality between manipulation and locomotion, the latter is an instantiation of non-prehensile manipulation [38]. Our work is inspired by model-based optimal control methods [39]–[42], that are not restricted by a quasi-static stability assumption. Next, we describe three hybrid motion generation approaches.

Hierarchical approaches: These approaches address hybrid problems by decomposing them into action planning [43], contact planning [44], and motion control [45]. Such hierarchies allow to exploit domain knowledge at the task planning level and have been used for online motion generation. Yet, as these elements are designed separately it is usual that the final solution is not optimal or sometimes not feasible. *In contrast, our method treats all the variables of the problem holistically*.

Mixed-integer programming: This formulation explicitly models the hybrid nature of the problem and has been used by [46]–[48] for both locomotion and planar manipulation. Yet mixed-integer methods need to explore both the continuous and discrete parts of problems, while reasoning for the discrete part is done using general combinatorial optimization methods like *Branch and Bound*. *This typically leads to large computation times that can be prohibitive for DcM needs*.

Continuous programming: Using continuous optimization, in [41] a mathematical problem with complementarity constraints was formulated in the presence of complex contact phenomena. In [40] and [39] smooth nonlinear optimization problems were formulated based on a key observation: Motions through contacts have phases, while the contact set remains invariant within each phase. One of their drawbacks is that the motion can only be conditioned on physical properties of the problem neglecting higher-level task objectives that are common in DcM scenarios [49], [50]. *Our formulation treats all the variables as continuous too and utilizes a graph search method to consider higher-level task objectives too*.

3) ***Sequential manipulation planning****:* On a different line of work, Simeon *et al.* [51] employed probabilistic roadmaps to produce motion plans with multiple grasp-hold changes. King *et al.* [52] used Monte Carlo Tree Search to plan sequences of discrete pushes and reason about object interactions. The A* algorithm was used by Gienger *et al.* [53], to demonstrate DcM scenarios with a human and a robot. These methods discretize the state space to employ search algorithms. Further, combinations of these methods with motion planners have been realized [54]. *Nevertheless, to the extend of our knowledge these methods are limited to kino-dynamic planning and have not been applied in hybrid problems, where the generation of dynamically feasible plans is of core importance*.

Logic geometric programming (LGP): LGP with physics synthesizes logical planning with optimal control to demonstrate a broad range of robot sequential manipulation planning capabilities. LGP has also been used for multi-agent cooperative manipulation tasks [55], such as handovers. The cooperative aspects are limited to the kinematic domain, where both agents act synchronously, but their actions are not physically coupled. In parallel to LGP with physics, the proposed method combines the benefits of informed-graph-search algorithms and optimal control formulations. Informed search methods such as A* are a more efficient special case of the *Branch and Bound algorithm* [56], a fact that makes them well suited for online motion adaptation during DcM tasks. *Toussaint's work [57] exhibits creative solutions for manipulation puzzles in simulation, while our method considers dynamic aspects of the dyadic interaction and enables online re-planning*.

III. Problem Formulation

Fig. 3 provides a graphical representation of DcM as a system. It is separated into three components; (i) the partner's policy, (ii) the dyadic interaction, and (iii) the agent's policy, similarly to Section II-A. Key in this formulation is the dyadic interaction, which is used to capture the binding between the two individuals, both in physical and in mental terms. The physical pairing arises due to the object, that acts as the physical medium for exchanging information, while the intentions of the individuals are naturally correlated due to the common task of the dyad. The exact pairing configuration—*e.g.* role distribution within the dyad—can be regulated through the physical constraints and the dyadic objectives.

A. Core Nomenclature

Here, we introduce the core notation used in this article, while section specific symbols are defined in each section. We use superscripts, k for the indexing of the agent's k_{th} end-effectors' quantities, as well as a and p, to refer to the same quantity for the agent and the partner, respectively. We use subscripts to denote both time and indices along a sequence.

$n \in \mathbb{N}$	Dimensionality of c-space (partner, agent)
$\nu \in \mathbb{N}$	Dimensionality of manipulation task
$N \in \mathbb{N}$	Total number of knots[1]
$i \in \mathbb{N}$	Knot indices of the transcribed problem
$j \in \mathbb{N}$	Indices of the graph-search problem
$K \in \mathbb{N}$	Total number of agent's end-effectors
$T \in \mathbb{R}_{>0}$	Total motion duration (final time)
$\mathbf{f}^k \in \mathbb{R}^\nu$	Forces applied by agent's k_{th} end-effector
$\boldsymbol{\lambda} \in \mathbb{R}^\nu$	Partner's applied wrench
$\mathbf{c}^k \in \mathbb{R}^\nu$	Agent's k_{th} end-effector position
$\mathbf{K}, \mathbf{D} \in \mathbb{R}^\nu$	Stiffness and damping (partner, agent)
$\mathbf{q} \in \mathbb{R}^n$	Configuration (partner, agent)
$\mathbf{y}_{t:T} \in \mathbb{R}^{\nu \times N}$	Pose trajectory of the object
$\boldsymbol{\Delta T} \in \mathbb{R}^N$	Agent's action timings

B. Formulation

The full control policy π^a of an agent participating in DcM tasks is defined as function $\pi^a(\cdot) \rightarrow \xi$. ξ is defined as

$$\xi = \begin{bmatrix} \mathbf{f}_i^k & \mathbf{c}_i^k & \boldsymbol{\Delta T}_i & \mathbf{K}_i^a & \mathbf{D}_i^a & \mathbf{q}_i^a \end{bmatrix} \ \forall i \in \mathbb{N}, \tag{1}$$

and denotes a trajectory. However, as this work aims to generate hybrid motion plans that belong in the force-contact space, the output of the agent's policy can be reduced to

$$\xi = \begin{bmatrix} \mathbf{f}_i^k & \mathbf{c}_i^k & \boldsymbol{\Delta T}_i \end{bmatrix} \ \forall i \in \mathbb{N}. \tag{2}$$

A trajectory ξ is a time-indexed sequence of actions, that guide the object to the goal state $\mathbf{x}_T$ given its current state $\mathbf{x}_t$, with $\mathbf{x}_t = \begin{bmatrix} \mathbf{y}_t & \dot{\mathbf{y}}_t \end{bmatrix}^T$. In the top right of Fig. 3, we illustrate few trajectories (grey) from all feasible ones, as well as the optimal one (black) with ξ^*.

$$\mathbf{x}_{t:T} = f(\pi^a, \pi^p) \tag{3}$$

The actual task, *i.e.* the object's motion, is a function of the two individuals' policies described by (3). Thus, the policies of the two individuals are coupled, forming the dyadic interaction. We represent this relationship with

$$\xi = \pi^a(\mathbf{x}_t, \hat{\pi}^p, \theta^{\mathcal{D}}, \theta^{\mathcal{M}}), \tag{4}$$

which indicates the dependency of the agent's policy to the estimated policy of the partner $\hat{\pi}^p$, the parameters of both the dyadic setup $\theta^{\mathcal{D}}$ and the manipulation task $\theta^{\mathcal{M}}$. As shown in Fig. 3, the estimation of the partner's policy can be obtained based on a set of sensory measurements, an intention estimation process and a parametric model. In the proposed DcM formulation, the parametric model of the partner's policy dependents on the state of the object, it outputs task-space wrenches and can be described by a set of parameters θ^p, formally written as

$$\boldsymbol{\lambda} = \hat{\pi}^p(\mathbf{x}_t; \theta^p). \tag{5}$$

To comply with the sequential nature of DcM tasks, the policy of the partner should be non-stationary. This can be represented with a multi-modal probability distribution $Pr(\theta^p | \mathbf{x}_t, \mathbf{q}_t^p, H^p)$ over parameters θ^p, given the sensed data $\mathbf{x}_t, \mathbf{q}_t^p$ and a history of partner's actions H^p. In every instance of the dyadic interaction the partner's policy is described by one of the modes of the distribution as shown in top left of Fig. 3. This can be obtained by an intention estimation process.

The aim in DcM scenarios is to obtain the agent's optimal policy that depends on the parameters of the dyadic setup, of the manipulation task and the current estimate of partner's policy. We express partner-aware dyadic planning with

$$\begin{aligned} \xi^* = \min_{\pi^a} & \int_0^T c(\pi^a, \hat{\pi}^p, \theta^{\mathcal{D}}, \theta^{\mathcal{M}}) dt \\ \text{s.t.} \ & \boldsymbol{g}(\pi^a, \mathbf{x}_t, \hat{\pi}^p, \theta^{\mathcal{M}}) \leq 0, \end{aligned} \tag{6}$$

by introducing the idea of considering the partner's actions into the motion plans of the agent through the constraint functions $\boldsymbol{g}(\cdot)$. As the partner's behaviour is non-stationary, the parameters θ^p of the partner's policy π_p need to be estimated

[1] Knots are the discretization points of the transcribed continuous problem.

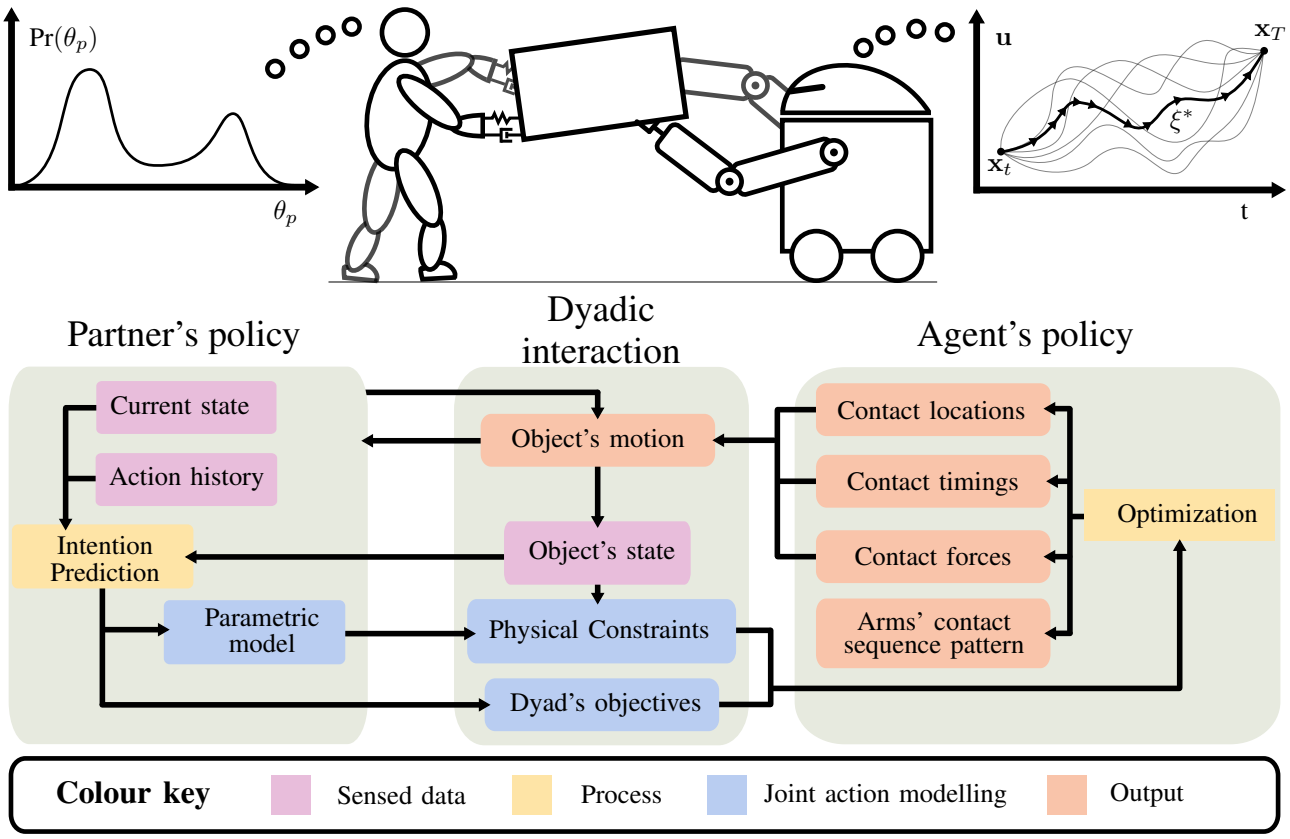


Fig. 3. A typical DcM scenario along with a modular description of DcM as a system. On top left a multi-modal distribution describes the policy of the partner and top right illustrates the optimal trajectory ξ^* computed from the policy of the agent along with few other feasible trajectories. The exact definitions of $\mathbf{u}$ and $\mathbf{x}$ are given in Section VI.

repeatedly during the interaction to provide $\hat{\pi}^p$, which will trigger an update of π^a. Further, the objectives of the dyad, such as $\theta^{\mathcal{D}}$, are met through the cost function $c(\cdot)$, where $\theta^{\mathcal{D}}$ can represent *i.e.* the role assignment within the dyad. Additionally, the task specifications can be satisfied either through the the cost function $c(\cdot)$ or the constraints $\boldsymbol{g}(\cdot)$, where $\theta^{\mathcal{M}}$ may define *i.e.* the final pose of the object or a constant linear/angular velocity of the object.

IV. Background

A. Hybrid motion preliminaries

As described in Section III, the policy of the agent π^a generates hybrid action trajectories, that guide the object from the current state to the goal. We illustrate one such trajectory in Fig. 4, where the object's pose $\mathbf{y}_t$, the end-effectors' positions $\mathbf{c}^k$ and the contact force $\mathbf{f}^l$ of the left end-effector are visualized. Such trajectories have hybrid nature due to the contact change. The elements we would like to highlight in Fig. 4 are: (i) critical transition instances exist within the trajectory, where *discontinuities* occur, *e.g.* the force at T_1 and T_3, (ii) according to these time-instances, the motion can be separated in phases—called *contact-invariant* phases, *e.g.* contact and swing phase, and (iii) the sequential arrangement of these phases defines the outline of the trajectory—which we refer to as *structure of the motion* and we denote with $\mathbf{H} \in \{0,1\}^{K\times N}$. In manipulation setups the *structure of the motion* specifies the arms' contact sequence pattern, *i.e.* the order with which the arms change contacts.

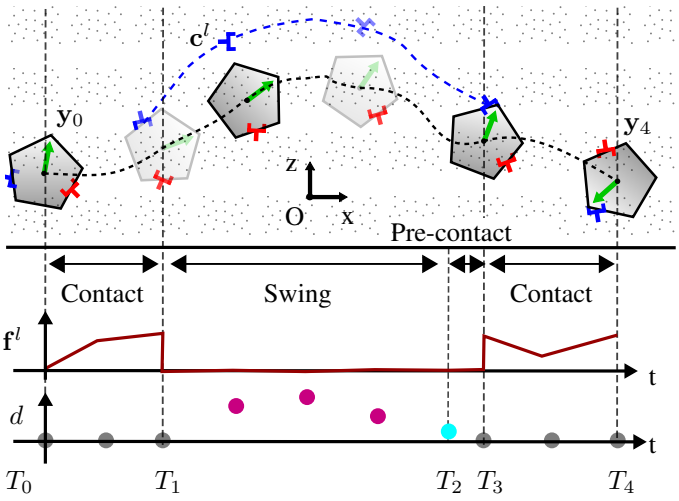


Fig. 4. Hybrid motion plan with one grasp-hold change, separated into phases. The grey dotted area on top illustrates the physical space (x,z,ϕ). Orientation ϕ is illustrated with the green arrow in the object. The force $\mathbf{f}^l$ applied by the left (blue) end-effector is shown with the middle plot. The knots of the trajectory with resolution 3 are shown in the bottom graph along with the contact distance d of the left end-effector to the object's surface. The contact knots are grey, the swing knots are pink, and the pre-contact knot is cyan. It is worth mentioning that all the quantities shown here are optimized.

B. Graph search algorithms preliminaries

Search algorithms are used to find paths within graphs [58]. A graph $G = (V, E)$ is described with a set of nodes V and set of directed edges E. Each edge $e^{pq} \in E$ denotes a directed link between node $v^p \in V$ and $v^q \in V$, where v^q is a *successor* of v^p. In the context of search algorithms; i) each edge e^{pq} has an associated cost $c^{pq} > 0$, ii) the graph can be obtained given a set of initial nodes $\{v^\iota\} \subset V$ and a successor operator Γ. Γ is defined on the set V and when applied on a node v^ι provides all its directed edges to *successor* nodes with the respective costs. Finally, given a node v^s and the successor operator Γ, a subgragh $G^s = (V^s, E^s)$ can be constructed such that all the nodes of G^s define the *accessible* set of nodes from node v^s.

Graph search algorithms address the problem of finding the optimal path[2] $\mathbf{v} = \{v_0, ..., v_{\mathcal{N}}\}$ of length $\mathcal{N} \in \mathbb{N}$ from a start node v^s to a set of goal nodes $\{v^g\}$, expressed as

$$\begin{aligned} \min_{\mathbf{v}} \quad & \mathcal{C}(\mathbf{v}) && \text{(7a)}\\ \text{s.t.} \quad & v_{j+1} = f_\gamma(v_j, e_j) && \text{(7b)}\\ & e_j \in \Gamma(v_j) && \text{(7c)}\\ & v_0 = v^s && \text{(7d)}\\ & v_J \in \{v^g\} && \text{(7e)} \end{aligned}$$

where $\mathcal{C}(\mathbf{v}) \in \mathbb{R}$ is the total cost along the path, (7c) indicates the set of all directed edges starting from a given node, $f_\gamma(\cdot)$ is the transition function responsible for computing the next node in the sequence, and (7d), (7e) specify the initial node and the set of final nodes, respectively [58]. The specifics for our problem regarding $\mathcal{C}$, Γ, f_γ and the representation of v are discussed in Section V-A. These methods compute efficiently a sequence of transitions, *i.e.* the *structure of the motion*. Yet, they neglect details of the actual continuous motion through the transitions.

C. Trajectory optimization preliminaries

Trajectory optimization (TO) addresses the problem of finding locally optimal trajectories for dynamical systems [59], [60]. We consider the following optimal control problem

$$\begin{aligned} \min_{\boldsymbol{x}(t), \boldsymbol{u}(t)} \quad & \int_0^T c\left(\boldsymbol{x}(t), \boldsymbol{u}(t)\right) dt + c_f\left(\boldsymbol{x}(T)\right) && \text{(8a)}\\ \text{s.t.} \quad & \dot{\boldsymbol{x}}(t) = \boldsymbol{f}\left(\boldsymbol{x}(t), \boldsymbol{u}(t)\right) && \text{(8b)}\\ & \boldsymbol{x}(0) \in \mathbb{X}_0 && \text{(8c)}\\ & \boldsymbol{x}(t_f) \in \mathbb{X}_f && \text{(8d)}\\ & \boldsymbol{g}\left(\boldsymbol{x}(t), \boldsymbol{u}(t)\right) \in \mathbb{Z} && \text{(8e)}\\ & t \in [0, T], && \text{(8f)} \end{aligned}$$

where η is the dimensionality of a general system, $\boldsymbol{x} \in \mathbb{R}^\eta$ is the model's state vector, $\boldsymbol{u} \in \mathbb{R}^\eta$ is the model's control vector, $c(\cdot), c_f(\cdot) \in \mathbb{R}$ in (8a) are the running and final cost functions, $\boldsymbol{f}(\cdot) \in \mathbb{R}^\eta$ in (8b) describes the system's dynamics, and (8c) to (8f) describe bounds on the initial state, final state, path constraints and motion duration, respectively.

[2]We use superscripts to index nodes and edges in a time agnostic fashion and subscripts to index the nodes in the optimal path sequence, as in Section III-A.

Description (8) belongs to a rather general class of optimization problems—termed Infinite Programming problems—since we seek to find a set of continuous functions that fulfill a set of continuous constraints. To make such problems computationally tractable, the usual approach is to parameterize the problem using a finite number of decision variables, *i.e.* express the problem as a constrained parameter optimization problem. In this work we express the hybrid problem utilizing *direct transcription* (using a *trapezoidal integration* rule); we further specify our problem's structure in Section V-B and Section VI, while more details on TO methods and transcription methods are provided in Appendix A. These methods are used to compute efficiently continuous motion plans through *discontinuities*, and typically require a proper initial seed.

V. Bilevel Optimization

In this section we provide the core computational formalism, that enables on-the-fly generation of hybrid motion plans, both for single agent manipulation planning and for joint manipulation planning in dyads. First, we describe how graph search (GS) algorithms can be formally combined with trajectory optimization (TO) methods. The former is the outer level and the latter is the inner level of the bilevel optimization. Next, we present the details of the outer and inner levels. The schematic shown in Fig. 5 illustrates the interplay between the outer and inner level, and reveals the nested structure of the inner level.

Bilevel formulation: The aim of this formalism is to provide the means to generate hybrid trajectories, like the one shown in Fig. 4. Motivated by the key observations mentioned in Section IV and inspired by the bilevel method presented in [61] as well as the "Mixed-Logic Program" [57], we combine the two formulations presented in (7) and (8) into a single bilevel optimization formulation, as follows

$$\begin{aligned} \min_{\mathbf{v}} \quad & \mathcal{C}(\mathbf{v}) && \text{(9a)}\\ \text{s.t.} \quad & v_0 = v^s && \text{(9b)}\\ & \boldsymbol{x}(v_J) \in \mathbb{X}_N(\theta^p) && \text{(9c)}\\ & e_j \in \Gamma(v_j, \theta^p) && \text{(9d)} \end{aligned}$$

$$v_{j+1} \in \left\{ \begin{array}{cc} \underset{\boldsymbol{x}(t), \boldsymbol{u}(t)}{\arg\min} & \int_0^T c(\boldsymbol{x}, \boldsymbol{u}, e_j) dt \\ \text{s.t.} & \dot{\boldsymbol{x}} = \boldsymbol{f}\left(\boldsymbol{x}, \boldsymbol{u}, e_j, \theta^p\right) \\ & \boldsymbol{x}_0 \in \mathbb{X}_0(v_j), \\ & \boldsymbol{x}(T) \in \mathbb{X}_f(e_j), \\ & \boldsymbol{g}\left(\boldsymbol{x}, \boldsymbol{u}\right) \in \mathbb{Z}(e_j), \\ & t \in [0, T] \end{array} \right\}. \quad \text{(9e)}$$

The outer level of the optimization is described with equations (9a) to (9d) and it is responsible to construct the *structure of the motion*. This is achieved by performing a discrete search using the GS method, shown in Fig. 5. The inner level of the optimization is described with the TO problem (9e) and its role is to compute the continuous trajectories, such that the discrete transitions can be realized. (9a) and (9b) are identical to (7a) and (7d) in formulation (7), however the discrete transition function f_γ described in (7b) is now replaced by (9e), which

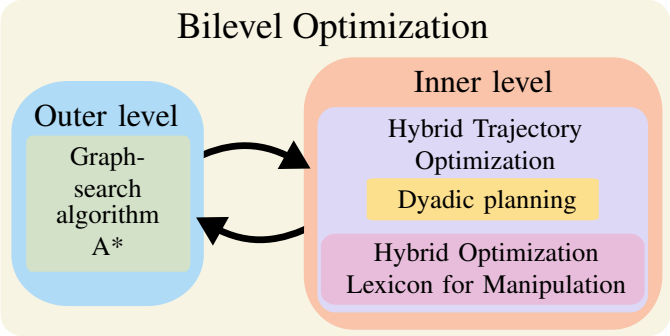


Fig. 5. Overview of the methods; optimized paths are obtained through an iterative execution of the outer (discrete) and inner (continuous) levels of the bilevel optimization.

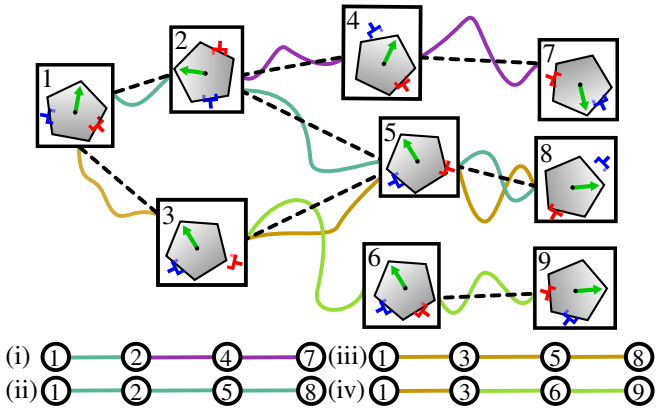


Fig. 6. A representative illustration of the of four different solution paths (i) to (iv) obtained with the proposed bilevel optimization method. The dashed lines depict the discrete transition found from the outer (discrete) level of the optimization, while the full lines are the continuous segments obtained from the inner (hybrid) level of the optimization. All four paths start from the same initial node with index 1. Solution path (i) ends at node 7. Solution paths (ii) and (iii) end at node 8 although they are different paths. In particular path (ii) will be generated when having a change of goal from final node 7 to node 8. Similarly, paths (iii) and (iv) end at different nodes which are identical with respect to the task, if we observe the state of the object only. An interesting point is the alternation of the transition from $e^{3,5}$ to $e^{3,6}$ by the inner (hybrid) level optimization, which results in a new path from node 6 to the goal.

denotes a nonlinear continuous optimization problem of the form (8). Additionally, to reflect in the computational formalism the dependency of the solution to the current mode of the partner's policy—as denoted by (4) and (5)—we modified (9c), (9d), and (9e) such that they depend on the parameters θ^p. The dyadic planning details are given in the Section VII.

***Outer-Inner level interplay**:* First, the outer level computes a discrete sequence of states, that define the initial *structure of the motion* towards the goal. Second, each segment of the motion is passed on and is optimized by the inner level. One or more of these segment may be altered by the inner level, resulting in a modification of the initial *structure of the motion*. Consequently the discrete sequence of states—subsequent to the modified segment—might become obsolete and might need to be re-computed by the outer level. This third step is closing the bilevel loop. Running these three steps iteratively, the bilevel optimization converges to the goal in a sliding window fashion. In Fig. 6 we illustrate the bilevel nature of the method with four examples.

A. Outer optimization level

For the outer optimization level we aim for a fast GS method, to compute the *structure of the motion*, *e.g.* a sequence of contact changes and object's motions. Given a discrete state representation, the state-space can be encoded into a graph, with each discrete state being a node v of the graph as described in Section IV-B. We use a coarse state representation that includes a discrete description of the object's state y and contact locations of agent's end-effectors c^k. A node v in the graph corresponds to the tuple (y, c^l, c^r), where $y, c^l, c^r \in \mathbb{N}$ and v is defined as an index to the tuple with $v \in \mathbb{N}$. Fig. 7 depicts a viable 2D state discretization.

A key element of the GS algorithms is the *successor* operator Γ, defined in Section IV-B. Γ allows us to attain a low branching factor and perform graph expansion more efficiently than brute-force node insertion [62]. We realize Γ for multi-contact manipulation and DcM scenarios specifically. We construct a simplified and intuitive physics model of the object-hand interactions based on the following rules [53].

Feasible states:

1) Left end-effector must always be on the left of the right.
2) A minimum distance between end-effectors is defined.
3) Applied forces have to be permissible given the contact location (see Section VI-3).
4) When both end-effectors are in contact, they must quasi-statically counteract gravity effects on the object's CoM.
5) The pivoting torque spawned in scenarios with single contact must not violate a given threshold (DcM-specific).

Feasible transitions (task-depend):

a) Both end-effectors must be in contact to rotate the object.
b) A single or both end-effectors can change contact within one transition.

Rules 1), 2) and a) are realized based on a mapping from the discrete state to the continuous Cartesian space of end-effectors. Rules 3) and 4) are computed based on quasi-static principles which are configuration dependent, typically used in grasping literature [30]. Further, rule 5) reserves as an implicit threshold on the required torque the partner has to apply, counteracting the pivoting torque applied by the agent as states with high torques are not allowed.

Regarding the particular choice of graph search method we use a heuristic A* algorithm for the following two reasons. First, the A* algorithm is considered a special case of Dynamic Programming (DP) [63], [64]. Thus, the solution of our overall problem is obtained by a bilevel optimization process. Second, A* is known for its computational efficiency, as it exploits heuristics to achieve a very low effective branching factor.

A* constructs an optimal sequence of states in terms of the evaluation function $\mathcal{C}(\cdot) = g(\cdot) + h(\cdot)$. The cost term $g(\cdot) \in \mathbb{R}$ is obtained from edges' costs (see Section IV-B), while the heuristic term $h(\cdot) \in \mathbb{R}$ needs to be admissible (always under-estimate the actual cost) and monotonic to ensure that the solution path found is optimal [58]. To facilitate optimal composition of the solution paths shown in Fig. 6, the heuristic term $h(\cdot)$ needs to be designed in accordance with the cost

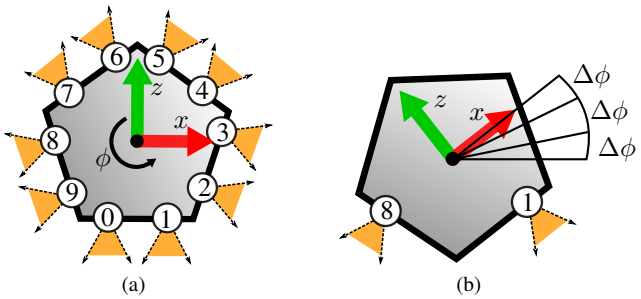


Fig. 7. 2D illustration of an example state space representation of the outer (discrete) level. The numbers in circles denote contact points. (a) Discretization of the contact space. (b) Discretization of the object's orientation (the translational part can be discretized with a checkerboard-like grid).

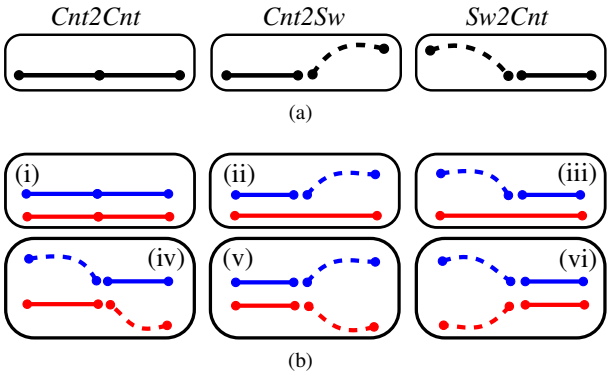


Fig. 8. The set of primitives referred as Hybrid Optimization Lexicon for Manipulation (HOLM). Dashed lines denote swing phase, while full lines denote contact phase. (a) Three primitives for a single end-effector. (b) Six bimanual primitives, where the left end-effector is coloured blue and the right is red. For primitives (ii) and (iii) end-effectors can be switched, such that the left (blue) remains in contact and the red performs a swing. Similarly, (iv) can be symmetrically switched.

function $c(\cdot)$ of the inner optimization level in (9e). The details on the heuristic and the cost terms are given in Section VIII.

The outer level provides the optimal *structure of the motion* efficiently—plus—an initial guess for the inner optimization level, by converting the optimal sequence of states to continuous trajectories using fifth-order-polynomials [53].

B. Inner optimization level

The inner level (9e) is responsible for optimizing the hybrid path (see Fig. 4), given the *structure of the motion*. Here all motion-relevant quantities (see Section IV-A) are optimized within their continuous manifold, while the discretized description of the quantities (see Section V-A) serves as a bases for the initial seed of the continuous problem.

For example, contacts $\mathbf{c}_i^k$ are optimally selected from the entire object surface, not only from the discrete contact locations shown in Fig. 7. As it has been reported in our previous work [7] and by others [39]–[42], [57], the computational times of optimizing the full path at once are extensively large for any type of online motion adaptation. Further, the non-convex nature of the problem gives no global optimality guarantees. Thus, to address the computational efficiency challenge, we propose to optimize each segment of the motion separately. To realize this, we introduce a decomposition of general hybrid motion into a set of hybrid motion primitives, referred as the Hybrid Optimization Lexicon for Manipulation (HOLM). Fig. 8a shows the primitives for a single end-effector and Fig. 8b illustrates a few combinations of HOLM primitives for bimanual agents.

In contrast to [65], where the hybrid motion is chopped into spacetime windows with fixed contact configuration and time duration, we choose to build each primitive as a sequence of two *contact-invariant phases* (see in Section IV-A) of variable time duration. The primitive *Cnt2Cnt* has to two consecutive contact phases, the *Cnt2Sw* has a contact phase followed by a swing phase—where the grasp-hold change starts—and the *Sw2Cnt* has a swing phase followed by a contact phase, where the grasp-hold change is completed. A single swing primitive does not contribute to the task, thus every swing phase is accompanied by a contact phase. Hence, each segment of the motion is optimized including the critical transitions of making and braking contact (*discontinuities*) to anticipate the next phase's contact configuration. The transition from one HOLM primitive to the next does not require special treatment as the contact configuration is not altered.

Regarding the collection of primitives used, *Sw2Cnt* and *Cnt2Sw* form the the minimal set of making-braking contact, while *Cnt2Cnt* is used to maintain contact, *e.g.* this is particularly useful when the robot rotates the object towards the goal. The use of the *Cnt2Cnt* primitive is encouraged with rule (a) described in Section V-A. In general, this set of primitives allows to fine-tune the hybrid robot motions to be legible [66].

The inner level accomplishes very fast optimal hybrid motion plan generation, given the *structure of the motion*. To the extend of our knowledge, this in turn empowers the bilevel optimization to be the first on-the-fly re-planning capable hybrid optimization method. This allow us to demonstrate online hybrid policy adaptation with respect to non-stationary dyadic interactions. Next, we provide the inner level details.

VI. Hybrid Planning via Trajectory Optimization

To solve the continuous optimization problem in (9e), we perform direct transcription as explained in Appendix A. This involves discretizing the trajectories of the following decision variables. For each i_{th} knot, the quantities of interest (see (2) and (3)) are, (i) the pose of the object $\mathbf{y}_i$, (ii) the velocity of the object $\dot{\mathbf{y}}_i \in \mathbb{R}^\nu$, (iii) action timings $\mathbf{\Delta T}_i$, (iv) the contact locations $\mathbf{c}_i^k$, and (v) the contact forces $\mathbf{f}_i^k$. We group these quantities in two vectors

$$\mathbf{x}_i = \begin{bmatrix}\mathbf{y}_i & \dot{\mathbf{y}}_i\end{bmatrix}^T \text{ and } \mathbf{u}_i = \begin{bmatrix}\mathbf{f}_i^k & \mathbf{c}_i^k & \dot{\mathbf{c}}_i^k & \mathbf{\Delta T}_i\end{bmatrix}^T. \quad (10)$$

$\forall i \in \mathbb{N}$ the trajectory of $\mathbf{x}_i$ and $\mathbf{u}_i$ describes a hybrid motion (described in Section IV-A). TO problems with intermittent contacts can be expressed using complementarity constraints, yet in practice convergence of these problems is difficult (see Appendix B). In our work, the *structure of the motion* is optimized by the outer level (Section V-A), which allows us to customize our transcription, and separate the motion in phase with different constraints—the *contact-invariant phases* mentioned in Section IV-A. Next, we present the phase-independent and the phase-specific constraints, respectively.

*1) **Phase-independent constraints**:* We introduce here constraints that are applied to all the knots of the trajectory regardless of the particular phase. We note that the object's dynamics $f_o(\cdot) \in \mathbb{R}^{2\nu}$ and the end-effectors' motion $f_e(\cdot) \in \mathbb{R}^{\nu}$ are integrated using trapezoidal quadrature. Also, $\psi_c \in \mathbb{R}^{2\nu}$ defines the reachable area of the agent's end-effectors, referred as arms workspace.

- Dynamics of the object (discussed further in Section VII)

$$\begin{bmatrix}\mathbf{y}_{i+1} & \dot{\mathbf{y}}_{i+1}\end{bmatrix}^T = \boldsymbol{f}_o(\mathbf{y}_i, \dot{\mathbf{y}}_i, \boldsymbol{\lambda}_i, \mathbf{f}_i^k, \mathbf{c}_i^k, \boldsymbol{\Delta}\mathbf{T}_i). \quad (11)$$

- Initial state of the object

$$\mathbf{y}_0 = \mathbf{y}_0^* \text{ and } \dot{\mathbf{y}}_0 = \dot{\mathbf{y}}_0^*. \quad (12)$$

- Desired final state of the object

$$\mathbf{y}_N = \mathbf{y}_N^* \text{ and } \dot{\mathbf{y}}_N = \dot{\mathbf{y}}_N^*. \quad (13)$$

- Kinematic limits of the agent's end-effectors

$$\mathbf{c}_i^k \in \psi_c. \quad (14)$$

We use simple box bounds to approximate them.

- Upper bound on the total time of the motion

$$\sum_{i=1}^{N} \boldsymbol{\Delta}\mathbf{T}_i \leq T. \quad (15)$$

*2) **Phase-specific constraints**:* The transcription of our hybrid optimization problem follows the phase-based parameterization (see Fig. 4) introduced in [40], also used in [39]. We extend this by considering the three possible collision states between two rigid bodies as described in [67], and we split the knots in three sets according to the *contact-invariant* phases; the contact, swing, and pre-contact sets, shown in Fig. 4. At each discretization point (knot) a constant subset of constraints needs to be satisfied. Most of the phase-specific constraints are time independent, which allows us to optimize each phase's duration and satisfy the constraints of each phase simultaneously. Each phase is characterized by a distinct set of decision variables that allows us to enforce a number of constraints implicitly and reduce the number of decision variables. A list of the constraints categorized according to the phase of the motion follows.

i) Contact phase:

- Permissible contact forces (discussed in detail next):

$$\psi(\mathbf{f}_i^k) \geq 0. \quad (16)$$

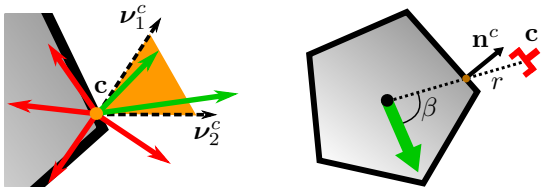
- No contact point slipping (*implicit constraint*):

$$\dot{\mathbf{c}}_i^k = 0. \quad (17)$$

- End-effectors in contact with the object (*implicit constraint*):

$$d(\mathbf{c}_i^k, \mathcal{S}_{obj}(\mathbf{y}_i, \mathbf{c}_i^k)) = 0, \quad (18)$$

where $d(\cdot) \in \mathbb{R}$ is the signed distance between end-effector and object. $\mathcal{S}_{obj} : (\mathbf{y}, \mathbf{c}^k) \rightarrow \mathbb{R}^{\nu}$ computes the closest point on the object's surface to the end-effector's location and stresses out the importance of object's shape representation described below.



(a) (b)

Fig. 9. (a) Illustration of an end-effector in contact with the object. Valid 2D contact forces (shown in green) are generated by the conical combination of the rays $\boldsymbol{\nu}_1^c$ and $\boldsymbol{\nu}_2^c$. This form can be preferred for interior-point methods that traverse the interior of the feasible region and avoid unnecessary considerations of invalid contact forces (shown in red). (b) The position of the end-effector described in 2D polar coordinates $c = [\beta, r]^T$, along with normal vector $\mathbf{n}^c$ at the imminent contact location are depicted.

ii) Swing phase:

- End-effector's motion:

$$\mathbf{c}_{i+1}^k = f_e(\mathbf{c}_i^k, \dot{\mathbf{c}}_i^k, \boldsymbol{\Delta}\mathbf{T}_i). \quad (19)$$

- End-effector's swing motion away from object:

$$d(\mathbf{c}_i^k, \mathcal{S}_{obj}(\mathbf{y}_i, \mathbf{c}_i^k)) > 0. \quad (20)$$

- No force (*implicit constraint*):

$$\mathbf{f}_i^k = 0. \quad (21)$$

iii) Pre-contact phase:

- End-effectors touching the object:

$$d(\mathbf{c}_i^k, \mathcal{S}_{obj}(\mathbf{y}_i, \mathbf{c}_i^k)) = 0. \quad (22)$$

- No force (*implicit constraint*):

$$\mathbf{f}_i^k = 0. \quad (23)$$

*3) **Permissible contact forces**:* With (16) we denote the allowable contact forces exerted by the end-effectors to the object. This forces should satisfy the constraints[3]

$$\mathbf{f}^T\mathbf{n}^c \geq 0 \quad (24a)$$

$$|\mathbf{f}^T\mathbf{t}^c| \leq \mu\mathbf{f}^T\mathbf{n}^c, \quad (24b)$$

where $\mathbf{f}$ is the force vector, $\mathbf{n}^c \in \mathbb{R}^{\nu}$ is the normal and $\mathbf{t}^c \in \mathbb{R}^{\nu}$ is the tangent vector at the contact point on the object's surface. $\mu \in \mathbb{R}$ is the friction coefficient. Here, (24a) is the unilateral contact constraint and (24b) is the friction cone constraint; we use the linearized friction cone form. In (24) the constraints are denoted using the halfspace representation. Alternatively, the force constraints can be enforced using the vertex representation, also used in [68]

$$\mathbf{f} = \sum_{\ell=1}^{\kappa} \alpha_\ell \boldsymbol{\nu}_\ell^c, \quad (25)$$

where $\boldsymbol{\nu}_\ell^c = \mathbf{n}_\ell^c + \mu\mathbf{t}_\ell^c$ are the extreme rays of the friction cone, $\alpha_\ell \geq 0$ are weighting coefficients and $\kappa \in \mathbb{R}$ is the number of rays used. Normals, tangents, and extreme rays are functions of the contact location and are obtained from $\mathcal{S}_{obj}(\cdot)$ according to the object shape representation. A 2D graphical illustration and intuitive comparison between the halfspace and the vertex forms is given in Fig. 9a. We choose to enforce constraint (16) with (25) as we have empirically noticed faster convergence.

[3]For readability, we drop the indices with respect to end-effectors and knots of the trajectory.

*4) **End-effectors' position representation**:* Equations (14), (18), (20), (22), and (25) are realized given a specific representation of the end-effectors position. We choose to represent the end-effectors position relative to the CoM of the object in polar coordinates, graphically shown for the 2D case in Fig. 9b.

*5) **Object's shape representation**:* The object's surface is represented with a closed cubic spline curve. The spline representation is a smooth description of the object's surface from which all relevant properties along with their gradients can be extracted, like normal and tangent vectors. The use of continuous representations of the object's surface is more generic than approaches like [69] that rely on the convexification of the object's shape, as scaling with respect to the number of edges/phases becomes cumbersome.

*6) **Input variables and hyper-parameters**:* The only required input variable is the description of the manipulation task, $\theta^{\mathcal{M}}$. Here we use the start and goal state of the object, denoted as $\begin{bmatrix}\mathbf{y}_0^* & \dot{\mathbf{y}}_0^*\end{bmatrix}$ and $\begin{bmatrix}\mathbf{y}_N^* & \dot{\mathbf{y}}_N^*\end{bmatrix}$, respectively. Nonetheless, the specification of $\theta^{\mathcal{M}}$ can be as flexible as needed, and can be specified either via the cost function $c(\cdot)$, *e.g.* minimize object's acceleration, or via the constraints $\boldsymbol{g}(\cdot)$, *e.g.* set upper object's velocity limits or set forbidden regions of the workspace. To solve the TO problem two hyper-parameters need to be specified. (i) The resolution of the grid N (number of knots) shown in Fig. 4. (ii) Motion's upper time bound T.

VII. Dyadic constraint and Partner model

In this section, we present the specifics on how to incorporate the partner's policy in the framework and generate dyadic hybrid plans. In DcM scenarios the object is jointly manipulated by both individuals—as specified by (3)—by applying forces on it. We propose to incorporate the partner's policy in the TO framework through the *transcription* constraints defined in (11). Only now, the object's dynamics are subject to the partner's wrenches too, described by

$$\begin{bmatrix} mI & 0 \\ 0 & J \end{bmatrix} \ddot{\mathbf{y}}_i + \begin{bmatrix} mg \\ \dot{\mathbf{y}}_i^{\omega} \times (J\dot{\mathbf{y}}_i^{\omega}) \end{bmatrix} = \sum_k^K \begin{bmatrix} I \\ \hat{\mathbf{c}}_i^k \end{bmatrix} \mathbf{f}_i^k + \boldsymbol{\lambda}, \quad (26)$$

where $m \in \mathbb{R}$ and $J \in \mathbb{R}_{\geq 0}^{\nu \times \nu}$ are the mass and inertia of the object, I is the identity matrix, g is the gravitational acceleration, $\dot{\mathbf{y}}_i^{\omega}$ is the object's angular velocity, and with $\hat{(\cdot)}$ we refer to the cross product matrix formed by the input vector. By realizing (11) according to the augmented dynamics—where $\boldsymbol{\lambda}$ represents the partner's contribution—the trajectory optimization generates plans in accordance to the partner's policy, referred as *partner-aware*. This is illustrated in Fig. 3 with the physical constraints block. In contrast to [55], where the method assumes full control authority over the partner's actions, the only requirement of our method is an estimate of the partner's policy.

Partner's policy parametric model (Fig. 3): This work aims to provide a principled way towards incorporating partner's actions into the policy of the agent. An essential step towards this goal is to identify the appropriate function space in which the partner's policy lies. We use here a simple but ample model for the partner's policy,

$$\boldsymbol{\lambda} = \mathbf{K}^p(\mathbf{y}_N^* - \mathbf{y}_i) + \mathbf{D}^p(\dot{\mathbf{y}}_N^* - \dot{\mathbf{y}}_i). \quad (27)$$

The parameters $\mathbf{K}^p$ and $\mathbf{D}^p$ denote a spring-damper behavior of the partner towards the goal $\begin{bmatrix}\mathbf{y}_N^* & \dot{\mathbf{y}}_N^*\end{bmatrix}$ of the co-manipulation task. $\mathbf{K}^p$ can be interpreted as the parameter that can shape whether the partner acts as a leader $\mathbf{K}^p \gg 0$ or as a follower $\mathbf{K}^p = 0$, along with all the intermediate behaviors. The goal $\begin{bmatrix}\mathbf{y}_N^* & \dot{\mathbf{y}}_N^*\end{bmatrix}$ captures the partner's intentions relative to the task. This model has been used in human motor control research [70], as it captures the essence of the partner's policy.

Partner's policy oracle function: As this work is not focused on estimating the partner's policy $\hat{\pi}^p$, we assume an oracle function exists. The oracle function can predict the parameters $\theta^p = (\mathbf{y}_N^*, \dot{\mathbf{y}}_N^*, \mathbf{K}^p, \mathbf{D}^p)$ that describe the current mode of the partner's policy (see Section III). This in turn enables the use of (5) without the need to compute $Pr(\theta^p|\mathbf{x}_t, \mathbf{q}_t^p, H^p)$. The implementation of the oracle function could be realized with the methods reviewed in Section II-A1.

VIII. Experiments

In this section, we first provide computational evaluations of the proposed method. We proceed with simulations on both a single agent and a dyadic setup. Last, we evaluate the proposed method with real-world DcM experiments. See the attached material for video footage of the simulations and the human-robot experiments during DcM tasks.

The purpose of the computational study is to highlight the computational gains of HOLM, in comparison to our previous work [7], and emphasize the importance of specific algorithmic choices. The objective of the single agent simulations is to demonstrate the capabilities of the method to plan highly dynamic motions. Likewise, the dyadic simulations present a multitude of situations where the resulting hybrid policy of the agent is conformed to the partner's policy. The experiments with a human-robot dyad demonstrate our method's viability to plan on-the-fly under real-world conditions.

Parameters of experimental setup: The state of the object is $\mathbf{y} = \begin{bmatrix}x & z & \phi\end{bmatrix}$ with task dimension $\nu = 2$, which is sufficient for the demonstrations; however, both levels (Section V) can be realised in 3D space with $\nu = 3$, *e.g.* the inner level can be modified based on our previous work [68]. To obtain the discrete state (y, c^l, c^r) mentioned in the outer level (Section V-A), we only need to consider ϕ and the contact locations, as the translational components of $\mathbf{y}$, do not affect the *structure of the motion*. ϕ is discretized with $30°$ resolution and for each of c^l and c^r we specify 16 contact locations. With these choices and with the rules defined in Section V-A the branching factor for a brute-force search method is $b \approx 23$. Yet, the A* algorithm uses heuristics to guide the search, thus the average effective branching factor for our setup is $b^* \approx 4$, which is the key to very efficient outer level computation times. Regarding the evaluation function C of A*, the heuristic term h models the angular difference between the current and goal rotation angles of the object, while the transition cost function g corresponds to the required movement length; shorter transitions in the continuous Cartesian space are cheaper. The cost function of the HOLM primitives similarly minimizes distance to goal and overall path length. With this setup—as discussed in Section V-A—the resulting A* discrete solution sequence is optimal and

TABLE I
Average computation time of majorly used HOLM primitive types described in Fig. 8.

HOLM type	No. of variables	Exact Hessian	Limited-memory BFGS
Cnt2Cnt	134	34ms, 28 iter	75ms, 34 iter
Cnt2Sw	128	47ms, 38 iter	>6000ms, >3000 iter
Sw2Cnt	129	50ms, 29 iter	2800ms, 1535 iter
Cnt & Cnt	194	44ms, 25 iter	58ms, 26 iter
Cnt & Sw2Cnt	189	131ms, 70 iter	>8000ms, 3000 iter
Cnt & Cnt2Sw	188	49ms, 29 iter	2740ms, 1521 iter

TABLE II
Computational evaluation of the bilevel optimization and inner level specifically with respect to five different groups of tasks.

Task	No. of contact changes	No. of segments	Graph size	First segment time	Time per segment	Full path time (HOLM)	Full path time
i	0	1	1011	0.24s	0.24s	0.02s	0.02s
ii	1	3-4	11567	1.08s	0.45s	0.09s	0.30s
iii	1-2	4-6	18510	1.52s	0.53s	0.13s	0.75s
iv	2-3	6-8	47097	4.86s	1.07s	0.18s	1,30s
v	4-6	8-12	102329	8.23s	2.93s	0.25s	5,20s

in accordance with the inner optimization level. The knot resolution used for the HOLM primitives is 6 knots per phase, and the friction cone is $\mu = 0.5$. For each HOLM primitive, we use an upper time bound of $T = 3.5sec$ for contact phases and $T = 6.5sec$ for swing phases. Once the hybrid motions are optimized in the task space they are being mapped to the configuration space of the robot using inverse kinematics [71].

Regarding the implementation details, we use `CasADi` [72] to realize the HOLM primitives[4], where each primitive is a separate parameterizable hybrid problem. Each hybrid problem is a large and sparse nonlinear optimization problem which is solved using `IPOPT` [73], while the automatic differentiation capabilities of `CasADi` allows us to provide exact gradient and hessian information. The A* planner and the lower-level control aspects of the robot, *e.g.* IK, are implemented in the `Robot Control Software (Rcs)` framework[5]. All experiments are conducted on a 64-bit Intel Quad-Core i7 3.40GHz workstation with 16GB RAM.

A. Computational Evaluations

We now show improved computational results over our previous single optimization based hybrid planning approach [7].

1) ***HOLM computation times****:* In Table I we present the average computation times for 15 runs of each HOLM primitive. Each primitive is evaluated on a variety of tasks, using three objects with different shape, a sphere, a rectangular box, and a parallelogram box. The tasks involve translation from $0m - 1m$ and rotation from $0°$ - $180°$, similar to the ones shown in Figs. 11 and 12. The computational times reported are obtained with zero initial seed and they scale linearly with respect to the number of knots and the time horizon. *These results reveal the computational benefits of HOLM.*

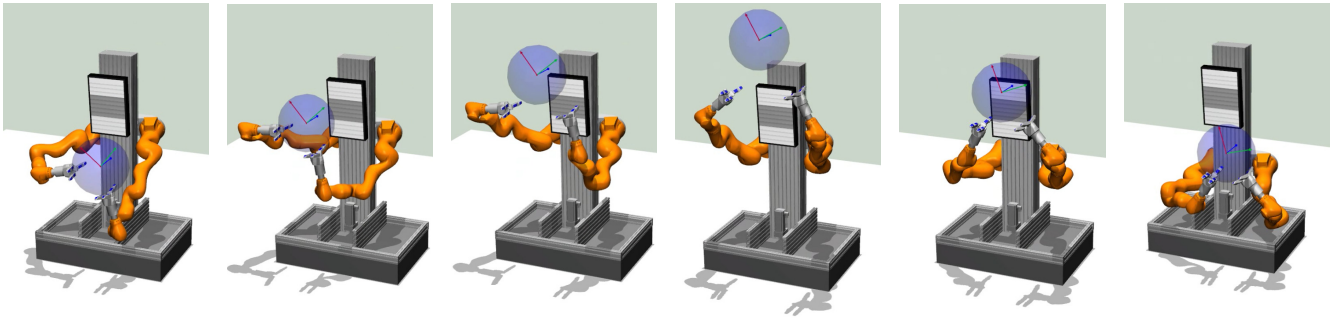
2) ***Bilevel optimization computation times****:* In Table II we present the average computation times for the bilevel optimization. We group tasks in terms of angular distance from the initial state of the object to the goal, as this grouping nicely relates to the number of contact changes required to complete the task. As the number of contact changes depends on the initial contact configuration, a range of contact changes is given rather than an exact number (second column of Table II). We also provide the approximate horizon of the resulting motion. These tasks are:

i) $0° < \Delta\phi < 20°$, with motion horizon ∼7 sec,
ii) $20° < \Delta\phi < 120°$, with motion horizon ∼20 sec,
iii) $120° < \Delta\phi < 140°$, with motion horizon ∼28 sec,
iv) $140° < \Delta\phi < 200°$, with motion horizon ∼71 sec,
v) $200° < \Delta\phi < 360°$, with motion horizon ∼114 sec.

We show the computation time required for the first segment of the motion, the average computation time for each one of the consecutive segments (fifth and sixth column of Table II). The former indicates the planning time until the receding horizon plan can be updated, while the latter specifies how fast the successive segments are computed. These computation times comprise revising *structure of the motion* too. These times are proportional to the graph size displayed with the number of explored nodes (fourth column of Table II). *These evaluations exhibit the online planning capabilities of the bilevel method.*

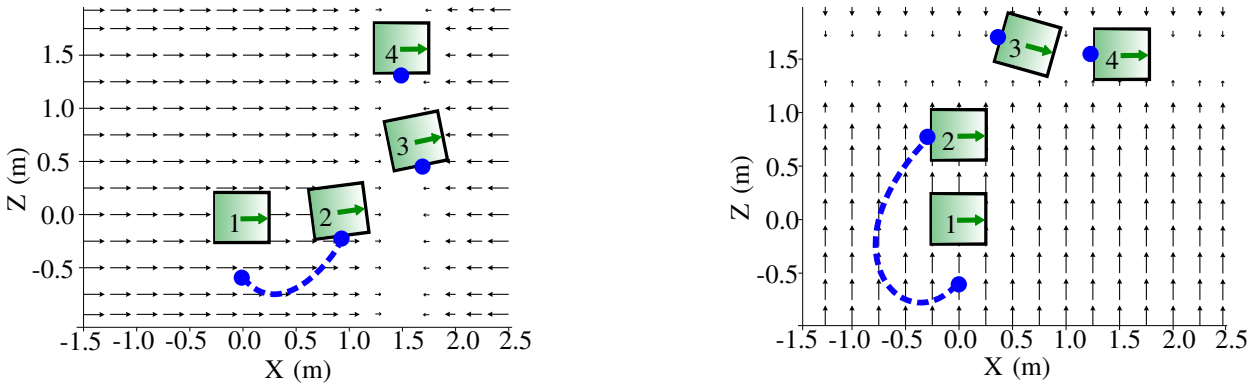
[4]An open-source repository with our HOLM implementation can be found in: https://github.com/stoutheo/HybridManip/tree/HOLM-primitives

[5]Information about `Rcs` can be found in: https://github.com/HRI-EU/Rcs



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

Fig. 10. In these keyframes a single robot agent performs a dynamic hybrid manipulation task, *i.e.* rotate a ball by throwing it up in the air and catching it.



(a) Partner's policy 1: $\pi^p 1$ (b) Partner's policy 2: $\pi^p 2$

Fig. 11. In (a) and (b) we illustrate generated motion plans in response to two different partner policies. The green rectangle is the manipulated object, the blue dot is the left end-effector of the agent. The object's start pose is annotated with 1 and the goal with 4. The arrow field illustrates the forces applied by the partner. (a) and (b) stress out the dependency of chosen contact location to the partner's policy.

*3) **Discussion**:* The main steps that allow us to improve the computation times from tens of seconds in our previous work [7], to milliseconds for HOLM and few seconds for the bilevel optimization are: (i) Decomposing the problem into HOLM primitives, which allows to keep the size of the hybrid problems small (second column of Table I), (ii) exploring the hybrid structure of the problem with an efficient graph-search algorithm, (iii) formulating a sparse problem that can be efficiently solved[6], (iv) providing the exact Hessian using automatic differentiation, (third and fourth column of Table I), and (v) selecting the end-effectors' and permissible force representation discussed in Sections VI-3 and VI-4.

The seventh column of Table II shows the average computation times required to optimize the full continuous path using the HOLM primitives only for the inner optimisation level. First, as the HOLM primitives utilize the initial seed provided by the outer level (see Section V-A), the computation times are much smaller than the ones in Table I. Second, in the eighth column of Table II we provide the computation times (only inner level) needed to compute the full path using a hierarchical approach, as in [74]. The comparison between the seventh and eights column of Table II reveals the computational gain of using HOLM primitives with respect to the baseline approach.

Finally, the success rate of the bilevel optimization depends on the selected discretization of the outer level. If a fine discretization is selected, an optimal solution is always found. However, this is achieved at the expense of computational efficiency. Therefore, we used a discretization of 30° that provides fast solutions and satisfying success rate. The inner optimization level has been empirically observed to provide robust solutions in terms of convergence, due to the appropriate initial seed given. This allows us to mitigate sensitivity issues with respect to the initial seed, which is a common drawback of continuous optimization methods. Further, even in case the inner level fails to converge, we can always use the interpolated trajectory obtained by the outer level.

[6]Interior-point methods are able to solve our specific problem more robustly than sequential quadratic programming methods.

The computation times presented demonstrate the online planning capabilities of our method. We gained approximately a $\times 10$ to $\times 50$ speedup in comparison to our previous work [7], while simultaneously the arm's contact sequence pattern (structure of the motion) is automatically computed.

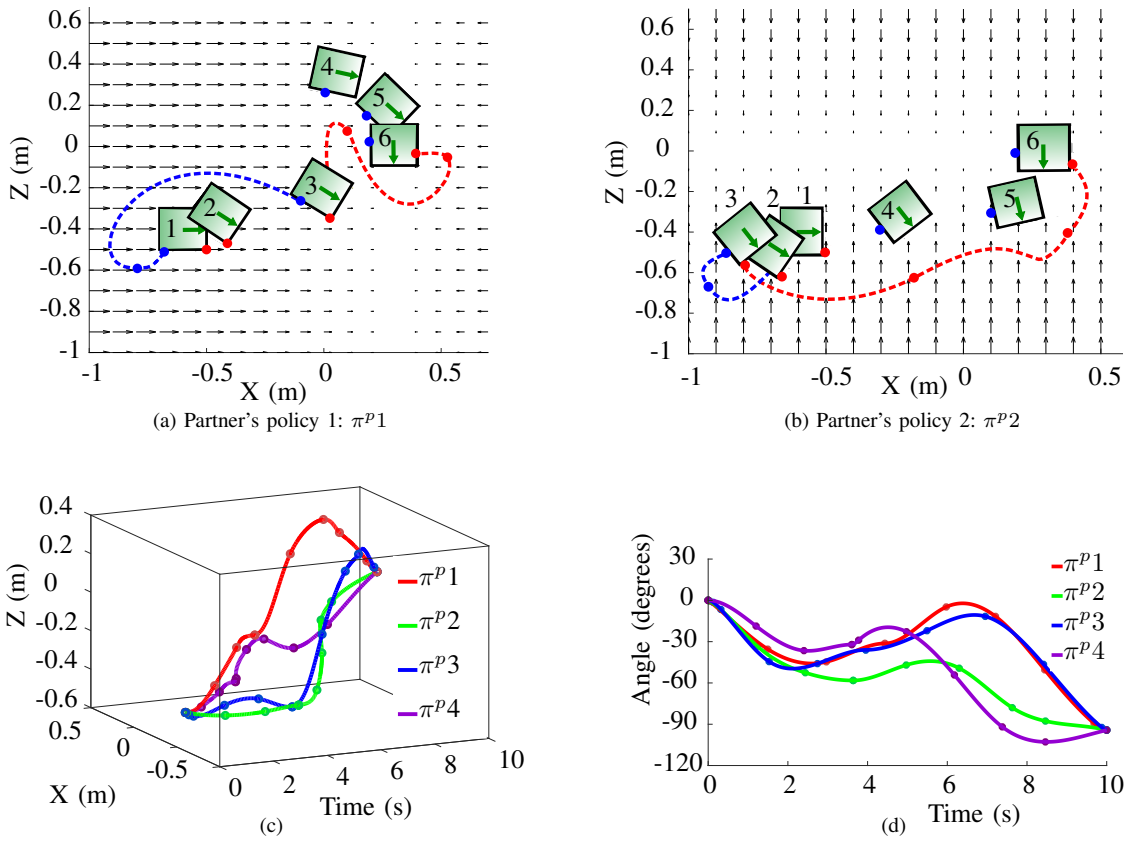
B. Simulations experiments

We present here a number of different motion plans generated by the proposed method that demonstrate the capability to find dynamic and partner-aware solutions. A dynamic motion is illustrated in Fig. 10(a-f), where a robot performs in simulation the challenging task of throwing and catching a ball. Next, the variability of the solutions generated with respect to the dyadic setup is analyzed.

*1) **Partner-aware solutions**:* First, we alter the partner's policy parameters $\mathbf{K}^p$ and $\mathbf{D}^p$ in (27). Each partner's policy is expressed as a force field along one axis: in $\pi^p 1$ along X, in $\pi^p 2$ along Z, in $\pi^p 3$ along the main diagonal, and in $\pi^p 4$ along ϕ axis. In Fig. 11a and Fig. 11b, the agent has one end-effector and jointly completes with the partner, a $2.12m$ translation task in a zero gravity (table-top) scenario. Fig. 12a and Fig. 12b illustrate solutions for a $0.98m$ translation and a -90° rotation task, generated as responses to two different partner policies in an scenario with gravity along the z axis. The former task highlights the effect of the partner's policy on the selected contact location. The variation of the computed solutions is evident in the latter task in Figs. 12c, 12d and 13, where we present trajectories for four distinct partner's policies.

Second, we adjust the partner's goal $\begin{bmatrix}\mathbf{y}_N^* & \dot{\mathbf{y}}_N^*\end{bmatrix}$ in (27). Fig. 14 shows the optimized contact locations and swing motions for three goals. *These experiments demonstrate the capability of our method to adapt trajectories, contact locations, and action timings in response to different partner policies.*

*2) **Outer vs. inner level solutions**:* With Fig. 16 we show the benefits of our method over solely search-based planning approaches [53]. During this 90° object rotation DcM task, the human partner does not properly support the object, as shown in Fig. 15, where the avatar's left hand is not in contact with the object. In our partner model, this is represented through parameters $\mathbf{K}^{p,\phi} = 0$ and $\mathbf{D}^{p,\phi} = 0$ in (27). The search-based



(a) Partner's policy 1: π^p1 (b) Partner's policy 2: π^p2

(c) (d)

Fig. 12. Similar to Fig. 11a and Fig. 11b, (a) and (b) depict the resulting trajectories in response to two partner policies. The red and blue dots are the right and left end-effectors of the agent. The start pose is annotated with 1 and the goal with 6. Most of the object's trajectory is planned at the active regions of the partner's force field, indicating that the robot utilizes the partner's contribution to the task accordingly. Trajectories are displayed separately for four distinct partner's policies: (c) in x and z dimensions, (d) in ϕ dimension.

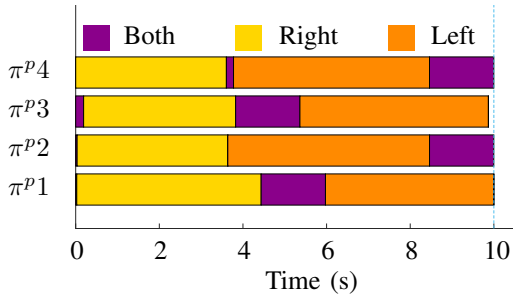


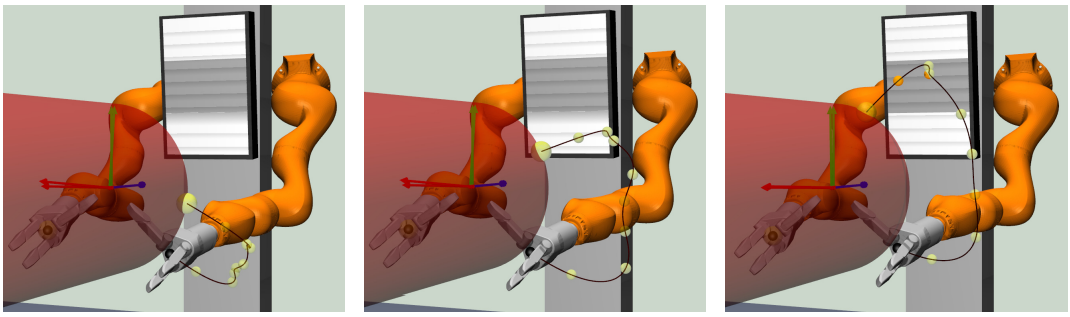
Fig. 13. The arms' contact sequence pattern for the four distinct partner's policies shown in Figs. 12c and 12d. The colours indicate which arm is in contact with the object.

outer level provides a coarse solution (blue line in Fig. 16) that does not take into account the policy of the partner, while the inner level significantly alters the plan (green line in Fig. 16) to conform to the dynamic constraints of the task, *i.e.* jointly balance the object. *This shows that the inner level significantly alters trajectories, durations, and action timings of the outer level solution, to respect dynamic aspects of the interaction.*

3) ***Online adaptation to alternations of the joint goal****:* During this DcM scenario, the initial object's target orientation of $150°$ changes to $-55°$, while the agent is not aware of this change in advance. The object's target serves as a proxy to the partner's intention. This is realized by altering the goal $\begin{bmatrix}\mathbf{y}_N^* & \dot{\mathbf{y}}_N^*\end{bmatrix}$ during the interaction shown in Fig. 18. In Figs. 17a to 17c we show the angular state evolution of the object and the two end-effectors. Once the change of joint goal occurs, replanning is completed in $0.95sec$ for the first segment of the receding horizon plan. The consecutive segments are adapted in $1.13sec$. *This illustrates that our method can adapt on-the-fly trajectories, action timings, durations, the structure of the motion and contact locations to respond to real-time changes of the joint task.*

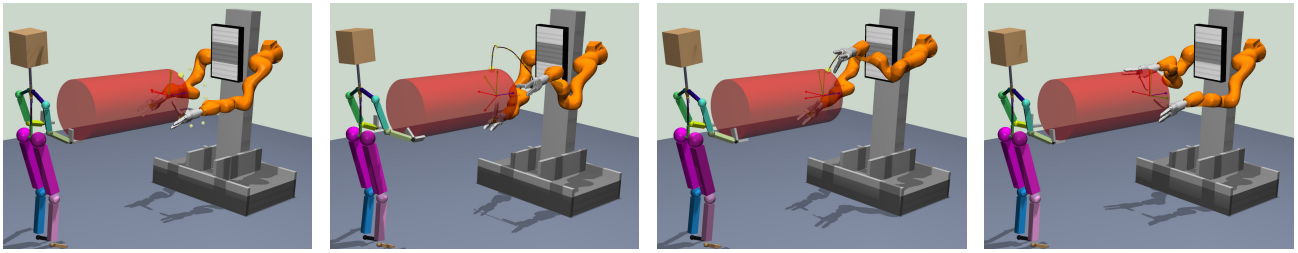
C. DcM experiments

We validate our approach in a real setting, where a human partner jointly manipulates two different objects with a bi-manual, *i.e.* $k \in \{1, 2\}$, and $n = 32$ DoF robot. The robot moves on the horizontal plane in a omni-directional fashion—due to its mobile base—and utilizes its two Kuka LBR iiwa 820 arms along with two Schunk dexterous 3-finger hands for manipulation and DcM tasks. A linear joint allows the arm base to be translated along the vertical axis. We use a box and a cylindrical object. Both are bulky, so that a human cannot perform the task alone. The hybrid motion plans are optimized in the task space and are realized on the robot in a open-loop fashion, after being mapped to the configuration



(a) (b) (c)

Fig. 14. The agent's left end-effector performs a swing motion, while the partner supports the object from the other side. Depending on the partner goal, the contact locations change. The small yellow spheres denote the knots of the trajectory and the larger one the anticipated contact location. The black curve is the interpolated trajectory. The partner's intended object orientations are (a) 30°, (b) 60° and (c) 90°.



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

Fig. 15. Keyframes of a rotational DcM task with $\mathbf{y}_N^* = 90°$ intended goal, where the partner is not properly supporting the object. (a) The left hand of the avatar is not in contact with the object. (b) The object is first rotated in the opposite direction to be properly supported by the agent's right hand. (c) The swing motion to change grasp hold is performed. (d) The object is properly held and jointly rotated to the intended target.

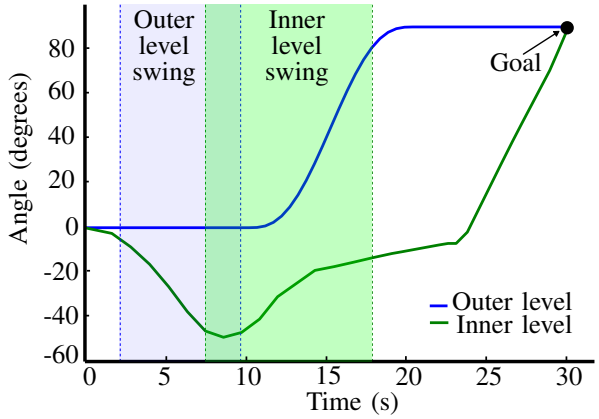


Fig. 16. Evolution of the object orientation for the 90° DcM task shown in Fig. 15. The blue curve is the path computed from the outer level of the optimization, while the green is the final path optimized by the inner level. The shaded areas indicate the duration and temporal placement of the swing motion of the left end-effector. The inner level initially rotates the object opposite to the goal to satisfy the dynamic constraints of the task.

space using IK. A detailed description of the physical system can be found in [53]. The robot utilizes surface contacts at the planned contact locations as a form of mechanical feedback. Further, it is worth noting that the joint-range of the robot only permits rotations of the object of about 90° before it reaches kinematic limits, thus grasp-hold changes are required.

1) ***Human-aware solutions****:* In correspondence to the simulation experiments shown in Figs. 12 and 13, we demonstrate a 90° box rotation task with one contact change per arm. During this real-world evaluation, the human partner is actively rotating the box and no change of goal occurs. The key-frames of the DcM scenario are depicted in Fig. 19.

2) ***Online adaptation to human's goals****:* We perform two experiments to demonstrate the on-the-fly adaptation to the humans' real-time changing goals. Similar to the simulations shown in Figs. 17 and 18 these changes are unexpected.

In the first experiment, shown in Fig. 20, the initial goal of the human partner is to rotate the cylindrical object to 90°. The full plan is computed by the bilevel optimization in $1.72sec$ and the duration of the resulting hybrid motion is $27.31sec$, with one contact change. During the experiment, the human partner decides that the preferred orientation of the object should be 180°. Given the change of the human's goal, the robot agent computes the first segment of the adapted plan in $0.54sec$ and the remaining segments of the hybrid plan are computed within $1.28sec$, while the total duration of the updated plan is $51.63sec$ with two contact changes.

In the second experiment, shown in Fig. 21, the initial goal of the human partner is to rotate the object to $-45°$. The hybrid motion plan includes a grasp-hold change of the right arm and has a total duration of $28.93sec$, which is computed within $1.69sec$. During execution, the human alters the intended dyadic goal and aims for a 90° desired object orientation. The first segment of the adapted motion (receding horizon) is computed in $2.60sec$, while the remaining hybrid plan is computed in parallel with the execution of the first segment in $4.98sec$. The total updated plan has a duration of $57.35sec$

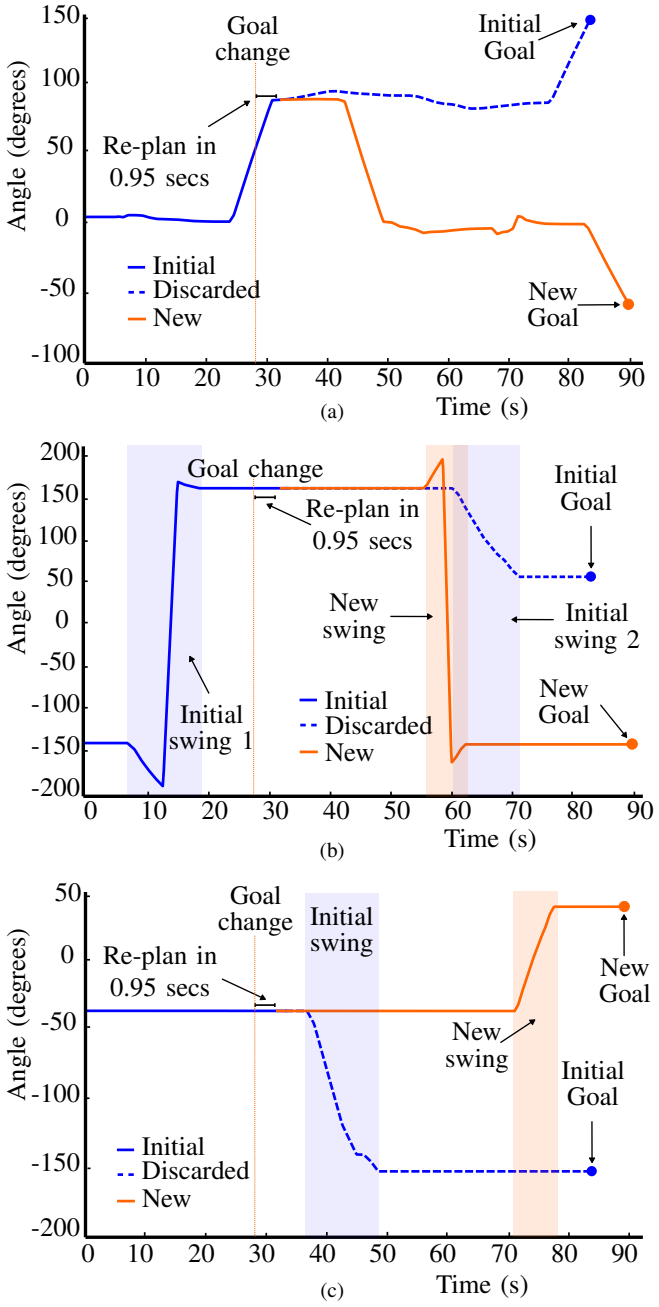


Fig. 17. Evolution of (a) the object's absolute orientation, and the relative to the object orientation of the (b) left end-effector and (c) right end-effector for a non-stationary task. The shaded areas indicate the duration and temporal placement of the end-effectors' swing motion and its adaptation according to the switch of the joint goal. The vertical orange dotted line indicates the exact point in time where the change happens. The re-planning duration of $0.95sec$ is shown with respect to the total motion duration of $88.27sec$.

and includes two contact changes. Note that this experiment requires a complete reversal of the object's orientation. The computed motion stops near $-45°$ (see attached video) and then an opposite rotation is initiated. The stop is due to the rotation reversal and not due to stretched computation time.

IX. Discussion

In this section, we discuss our dyadic modelling choice, practical considerations and possible extensions of the proposed approach.

In our partner-aware dyadic formulation presented in Section III we treat the two individuals and their dyadic interaction separately (see Fig. 3); such treatment was used to analyze Human–Human interactions [24]. We believe that this design choice is of core importance; as it has been shown in a variety of scenarios in Section VIII. This allows our method to generalize over different tasks and partner behaviors, assuming that the partner behavior can be approximated with the simple but rich spring-damper model.

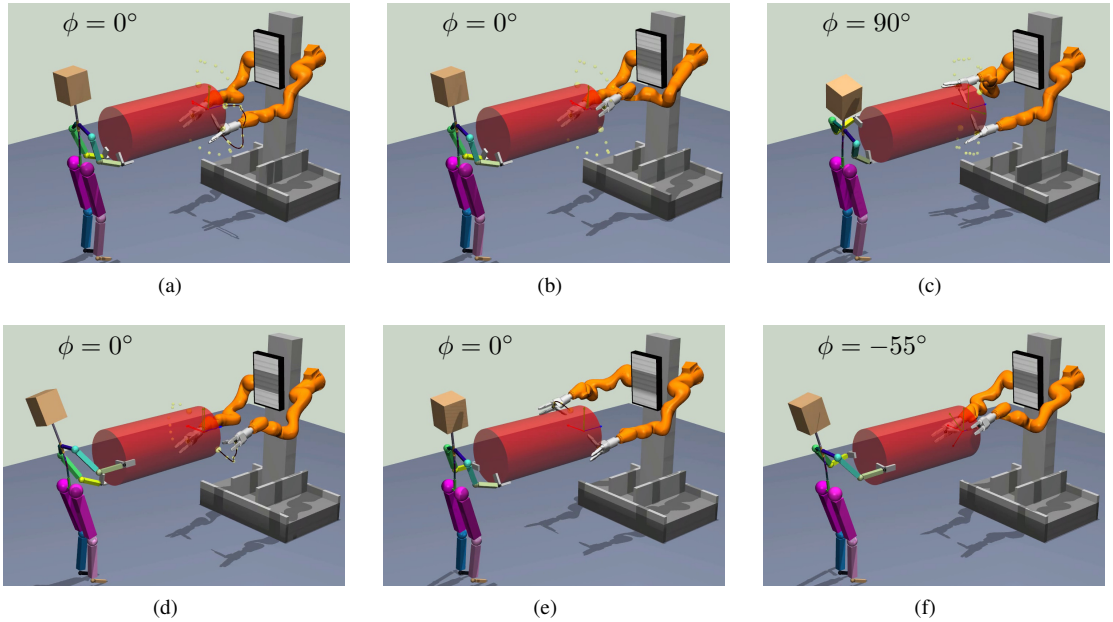
For the outer level described in Section V-A, we used a specific discrete state representation and rules that do not model the partner explicitly. Nevertheless, our framework can be easily extended to enable multi-layered dyadic interaction modelling. The inner level (see Section V-B) takes into account geometric [55] and dynamic aspects of the interaction (see Section VIII-B2), while the outer level could incorporate logical interaction rules, *e.g.* if one of the partner's arm is swinging, the agent's end-effectors should remain in contact with the object. Also due to the A* choice, the outer level finds only the optimal discrete solution. One could enhance the planning robustness at the expense of optimality or computation time, by realizing the outer level with Anytime Repairing A* [75], that computes multiple incrementally optimal solutions.

Last, during the robot experiments, we identified the potential usefulness of micro-scale adaptation to task's current state. Essentially, coping with arbitrary dyadic situations requires both our online planning adaptation method (long horizon) and closed-loop control (short horizon). To this front, a Hybrid Model Predictive Control (MPC) implementation based on the HOLM primitives would allow the robot to correct for small errors during the evolution of the task, *e.g.* close the loop with respect to the object's state. The HOLM computation times presented in Section VIII-A1 serve as a first promising step.

X. Conclusion

This article presents a novel concept towards online adaptive robot motion generation for physical human-robot collaboration tasks, such as Dyadic collaborative Manipulation (DcM) scenarios. We propose a formalization towards addressing the joint action problem based on the assumption that an estimate of the partner's non-stationary intentions can be attained.

Further, we propose a novel computational formalism to exploit the efficiency of informed-graph-search methods in combination with the dynamic and geometric reasoning of optimal control methods. Our approach computes the optimal hybrid policy for the robot to complete manipulation tasks as a member of a dyad or alone. The method only assumes a roughly estimated model of the partner's policy and a model of the object. With this information, our bilevel optimization computes dynamically consistent and optimal hybrid paths for the (i) trajectory of the object, (ii) agent's forces, (iii) agent's contact locations, (iv) respective timings of these actions, and (v) arms' contact sequence pattern. Due to the computational efficiency of the method the optimal paths can be computed online, such that on-the-fly adaptation to real-time changes of the dyadic interaction can be realized. This capability of the proposed method is particularly important for HRC



(a) (b) (c)

(d) (e) (f)

Fig. 18. A sequence of frames of a non-stationary DcM scenario. The orientation of the object is given at the top left corner of every keyframe. The initial joint goal is to rotate the object to $150°$; keyframes (a) and (b) show the hybrid plan and the early execution steps for achieving this joint partner-agent goal. However, in between (b) to (c) the joint intended goal changes to rotate the object to $-55°$. This causes an on-the-fly adaptation to a new hybrid motion plan in (c). Keyframes during the execution of the adapted plan are shown in (d), (e) and (f).

scenarios, where typically the human partner alters intentions and behaviors multiple times throughout the interaction.

In summary, the proposed method is able to optimize over a variety of different modes, which span both

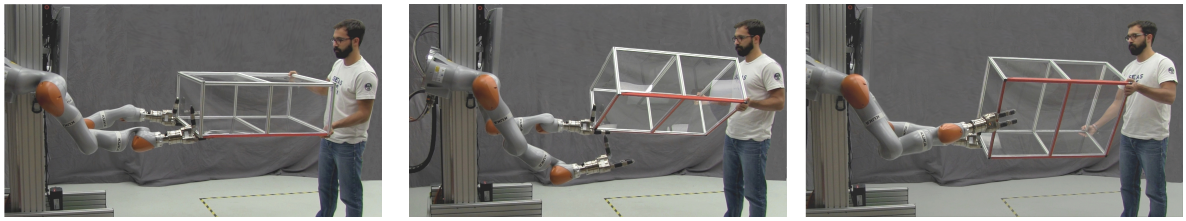
1) the hybrid action space that arises, due to the multi-contact nature of the task,
2) the multi-modal nature of joint-action planning, due to the non-stationary policy of the partner.

The pivotal aspects that enables the method to holistically optimize over such a complex and multi-modal space efficiently is the use of an informed-graph-search algorithm in combination with the decomposition of the hybrid motion into the HOLM primitives. The outer level's rules explore only the useful part of the solution space and with the HOLM primitives hybrid motion plans are optimized very efficiently.

We evaluated the method both in simulation and with an actual human-robot dyad. Both results demonstrate that the proposed method enables the robot agent to adapt its motion plans online, in response to real-time changes of the dyadic setup. These indicate the large potential of the method to be employed in general co-manipulation scenarios. Our future work will focus on methods to estimate the human intentions online and fully realize our vision presented in Fig. 3.

APPENDIX A
TRAJECTORY OPTIMIZATION METHODS

Trajectory optimization methods can be categorized as either indirect or direct. *Indirect methods* are based on the calculus of variations or the Maximum Principle [76]. They first use necessary conditions, usually as a boundary value problem in ordinary differential equations, and then discretize the resulting equation to obtain the optimal solution. For our problem, multiple path constraints (8e) have to be included, which is not straightforward using this transcription method. *Direct methods* for TO first discretize (8) and then use standard nonlinear optimization techniques to solve the resulting parametric problem [60]. Since standard optimization techniques are used, general constraints are easily incorporated. This comes with costs regarding the accuracy of the obtained solution, for which the required level is always application



(a) (b) (c)

Fig. 19. Keyframes of a $90°$ box rotation DcM scenario. The human and the robot jointly complete the task. (a) Initial configuration. (b) Contact change by the right arm. (c) The left arm has changed contact and the task is completed.

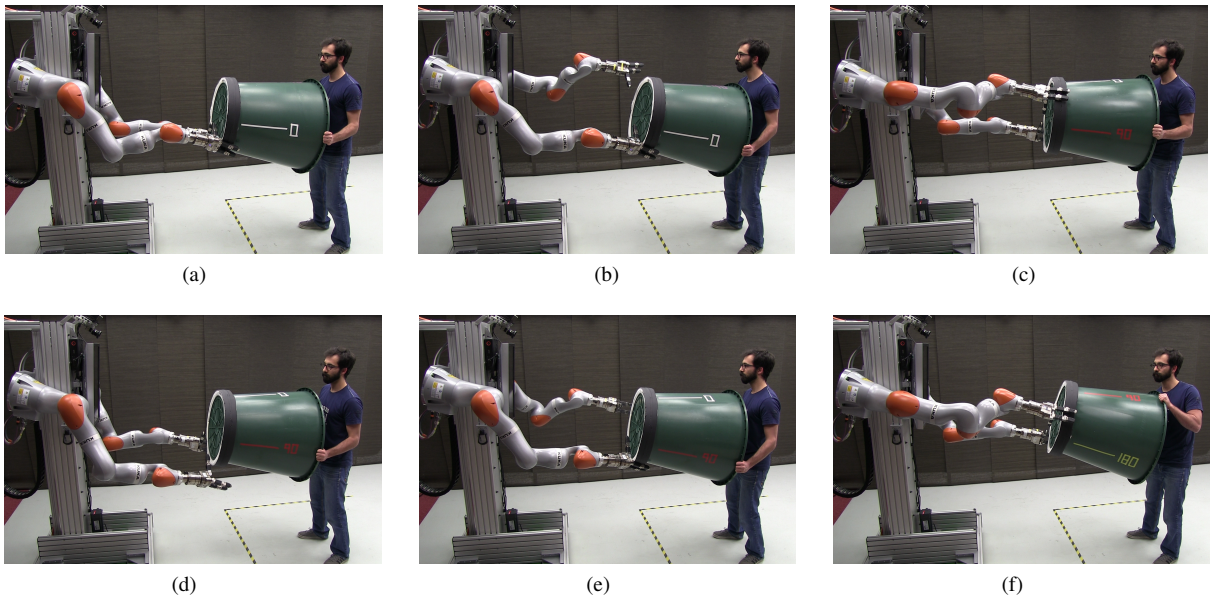


Fig. 20. Keyframes of a DcM task, where the human and the robot rotate a cylinder. (a) Initial state. (b) To realize the initial dyadic goal of orienting the cylinder at 90°, the robot performs a left arm contact change. (c) The partner's goal changes to a 180° orientation for the cylinder. (d), (e) The robot performs the new contact changes in accordance to the adapted joint plan. (f) The updated plan is completed given the latest human goal.

dependent, while the resulting problems are easier to pose and solve. Our method falls into the direct TO category.

Next, there are three main direct TO methods: shooting, transcription, and collocation [77]. In *direct shooting methods*, an integration scheme (*e.g.* an ODE solver) is used to eliminate state trajectory variables from the problem. As a result, problems in this class require only discretization of the control and possibly of the path constraints. To compute the state trajectory calls to an embedded integrator are needed, which first requires the integrator to provide sensitivities and second it can be especially problematic for unstable systems. To mitigate this, *direct multiple shooting methods* perform both a state and control discretization, while calls to an integrator are still used, albeit for a shorter horizon [78]. This leads to larger but structured nonlinear problems.

Direct transcription methods do not require calls to an embedded integrator; the discrete system's dynamics are enforced as constraints. This is achieved by discretizing both the controls and the states in a grid as well as the objective integral, where the grid points are called knots. Using direct transcription, problem (8) can be expressed as

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{u}} \quad & \sum_{i=0}^{N-1} c_i\left(\boldsymbol{x}_i, \boldsymbol{u}_i\right) + c_N\left(\boldsymbol{x}_N\right) \\ \text{s.t.} \quad & \boldsymbol{x}_{i+1} = \boldsymbol{f}\left(\boldsymbol{x}_i, \boldsymbol{u}_i\right) \\ & \boldsymbol{x}_0 \in \mathbb{X}_0 \\ & \boldsymbol{x}_N \in \mathbb{X}_N \\ & \boldsymbol{g}\left(\boldsymbol{x}_i, \boldsymbol{u}_i\right) \leq 0 \\ & i \in \{0, N\}, \end{aligned}$$

where the notation is the same as in Section IV-C. The optimization problem is typically large and sparse, and nonlinear solvers which exploit sparsity (`SNOPT` [79] or `IPOPT` [73]) can be used. Direct transcription methods have similar convergence characteristics with direct multiple shooting, are well parallelizable, and are preferred for problems with challenging path constraints. Their drawback is that the time discretization and the integration scheme should be carefully selected, since there is a trade-off between accuracy and computation effort.

Finally, in *direct collocation methods* both the input and the state are parameterized by piecewise polynomial functions (splines). Using these polynomial functions, the values of the state and control are computed outside of the knot points (typically at the midpoint of each segment), referred as collocation points. At these collocation points, the derivative of the spline is enforced to match the dynamics. Most commonly, first-order polynomials are used for the input and third-order for the state. Defining the collocation points at the midpoints of the spline allows their practitioners to compute the values of the state and control at the collocation points without computing the spline coefficients [80].

APPENDIX B
TRAJECTORY OPTIMIZATION THROUGH CONTACT

In trajectory optimization through contact, the hybrid nature of the intermittent contacts is usually expressed via a complementarity formulation defined as $0 \leq d \perp f \geq 0$, where d is a signed distance between the contacting objects and f is the constraint normal force between them. This states that only unilateral force can be exerted between the bodies, penetration is not allowable, and that situations involving no contact but contact force are excluded. Mathematical programs with complementarity constraints are in practise difficult to solve as they do not satisfy constraint qualifications and relaxations are usually needed [41].

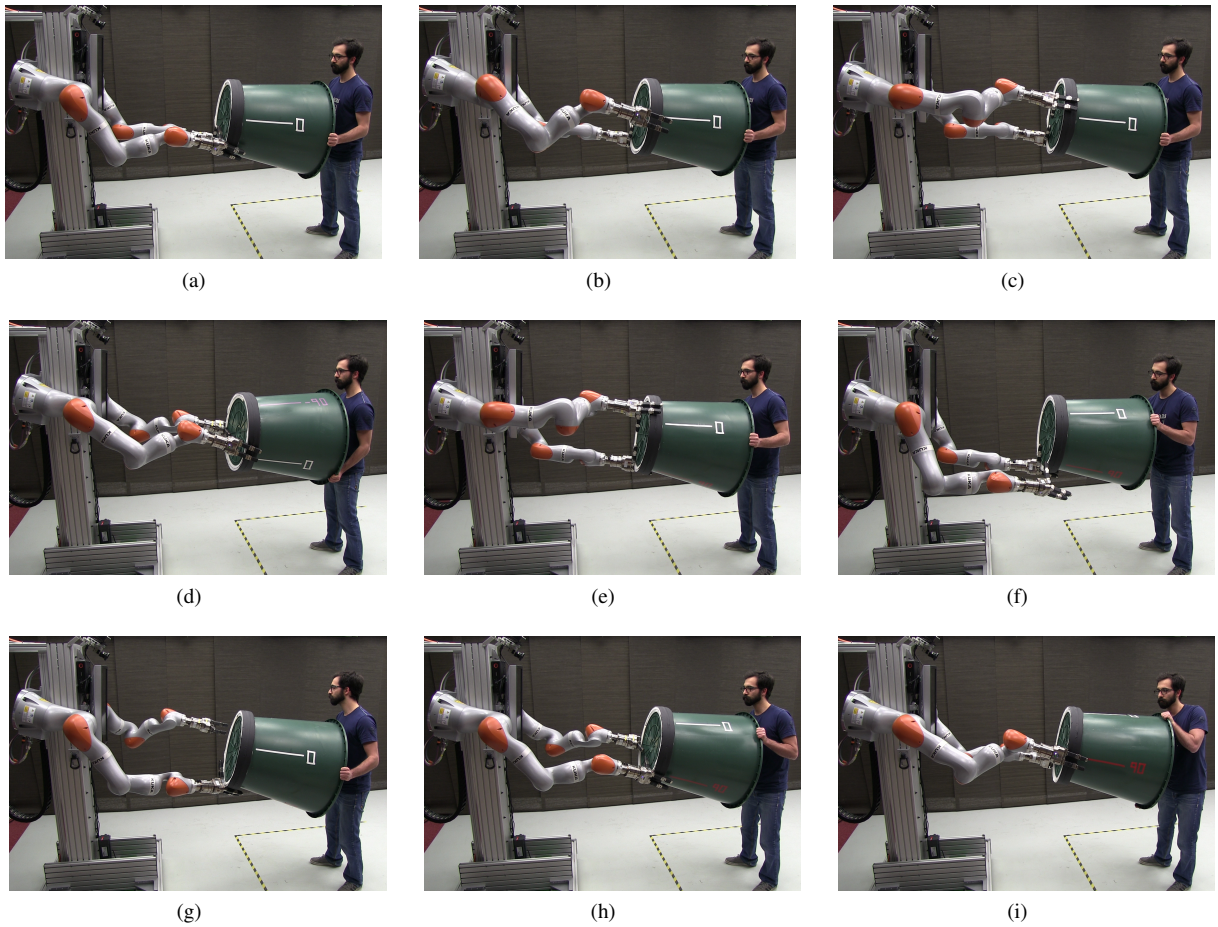


Fig. 21. A sequence of frames of a DcM scenario, where the human's initial goal is to orient the cylinder at $-45°$ and during the task execution his goal changes to $90°$. (a) Initial configuration. (b) Right arm during swing phase. (c) Right arm grasp hold change has completed. (d) Human and robot jointly rotate the cylinder towards the original goal. (e) Given the human's goal change, the adapted hybrid motion plan is in progress. (f) Right arm contact location change according to the updated plan. (g) Object's weight is transferred to the right arm and the left arm changes grasp hold. (h) All grasp hold changes have finished and the final object's rotation starts. (i) The dyad reaches the object's goal orientation.

ACKNOWLEDGMENT

This work is supported by the Honda Research Institute Europe, the Engineering and Physical Sciences Research Council (EPSRC: EP/L016834/1), EPSRC UK RAI Hub for Offshore Robotics for Certification of Assets (ORCA:EP/R026173/1) and UK-India Education and Research Initiative (UKIERI-DST2016-17-0152). The authors would like to thank Tamas Bates, Jiayi Wang and Evripidis Gkanias for their help with the experimental setups, João Moura for feedback and review of the drafts and the anonymous reviewers for their suggestions for improving the quality of the paper.

REFERENCES

[1] S. S. Obhi and N. Sebanz, "Moving together: toward understanding the mechanisms of joint action," *Experimental Brain Research*, vol. 211, no. 3, p. 329, May 2011.

[2] P. Dayan and N. D. Daw, "Decision theory, reinforcement learning, and the brain," *Cognitive, Affective, & Behavioral Neuroscience*, vol. 8, no. 4, pp. 429–453, Dec 2008.

[3] T. B. Sheridan, "Eight ultimate challenges of human-robot communication," in *the 6th International Symposium on Robot and Human Interactive Communication (RO-MAN)*. IEEE, Sept 1997, pp. 9–14.

[4] A. Ajoudani, A. M. Zanchettin, S. Ivaldi, A. Albu-Schäffer, K. Kosuge, and O. Khatib, "Progress and prospects of the human–robot collaboration," *Autonomous Robots*, vol. 42, no. 5, pp. 957–975, Oct 2018.

[5] J. R. Flanagan, M. C. Bowman, and R. S. Johansson, "Control strategies in object manipulation tasks," *Current opinion in neurobiology*, vol. 16, no. 6, pp. 650–659, Dec 2006.

[6] N. C. Dafle, A. Rodriguez, R. Paolini, B. Tang, S. S. Srinivasa, M. Erdmann, M. T. Mason, I. Lundberg, H. Staab, and T. Fuhlbrigge, "Extrinsic dexterity: In-hand manipulation with external forces," in *International Conference on Robotics and Automation (ICRA)*. IEEE, June 2014, pp. 1578–1585.

[7] T. Stouraitis, I. Chatzinikolaidis, M. Gienger, and S. Vijayakumar, "Dyadic collaborative manipulation through hybrid trajectory optimization," in *Conference on Robot Learning (CoRL)*, Oct 2018, pp. 869–878.

[8] A. Thobbi, Y. Gu, and W. Sheng, "Using human motion estimation for human-robot cooperative manipulation," in *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, September 2011, pp. 2873–2878.

[9] C.-M. Huang, M. Cakmak, and B. Mutlu, "Adaptive coordination strategies for human-robot handovers," in *Proceedings of Robotics: Science and Systems (RSS)*, July 2015.

[10] J. Lanini, H. Razavi, J. Urain, and A. Ijspeert, "Human intention detection as a multiclass classification problem: Application in physical human–robot interaction while walking," *IEEE Robotics and Automation Letters (RA-L)*, vol. 3, no. 4, pp. 4171–4178, Oct 2018.

[11] C. E. Madan, A. Kucukyilmaz, T. M. Sezgin, and C. Basdogan, "Recognition of haptic interaction patterns in dyadic joint object manipulation," *IEEE Transactions on Haptics (ToH)*, vol. 8, no. 1, pp. 54–66, Jan 2015.

[12] J. R. Hoare and L. E. Parker, "Using on-line conditional random fields to determine human intent for peer-to-peer human robot teaming," in *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Oct 2010, pp. 4914–4921.

[13] H. S. Koppula and A. Saxena, "Anticipating human activities using object affordances for reactive robotic response," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 38, no. 1, pp. 14–29, Jan 2016.

[14] J. Mainprice and D. Berenson, "Human-robot collaborative manipulation planning using early prediction of human motion," in *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Nov 2013, pp. 299–306.

[15] S. V. Albrecht and P. Stone, "Autonomous agents modelling other agents: A comprehensive survey and open problems," *Artificial Intelligence*, vol. 258, pp. 66 – 95, May 2018.

[16] D. J. Agravante, A. Cherubini, A. Bussy, P. Gergondet, and A. Kheddar, "Collaborative human-humanoid carrying using vision and haptic sensing," in *International Conference on Robotics and Automation (ICRA)*. IEEE, June 2014, pp. 607–612.

[17] M. Lawitzky, A. Mörtl, and S. Hirche, "Load sharing in human-robot cooperative manipulation," in *International Symposium on Robot and Human Interactive Communication (RO-MAN)*. IEEE, Sept 2010, pp. 185–191.

[18] G. Maeda, M. Ewerton, G. Neumann, R. Lioutikov, and J. Peters, "Phase estimation for fast action recognition and trajectory generation in human–robot collaboration," *The International Journal of Robotics Research (IJRR)*, vol. 36, no. 13-14, pp. 1579–1594, Feb 2017.

[19] L. Rozo Castañeda, S. Calinon, D. Caldwell, P. Jimenez Schlegl, and C. Torras, "Learning collaborative impedance-based robot behaviors," in *the Twenty-Seventh Conference on Artificial Intelligence (AAAI)*. AAAI, July 2013, pp. 1422–1428.

[20] D. Vogt, S. Stepputtis, B. Jung, and H. B. Amor, "One-shot learning of human–robot handovers with triadic interaction meshes," *Autonomous Robots*, vol. 42, no. 5, pp. 1053–1065, Feb 2018.

[21] Y. Maeda, T. Hara, and T. Arai, "Human-robot cooperative manipulation with motion estimation," in *International Conference on Intelligent Robots and Systems, (IROS)*. IEEE, Nov 2001, pp. 2240–2245.

[22] E. Gribovskaya, A. Kheddar, and A. Billard, "Motion learning and adaptive impedance for robot control during physical interaction with humans," in *International Conference on Robotics and Automation (ICRA)*. IEEE, May 2011, pp. 4326–4332.

[23] A. Ghadirzadeh, J. Bütepage, A. Maki, D. Kragic, and M. Björkman, "A sensorimotor reinforcement learning framework for physical human-robot interaction," in *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Oct 2016, pp. 2682–2688.

[24] A. Takagi, G. Ganesh, T. Yoshioka, M. Kawato, and E. Burdet, "Physically interacting individuals estimate the partner's goal to enhance their movements," *Nature Human Behaviour*, vol. 1, no. 54, pp. 2397–3374, Mar 2017.

[25] B. Busch, G. Maeda, Y. Mollard, M. Demangeat, and M. Lopes, "Postural optimization for an ergonomic human-robot interaction," in *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Sept 2017, pp. 2778–2785.

[26] L. Peternel, N. Tsagarakis, and A. Ajoudani, "A human–robot co-manipulation approach based on human sensorimotor information," *IEEE Transactions on Neural Systems and Rehabilitation Engineering (TNSRE)*, vol. 25, no. 7, pp. 811–822, July 2017.

[27] H.-C. Lin, J. Smith, K. K. Babarahmati, N. Dehio, and M. Mistry, "A projected inverse dynamics approach for multi-arm cartesian impedance control," in *International Conference on Robotics and Automation (ICRA)*. IEEE, May 2018, pp. 1–5.

[28] K. Otani, K. Bouyarmane, and S. Ivaldi, "Generating assistive humanoid motions for co-manipulation tasks with a multi-robot quadratic program controller," in *International Conference on Robotics and Automation (ICRA)*. IEEE, May 2018, pp. 3107–3113.

[29] E. Noohi, M. Žefran, and J. L. Patton, "A model for human–human collaborative object manipulation and its application to human–robot interaction," *IEEE Transactions on Robotics (T-RO)*, vol. 32, no. 4, pp. 880–896, Aug 2016.

[30] M. A. Roa and R. Suárez, "Grasp quality measures: review and performance," *Autonomous robots*, vol. 38, no. 1, pp. 65–88, July 2014.

[31] M. T. Mason, "Mechanics and planning of manipulator pushing operations," *The International Journal of Robotics Research (IJRR)*, vol. 5, no. 3, pp. 53–71, Sept 1986.

[32] S. Goyal, A. Ruina, and J. Papadopoulos, "Planar sliding with dry friction part 1. limit surface and moment function," *WEAR*, vol. 143, no. 2, pp. 307–330, Mar 1991.

[33] K. M. Lynch, H. Maekawa, and K. Tanie, "Manipulation and active sensing by pushing using tactile feedback," in *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Jan 1992, pp. 416 – 421.

[34] J. Zhou, Y. Hou, and M. T. Mason, "Pushing revisited: Differential flatness, trajectory planning, and stabilization," *The International Journal of Robotics Research (IJRR)*, vol. 38, no. 12-13, pp. 1477–1489, Sept 2019.

[35] F. R. Hogan, M. Bauza, and A. Rodriguez, "A data-efficient approach to precise and controlled pushing," in *Conference on Robot Learning (CoRL)*, Oct 2018, pp. 336–345.

[36] J. Zhou, R. Paolini, A. M. Johnson, J. A. Bagnell, and M. T. Mason, "A probabilistic planning framework for planar grasping under uncertainty," *IEEE Robotics and Automation Letters (RA-L)*, vol. 2, no. 4, pp. 2111–2118, June 2017.

[37] N. Chavan-Dafle, R. Holladay, and A. Rodriguez, "In-hand manipulation via motion cones," in *Robotics: Science and Systems (RSS)*, July 2018.

[38] M. T. Mason, "Toward robotic manipulation," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, pp. 1–28, Mar 2018.

[39] 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 Robotics and Automation Letters (RA-L)*, vol. 3, no. 3, pp. 1560–1567, Feb 2018.

[40] I. Mordatch, E. Todorov, and Z. Popović, "Discovery of complex behaviors through contact-invariant optimization," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 4, pp. 43–51, Jul 2012.

[41] M. Posa, C. Cantu, and R. Tedrake, "A direct method for trajectory optimization of rigid bodies through contact," *The International Journal of Robotics Research (IJRR)*, vol. 33, no. 1, pp. 69–81, Oct 2014.

[42] H. Dai, A. Valenzuela, and R. Tedrake, "Whole-body motion planning with centroidal dynamics and full kinematics," in *International Conference on Humanoid Robots (Humanoids)*. IEEE, July 2014, pp. 295–302.

[43] A. W. Winkler, C. Mastalli, I. Havoutis, M. Focchi, D. G. Caldwell, and C. Semini, "Planning and execution of dynamic whole-body locomotion for a hydraulic quadruped on challenging terrain," in *International Conference on Robotics and Automation (ICRA)*. IEEE, May 2015, pp. 5148–5154.

[44] S. Tonneau, A. Del Prete, J. Pettré, C. Park, D. Manocha, and N. Mansard, "An efficient acyclic contact planner for multiped robots," *IEEE Transactions on Robotics (T-RO)*, vol. 34, no. 3, pp. 586–601, June 2018.

[45] J. Carpentier, S. Tonneau, M. Naveau, O. Stasse, and N. Mansard, "A versatile and efficient pattern generator for generalized legged locomotion," in *International Conference on Robotics and Automation (ICRA)*. IEEE, May 2016, pp. 3555–3561.

[46] R. Deits and R. Tedrake, "Footstep planning on uneven terrain with mixed-integer convex optimization," in *International Conference on Humanoid Robots (Humanoids)*. IEEE, Sept 2014, pp. 279–286.

[47] A. K. Valenzuela, "Mixed-integer convex optimization for planning aggressive motions of legged robots over rough terrain," Ph.D. dissertation, Massachusetts Institute of Technology, 2016.

[48] F. R. Hogan and A. Rodriguez, "Feedback control of the pusher-slider system: A story of hybrid and underactuated contact dynamics," in *International Workshop on the Algorithmic Foundations of Robotics (WAFR)*, Dec 2016.

[49] N. Jarrassé, V. Sanguineti, and E. Burdet, "Slaves no longer: review on role assignment for human–robot joint motor action," *Adaptive Behavior*, vol. 22, no. 1, pp. 70–82, Sept 2013.

[50] S. Nikolaidis, A. Kuznetsov, D. Hsu, and S. Srinivasa, "Formalizing human-robot mutual adaptation: A bounded memory model," in *International Conference on Human Robot Interaction (HRI)*. ACM/IEEE, Mar 2016, pp. 75–82.

[51] T. Siméon, J.-P. Laumond, J. Cortés, and A. Sahbani, "Manipulation planning with probabilistic roadmaps," *The International Journal of Robotics Research (IJRR)*, vol. 23, no. 7-8, pp. 729–746, Aug 2004.

[52] J. E. King, J. A. Haustein, S. S. Srinivasa, and T. Asfour, "Nonprehensile whole arm rearrangement planning on physics manifolds," in *International Conference on Robotics and Automation (ICRA)*. IEEE, May 2015, pp. 2508–2515.

[53] M. Gienger, D. Ruiken, T. Bates, M. Regaieg, M. MeiBner, J. Kober, P. Seiwald, and A.-C. Hildebrandt, "Human-robot cooperative object manipulation with contact changes," in *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Oct 2018, pp. 1354–1360.

[54] S. Srivastava, E. Fang, L. Riano, R. Chitnis, S. Russell, and P. Abbeel, "Combined task and motion planning through an extensible planner-independent interface layer," in *International Conference on Robotics and Automation (ICRA)*. IEEE, June 2014, pp. 639–646.

[55] M. Toussaint and M. Lopes, "Multi-bound tree search for logic-geometric programming in cooperative manipulation domains," in *In-*

ternational Conference on Robotics and Automation (ICRA). IEEE, June 2017, pp. 4044–4051.
[56] D. S. Nau, V. Kumar, and L. Kanal, "General branch and bound, and its relation to A* and AO*," *Artificial Intelligence*, vol. 23, no. 1, pp. 29–58, 1984.
[57] M. Toussaint, K. Allen, K. Smith, and J. B. Tenenbaum, "Differentiable physics and stable modes for tool-use and manipulation planning," in *Robotics: Science and Systems (RSS)*, June 2018.
[58] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics (TSSC)*, vol. 4, no. 2, pp. 100–107, July 1968.
[59] M. Kelly, "An introduction to trajectory optimization: How to do your own direct collocation," *SIAM Review*, vol. 59, no. 4, pp. 849–904, Nov 2017.
[60] J. T. Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*, 2nd ed. SIAM, 2010.
[61] J. Nakanishi, A. Radulescu, D. J. Braun, and S. Vijayakumar, "Spatio-temporal stiffness optimization with switching dynamics," *Autonomous Robots*, vol. 41, no. 2, pp. 273–291, Jan 2016.
[62] S. Wang and K. Hauser, "Unified multi-contact fall mitigation planning for humanoids via contact transition tree optimization," in *International Conference on Humanoid Robots (Humanoids)*, Nov 2018, pp. 1–9.
[63] D. P. Bertsekas, *Dynamic Programming and Optimal Control*. Athena scientific Belmont, MA, 1995, vol. 1.
[64] S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.
[65] M. Al Borno, M. de Lasa, and A. Hertzmann, "Trajectory optimization for full-body movements with complex contacts," *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, vol. 19, no. 8, pp. 1405–1414, Aug 2013.
[66] A. D. Dragan, K. C. Lee, and S. S. Srinivasa, "Legibility and predictability of robot motion," in *International Conference on Human-Robot Interaction (HRI)*. ACM/IEEE, Mar 2013, pp. 301–308.
[67] R. Featherstone, *Rigid Body Dynamics Algorithms*. Springer, 2014.
[68] I. Chatzinikolaidis, T. Stouraitis, S. Vijayakumar, and Z. Li, "Nonlinear optimization using discrete variational mechanics for dynamic maneuvers of a 3d one-leg hopper," in *International Conference on Humanoid Robots (Humanoids)*. IEEE, Nov 2018, pp. 1–9.
[69] J. Pajarinen, V. Kyrki, M. Koval, S. Srinivasa, J. Peters, and G. Neumann, "Hybrid control trajectory optimization under uncertainty," in *International Conference on Intelligent Robots and Systems (IROS)*, Sep 2017, pp. 5694–5701.
[70] A. Takagi, F. Usai, G. Ganesh, V. Sanguineti, and E. Burdet, "Haptic communication between humans is tuned by the hard or soft mechanics of interaction," *PLOS Computational Biology*, vol. 14, no. 3, pp. 1–17, Mar 2018.
[71] M. Gienger, M. Toussaint, and C. Goerick, "Whole-body motion planning–building blocks for intelligent systems," in *Motion Planning for Humanoid Robots*. Springer, 2010, pp. 67–98.
[72] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, Jul 2018.
[73] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, Mar 2006.
[74] S. Liu, K. Mohta, N. Atanasov, and V. Kumar, "Search-based motion planning for aggressive flight in SE (3)," *IEEE Robotics and Automation Letters (RA-L)*, vol. 3, no. 3, pp. 2439–2446, Jan 2018.
[75] M. Likhachev, G. J. Gordon, and S. Thrun, "ARA*: Anytime A* with provable bounds on sub-optimality," in *Advances in Neural Information Processing Systems (NIPS)*. MIT Press, Dec 2003, pp. 767–774.
[76] A. E. Bryson, *Applied Optimal Control: Optimization, Estimation And Control*. Routledge, May 2018.
[77] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model Predictive Control: Theory, Computation, and Design*. Nob Hill Publishing, 2017.
[78] M. Diehl, H. Bock, H. Diedam, and P.-B. Wieber, "Fast direct multiple shooting algorithms for optimal robot control," in *Fast Motions in Biomechanics and Robotics: Optimization and Feedback Control*, ser. Lecture Notes in Control and Information Sciences, M. Diehl and K. Mombaur, Eds. Springer, Sept 2006, pp. 65–93.
[79] P. E. Gill, W. Murray, and M. A. Saunders, "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM Review*, vol. 47, no. 1, pp. 99–131, Aug 2005.
[80] C. Hargraves and S. Paris, "Direct trajectory optimization using nonlinear programming and collocation," *Journal of Guidance, Control, and Dynamics*, vol. 10, no. 4, pp. 338–342, Jul 1987.



Theodoros Stouraitis (S'19) received the Diploma degree in Computer Engineering and Informatics from the University of Patras, Greece, in 2011. From 2012 to 2015, he was research assistant at the Institute of Robotics and Mechatronics of the German Aerospace Center (DLR), focusing on sampling and optimization based grasp planning. In 2016 he received an M.Sc. in Artificial Intelligence from the University of Edinburgh in UK.

He is currently working toward the Ph.D. degree at the University of Edinburgh in UK and in collaboration with the Honda Research Institute Europe in Germany. His research interests include motion planning and control, hybrid systems and human-robot interaction.



Iordanis Chatzinikolaidis (S'13) received the Diploma degree in electrical and computer engineering and the M.Sc. degree in automation systems from the National Technical University of Athens, Athens, Greece, in 2015 and 2016, respectively.

He is currently pursuing a Ph.D. degree in robotics and autonomous systems from the University of Edinburgh, Edinburgh, UK. His research interests focus on the intersection of optimization, planning, dynamics, and control, especially for agile and physically interacting robotic systems.



Michael Gienger (M'16) received the Diploma degree in Mechanical Engineering from the Technical University of Munich, Germany, in 1998. From 1998 to 2003, he was research assistant at the Institute of Applied Mechanics of the TUM, addressing issues in design and realization of biped robots. He received his Ph.D. degree with a dissertation on "Design and Realization of a Biped Walking Robot". After this, Michael Gienger joined the Honda Research Institute Europe in Germany in 2003, where he currently holds a position as Chief Scientist. He served and serves as a scientific advisor for several large European research programs. His research interests include mechatronics, robotics, control systems, imitation learning and human-robot interaction.



Sethu Vijayakumar received his Ph.D. in Computer Science and Engineering from the Tokyo Institute of Technology, Japan in 1998. He is the Professor of Robotics at the University of Edinburgh, an adjunct faculty of the University of Southern California, Los Angeles and the founding Director of the Edinburgh Centre for Robotics. His research interests include statistical machine learning, anthropomorphic robotics, planning, multi objective optimisation and optimal control in autonomous systems as well as the study of human motor control. Professor Vijayakumar helps shape and drive the national Robotics and Autonomous Systems (RAS) agenda in his recent role as the Programme co-Director for Artificial Intelligence (AI) at The Alan Turing Institute, the UK's national institute for data science and AI. Sethu is a Fellow of the Royal Society of Edinburgh, a judge on BBC Robot Wars and winner of the 2015 Tam Dalyell Prize for excellence in engaging the public with science.