IEEE Robotics and Automation Letters (RAL) paper presented at the
2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Describing Physics *For* Physical Reasoning: Force-based Sequential Manipulation Planning

Marc Toussaint[1,2] and Jung-Su Ha[1,2] and Danny Driess[1,3]

***Abstract*— Physical reasoning is a core aspect of intelligence in animals and humans. A central question is what model should be used as a basis for reasoning. Existing work considered models ranging from intuitive physics and physical simulators to contact dynamics models used in robotic manipulation and locomotion. In this work we propose descriptions of physics which directly allow us to leverage optimization methods for physical reasoning and sequential manipulation planning. The proposed multi-physics formulation enables the solver to mix various levels of abstraction and simplifications for different objects and phases of the solution. As an essential ingredient, we propose a specific parameterization of wrench exchange between object surfaces in a path optimization framework, introducing the point-of-attack as decision variable. We demonstrate the approach on various robot manipulation planning problems, such as grasping a stick in order to push or lift another object to a target, shifting and grasping a book from a shelve, and throwing an object to bounce towards a target.**

I. Introduction

Reasoning is an essential form of generalization in AI systems, implying decision making competences in situations that are not part of the training data. Understanding the structure of reasoning problems in the real world therefore yields important insights in what kind of structure we might want to impose on learning systems for strong in-built generalization.

In this work we aim to contribute towards general-purpose physical reasoning, by which we mean performing inference over unknowns or controls given a model of physics, and constraints or objectives on (future) configurations. We believe that a core question is *how to model physics* for the purpose of physical reasoning. In other words, perhaps the pressing challenge is not to develop algorithmic solvers for *any* kind of forward model $\ddot{x} = f(x, \dot{x}, u)$ of physics, or any kind of physical simulator. Instead, the challenge is to formulate specific models and abstractions of physics that are appropriate for physical inference. We particularly touch on the issues of multi-physics descriptions and exposing a logic of physical interaction.

What are appropriate models of physics for enabling physical reasoning? Physics is usually described as a differential equation $\ddot{x} = f(x, \dot{x}, u)$ and simulation as numerical forward integration. In such forward descriptions of physics, reasoning becomes a problem of *inverting physics*, inferring decisions that lead to desired future configurations. Physics can also be described via constraints on correct paths, which we refer to as a *path description of physics*. Of course, any differential equation description of physics directly implies also a path description, namely by imposing the equality constraint $f(x, \dot{x}, u) - \ddot{x} = 0$ on the path. And a path description can, by iteratively solving only short horizon problems (MPC-like), be used to implement a forward simulator of physics.

In the case of contacts, forward models often describe f itself as a mathematical program, i.e. via a linear complementary problem [1], [2], or a Gaussian principle [3]. This is no issue for forward integration in simulators, and can also be made (piece-wise!) differentiable based on classical sensitivity analysis of NLP solutions [4], [5], [6]. But when translating this to path constraints this implies an equality constraint that itself contains a local mathematical program, which is known as bi-level optimization and makes long-term physical reasoning hard. Posa [7] thoroughly discussed the benefits of direct path optimization over bi-level optimization, with which we fully agree.

Is there just one correct model of physics for reasoning? The sciences describe physics on many levels of simplification: quantum field dynamics, fluid dynamics, rigid body Newtonian physics, quasi-static physics, toy-like physics as in some games, and intuitive conceptions of physics we find in humans [8]. It would be too limiting for physical reasoning to make use of only one particular abstraction of physics, or one particular physical simulator. Complex reasoning requires the reasoning process to deliberately apply different levels of simplification for different aspects of inference. One approach to enable this is by describing physics itself as if it would switch laws, as if physics would decide itself that certain objects follow sometimes Newtonian laws, while others follow blocks-world pick-and-place laws, and yet others follow quasi-static equations. We use the term *multi-physics*[1] to refer to this approach.

The contributions of this work are as follows:

1) We propose a novel approach to introduce decision variables for the wrench exchange between object surfaces

This work was supported by the Baden-Württemberg Stiftung (NEURO-ROBOTICS *DeepControl*) and via the Max Planck Fellowship (MPI for Intelligent Systems).

[1]Max Planck Institute for Intelligent Systems, 70569 Stuttgart, Germany. [2]Intelligent Systems Lab, TU Berlin, 10587 Berlin, Germany. [3]Machine Learning & Robotics Lab, University Stuttgart, 70569 Stuttgart, Germany.

[1]In the numerical simulation sciences, the term is used for simulations that involve multiple physical models, or multiple simultaneous physical phenomena, or multiple components where each is governed by its own principle(s). This typically refers to fully different types of physical interactions, such as fluid dynamics, electromagnetism, or chemical interactions. Our use of the term is yet more limited to just mixing stable and dynamic modes, rather than completely different areas of physics. But in both cases it conceptually means to leverage multiple physical models for physical modeling and reasoning.

in a path optimization framework. Our parameterization of wrenches, based on the *point-of-attack* (POA), can naturally handle any contact geometries (e.g., also line-to-line or point-to-surface geometries) and continuous transitions between them, and thereby allows the optimizer to find wrench interactions consistent with both, contact geometry and the Newton-Euler equations.

2) We propose a multi-physics approach to allow the optimizer to use various models of physical interaction for general physical reasoning, which include and integrate general force based interactions, quasi-static dynamics, and pick-and-place type interaction modes.

3) We extend Logic-Geometric Programming [9] to incorporate these interaction models and introduce additional decision variables for time-scaling between discrete configurations. In this way we enable joint optimization the path of configurations and their real-time scaling, which is essential as physics constraints can only be fulfilled when co-optimizing the timing of physical interactions.

We integrate these methods in a path optimization framework that takes a skeleton (a logic specification of the sequenced interactions) as input and tries to solve for a physically feasible and optimal path [10], [11], [12], [9]. Correct paths mix interaction modes of different abstraction levels for different object pairs in different time intervals, and we optimize full manipulation sequences across such switches in description. Using this, we demonstrate the approach on sequential dynamic and quasi-static manipulation problems, such as pushing with a stick, lifting a ring with a stick, toppling over a box, and sliding a book from a shelf before grasping. The breadth of tasks highlights the generality of the formulation.

After discussing related work in Sec. II, we detail our modeling approach in Sec. IV. Sec. V presents our demonstrations.

II. Related Work

A. Trajectory optimization through contacts

Trajectory optimization through contact interactions have previously been considered for footstep planning, dynamic locomotion, and manipulation [13], [7]. Specifically, in [7] point-to-point contacts are considered (patches are represented as multiple contact points) and one fundamental type of interaction (complementary contacts) is modeled. In comparison, our multi-physics formulation allows to mix interaction modes and our POA approach parameterizes the wrench between any two object surfaces, naturally including all contact geometries (point-to-point, point-to-line, line-to-line, point-to-surface, line-to-surface, surface-to-surface) and continuous transitions between them (see the box sliding example).

Contact-invariant optimization [14] has demonstrated impressive sequential manipulation plans. It relies on the pose difference between contact frames ($e_{i,t}(s)$ in their notation), which means that the supposed contact points (and contact geometry) have to be pre-fixed on the endeffector. Further, the contact invariant objective does not allow for sliding (because the relative pose velocity is penalized). Therefore it cannot not be applied to sliding or using a tool to exert forces (as no dedicated contact endeffector frame on the tool can be ad hoc fixed), which are both inherent in our demonstrations.

The core of our approach is to introduce a general parameterization of contact interactions that allows the optimizer to find wrench interactions consistent with both, contact geometry and the Newton-Euler equations.

B. Wrench exchange parameterization

A core question in describing force based interactions in path optimization is how precisely decision variables are introduced to represent wrench exchange. Fazeli et al. [2] considered general transmission of wrenches through contact patches or multiple contacts. However, in a path optimization setting, the number of contact points depends on the current geometry and is variable throughout optimization. To address this, Xie et al. [15] introduced the concept of the equivalent contact point, which subsumes wrenches exchanged via a line or surface contacts into a single contact point. With λ the forces and $d(q)$ the signed distance between two shapes, they start with generic complementarity $0 \leq \lambda \quad \perp \quad d(q) \geq 0$ (equation (8) in [15]), where $\perp$ means that at least one of the inequalities holds with equality. In their equations (9-13) this is then reformulated by introducing the effective contact point as two 3D positions a_1 and a_2, one on each shape surface, and constraining them to be inside the convex shape polytopes with explicit inequalities for each shape face. Analytic solutions for a single simulation step are then provided for the surface and line contact cases.

Our POA approach adopts the general idea of an effective contact point, but changes the formulation to only introduce a single 3D decision variable (namely the point of attach), formulate constraints based on a generic signed distance function rather than an explicit convex polytope description of shapes, and embedding the formulation in path optimization rather than building on analytic solutions for one step simulation.

C. Task and Motion Planning (TAMP)

Most existing TAMP approaches build on a discretization of the configuration space or action parameter space [16], [17], [18] and/or sample-based planners [19], [20], [21]. Our own previous work [9] proposed Logic-Geometric Programming (LGP), an optimization-based approach that can also plan tool-use and dynamic interactions using simplified Newtonian equations and impulse exchange. However, the particular physics description used in this previous LGP formulation is not general enough to enable broader physical reasoning. In particular, our previous work was missing force-based contact models, proper Newton-Euler equations, and quasi-static variants. The present paper proposes exactly such extensions and thereby aims to consider more generally what are appropriate building blocks in a multi-physics approach to sequential manipulation planning. We focus on the modeling questions and our experiments solve manipulation problems for a given skeleton. Therefore, what is proposed

here is only a component of a complete physical TAMP solver that also searches over skeletons, such as our Multi-Bound Tree Search [12].

D. Tool-use in animals and humans

Tool-use in animals and humans was described, e.g., in [22], [23], [24]. With our work we aim to provide computational methods to enable such reasoning. General physical reasoning in animals and humans is studied and discussed in [25], [26], [27], [28]. Particularly interesting is the discussion of simplistic and intuitive models of physics [8] that one might consider as being "incorrect", but which are effective heuristics for real-world reasoning and decision making. This motivated our approach of enabling multi-physics descriptions within a coherent planning framework.

Finally, one of our demonstration scenarios is inspired by [29], which leverages machine learning methods to enable real-time MPC control through planar manipulation interactions. Fig. 1 in [29] describes a scenario where a book is pulled from a shelf to enable a subsequent stable grasp, which exploits different contact modalities and motives our multi-physics description. We consider this scenario in section V-B.4.

III. Trajectory Optimization Framework

This section presents the path optimization framework as an extension of the LGP-formulation from [9] to allow optimizing for physical interactions. We optimize a path $x : [0, KT] \to \mathcal{X}$ consisting of $K \in \mathbb{N}$ phases or modes. A discrete variable s_k defines the constraints and costs in each phase k of the path. We call a sequence of discrete variables $s_{1:K}$ a *skeleton*. The configuration space $\mathcal{X} \subset \mathbb{R}^n \times SE(3)^m \times \mathbb{R}^{6\cdot n_{\text{cp}}} \times \mathbb{R}$ includes the n-dimensional generalized coordinate of robot links ($\mathbb{R}^n$) and the poses of m rigid objects ($SE(3)^m$), as in the original formulation [9]. However, in this work a configuration additionally includes wrench interactions for each of n_{cp} possible contact pairs ($\mathbb{R}^{6n_{\text{cp}}}$) as well as a single scalar $\tau \in \mathbb{R}$ with the following semantics: The continuous path x in the configuration space is indexed by the continuous path coordinate $t \in [0, KT]$, which corresponds to a virtual time, not real-world time. Hence the duration of the path in terms of the path coordinate is fixed to KT. To make this consistent with physics we jointly optimize for the time scaling τ, which defines the relation between the path coordinate t and real time, see Sec. IV-G for details.

Given a skeleton $s_{1:K}$, we solve the path problem

$$\min_{x:[0,KT]\to\mathcal{X}} \int_0^{KT} f_{\text{path}}(\bar{x}(t))\ dt \tag{1a}$$

$$\text{s.t.} \quad x(0) = x_0,\ h_{\text{goal}}(x(T)) = 0, \tag{1b}$$

$$\forall_{t\in[0,T]}:\ h_{\text{path}}(\bar{x}(t), s_{k(t)}) = 0,\quad g_{\text{path}}(\bar{x}(t), s_{k(t)}) \le 0 \tag{1c}$$

$$\forall_{k\in\{1,..,K\}}:\ h_{\text{switch}}(\hat{x}(t_k), s_{k\text{-}1}, s_k) = 0,\quad g_{\text{switch}}(\hat{x}(t_k), s_{k\text{-}1}, s_k) \le 0, \tag{1d}$$

Here, f_{path} define path costs, which in our case are squared accelerations of the robot joints plus additional regularizations, as described in Section IV-E, and $h_{\text{goal}}(x(T))$ define goal equality constraints. Further, $(h, g)_{\text{path}}$ define differentiable path constraints for a given mode s_k with $k(t) = \lfloor t/T \rfloor$, which depend on $\bar{x}(t) = (x(t), \dot{x}(t), \ddot{x}(t))$, and $(h, g)_{\text{switch}}$ define differentiable switch constraints between modes $s_{k\text{-}1}$ and s_k, which depend on $\hat{x} = (x, \dot{x}, \dot{x}')$, where $\dot{x}$ is the velocity before, and $\dot{x}'$ is the velocity after a switch (e.g. impulse exchange) [9].

To formulate physics laws in terms of real-time, the real-time velocities and accelerations can trivially be determined by the chain-rule, e.g.

$$\frac{\mathrm{d}x}{\mathrm{d}\tau}(t) = \frac{\partial x}{\partial t}(t)\frac{\mathrm{d}t}{\mathrm{d}\tau}(\tau(t)). \tag{2}$$

Our path solver, KOMO [10], parameterizes each configuration with only the minimal set of degrees-of-freedom (dofs). For instance, actual optimization variables for force exchange are only introduced when the skeleton introduces the existence of a force interaction. Therefore, the path solver deals with varying dofs in each mode and at each switch, which depend on the skeleton.

The following section details the specific dofs, inequality and equality constraints that are introduced into this path optimization formulation to by force-based interaction modes.

IV. Contact Models, Path Constraints, and Quasi-Static Motion

In this section we first formulate an efficient parameterization of force interactions using the *point-of-attack* (POA), and then specific forced and complementary contact models. Finally, we give more details on the optimizer employed.

A. Contact interaction modes

In our approach the skeleton [9] decides between which objects and in which phase contact interactions are accounted for. When contact interaction between a pair of objects is created, this has several effects on the resulting path problem: (1) A 6D decision variable (wrench, or force and POA) for each time step is introduced, (2) constraints are added to the path problem that describe physically correct forces and POA in consistency with the geometric configuration, and (3), if one of the objects are either in quasi-static or in dynamic motion mode, the effective wrench of the contact enters its quasi-static or dynamic motion constraint.

We provide several options to impose contacts during the manipulation. Specifically, we allow for a forced contact (requiring zero-distance throughout the interval), an instantaneous contact (active at one time slice only, realizing elastic impulse exchange), and the standard complementarity (enforcing complementarity throughout the interval). Each of these three contact modes comes in two versions, one allowing for slip, the other enforcing stick. We describe the details in the following.

B. Wrench as Force & Point-of-Attack

For each force interaction we introduce a 6D decision variable in the path optimization problem to represent the total wrench exchange between the two rigid bodies that enters their Newton-Euler equation. However, instead of parameterizing a wrench (f, ω) directly as linear force $f \in \mathbb{R}^3$ and torque $\omega \in \mathbb{R}^3$, we equivalently parameterize it as (f, p), where $p \in \mathbb{R}^3$ is the 3D point-of-attack (POA) in world coordinates (or zero-momentum point), with $\omega = f \times p$. This formulation follows the idea of the equivalent contact point [15] and resolves several issues that arise when deciding on force interaction during optimization.

We want to emphasize that the primary semantics of the POA is a parameterization of the exchanged 6D wrench in the Newton-Euler equations – at initialization or during optimization, the POA might well not be located on the object surfaces. Only at the point of convergence of optimization, and only if the exchanged force is non-zero, the POA fulfills the necessary geometric constraints (described below) to bear the semantics of a contact point representative. In other terms, the POA is a means to let the optimizer try to find a wrench exchange that is eventually consistent with both, contact geometry and the Newton-Euler equations.

This approach is in contrast to typical forward simulation models, where the point of force exchange is assumed to be at contact points computed from the current geometric configuration. However, contact point(s) computed from the geometric configuration are unstable (chaotic) for flat-on-flat interactions, cause jittering or bouncing, and raise serious convergence issues for path optimization.

Note that using the POA does not mean we limit ourselves to only point contacts. The POA parameterizes the wrench between two object surfaces: by constraining it (if force is exchanged) to lie on both object surfaces we can elegantly handle any contact configurations (point-to-point, point-to-line, line-to-line, point-to-surface, line-to-surface, surface-to-surface) and continuous transitions between them, as highlighted by the example of the box sliding over an edge. However, note that by constraining the POA to be on the surfaces and only allowing for a linear force f there, our current implementation excludes the possibility of a torque around the normal (e.g., from patch friction) [2] or wrenches via adhesion.

C. Forced and Complementary Contacts

We distinguish a *forced* contact and a *complementary* contact. When the skeleton imposes a forced contact, we add the constraints

$$d_1(p) = 0 \qquad \text{(POA is on object 1)} \tag{3}$$
$$d_2(p) = 0 \qquad \text{(POA is on object 2)} \tag{4}$$
$$d_{12} = 0 \qquad \text{(object 1 and 2 touch)}\ , \tag{5}$$

where $d_1(p)$ is the (signed) distance or penetration of p to the convex mesh of object 1; and d_{12} is the signed distance or penetration between two convex meshes. Both are evaluated with either GilbertJohnsonKeerthi (GJK) for non-penetrating objects, and Minkowski Portal Refinement (MPR) for penetrating objects.

When the skeleton imposes a complementary contact, we only add the 7D constraint

$$(d_1(p)f, d_2(p)f) = 0 \qquad \text{(force complementarity)} \tag{6}$$
$$d_{12} \geq 0 \qquad \text{(no penetration)}\ . \tag{7}$$

Note that complementarity is imposed with both(!) POA distances, not via the numerically less stable geometric distance d_{12}. The POA is directly a decision variable, with trivial Jacobian, whereas d_{12} is only piece-wise differentiable. But the combined constraints do eventually imply complementarity w.r.t. object touch.

D. Positivity, Slip and Elasticity

We always constrain the force to be positive,

$$-n^\top f \leq 0 \qquad \text{(force is positive)} \tag{8}$$

where $n \in \mathbb{R}^3$ is the normal of the pair's distance or penetration vector, which we retrieve differentiably from the witness simplices.

To model stick as well as elastic bounce we impose velocity constraints on the actual object surface points that relate to the POA. Let $V_1 = v_1 + w_1 \times (p - p_1)$ be the object-associated POA velocity, where (v_1, w_1) are the linear and angular velocities of object 1, and p_1 its center. Let $V = V_1 - V_2$ be the relative POA velocity between the interacting objects 1 and 2. For a non-slip contact we impose the equality constraint

$$(\mathbf{I} - nn^\top)V = 0 \quad \text{(zero tangential surface velocities)} \tag{9}$$

and the inequality constraint

$$||(\mathbf{I} - nn^\top)f||^2 < \mu^2 ||n^\top f||^2 \quad \text{(quadratic friction cone)} \tag{10}$$

where μ is the coefficient of friction in Coulomb's friction model. In contrast, for sliding contacts, the force f must be on the edge of the friction cone and its tangential component needs to align with the negative relative POA velocity, which both is ensured by

$$(\mathbf{I} - nn^\top)f = -a(\mathbf{I} - nn^\top)V, \tag{11}$$

for $a = \mu |n^\top f| / |V|$. Note that in our experiments we only consider very large (stick) or low (slip) friction. For frictionless contact this implies a normal force

$$(\mathbf{I} - nn^\top)f = 0 \qquad \text{(force is normal)}. \tag{12}$$

Finally, let V' be the relative POA velocity one time step later – e.g., after an instantaneous bounce. For an instantaneous bounce with elasticity coefficient β we add the constraint (cf. [30])

$$n^\top(V + \beta V') = 0 \qquad \text{(normal velocity reflection)} \tag{13}$$

In summary, using these equations we can impose forced, instantaneous, and complementary contacts, each with slip or stick.

E. Regularization costs

While we impose hard constraints to ensure physical correctness, we additionally can have soft penalties to favor smooth interactions. This can be interpreted as a prior on which kinds of robot manipulations we favor – for instance, those where the POA does not exceedingly jump around. Adding such regularizations has a positive effect on the convergence behavior of the solver. Specifically we add cost terms

$$\|\ddot{p}\|^2 \quad \text{(POA acceleration penalization)} \tag{14}$$
$$\|\ddot{f}\|^2 \quad \text{(force acceleration penalization)} \tag{15}$$
$$\|f\|^2 \quad \text{(force penalization)} \tag{16}$$
$$\|V_1 - V_2\|^2 \quad \text{(sliding velocity penalization)} \; . \tag{17}$$

F. Dynamic and Quasi-Static Motion

For every object that is in dynamic or quasi-static motion mode we collect the total 6D wrench F on its center that arises from all contacts with their forces and POAs. From the current path we compute the object's linear and angular velocity (v, w) and acceleration $(\dot{v}, \dot{w})$. Given the gravity vector $g \in \mathbb{R}^6$, the inertia matrix $M \in \mathbb{R}^{6\times6}$ and a potential friction λ, we have the Newton-Euler equation,

$$\begin{pmatrix} \dot{v} \\ \dot{w} \end{pmatrix} + g - M^{-1}(F - \lambda) = 0 \qquad \text{(Newton-Euler)}. \tag{18}$$

For free flight objects we assume $\lambda = 0$.

For the quasi-static sliding, we assume some friction λ such that the inertia forces can be ignored $(\dot{v}, \dot{w}) \equiv 0$. In particular, when an object is pushed by a manipulator on a table, this quasi-static model constrains the object's motion to be only along the surface of the table, $(v_z, w_{\phi,\psi}) = 0$. In the remaining degrees of freedom, the wrench applied by the manipulator exactly cancels out the friction, i.e., $F' = F_{x,y,\theta} = \lambda_{x,y,\theta}$, and is related to the velocity as

$$\begin{pmatrix} v_{x,y} \\ w_\theta \end{pmatrix} = k \nabla H(F') \tag{19}$$

with a convex function $H(F') : \mathbb{R}^3 \to [0, \infty]$. Note that, in this case, we need to consider the scaled wrench kF' as optimization variables instead of F' [31], [32]. In the experiment, we assumed the pressure is distributed constantly, thus utilized a simple quadratic representation of H: $H(F') = \frac{1}{2}(F')^T (M')^{-1} F'$.

G. Time scale optimization and impulse vs. force exchange

Finally, for the case of truly dynamic sequences, e.g. where a ball is bouncing on a table, it is physics that decides on the true time between two interactions. However, the skeleton imposes interactions at fixed steps. To resolve this conflict we introduce a scalar decision variable $\tau_t \in \mathbb{R}$ for every discretization step of the path which represents the real-time step in seconds. The optimizer can thereby find τ_t that scales a fixed path section to a correct physical time interval. We impose positive time evolution, $\tau_t \geq 0$, choose piece-wise constant time scaling $\dot{\tau}_t = 0$ within phases, and introduce a regularization $\|\tau_t - \hat{\tau}\|^2$ to favor time steps close to the default. All velocities mentioned above are differentiably evaluated by finite differencing along the path and dividing by τ_t. The acceleration terms $(\dot{v}, \dot{w}) + g$ in the Newton-Euler equation are actually multiplied by τ_t, which semantically makes it an impulse exchange equation, and the decision variable f associated with contacts actually represent impulses.

H. Integration with Existing Solver and Stable Interaction Models

The models described above were integrated in the existing solver described in [9]. This means that the above interaction modes can be mixed with modes for stable grasping and placement of objects. We also adopt the switch constraints, which enforce zero object accelerations at the switch, except for instantaneous contacts. As stated previously, in this work we focus on the path problem for a given skeleton, which is represented as a list of first order literals.

The resulting optimization problem is a non-linear mathematical program, which we address with an Augmented Lagrangian method [10] that exploits the sparse structure of the global path Hessian when computing Newton steps. We initialize the solver with the constant path (no object is moving) plus small Gaussian noise (0.01 sdv in joint space) to break symmetry. Restarts are used to find good local optima. Additional decision variables that are added due to the interactions (see Sec. IV-A) are initialized to match the initial scene poses (e.g., grasps have huge offset between endeffector and object). The introduced wrench decision variables are initialized as follows: linear forces are initialized as zero, while the POA is initialized as mean between the centers of the interacting objects.

V. Experiments

Please see the accompanying video[2] to get a first impression on our results. The source code to reproduce all examples in the video is available[3]. All experiments were done on an Intel i7-6500U CPU @ 2.50GHz.

A. Passive Tests (Pure Simulation as Path Optimization)

We start with first reporting on tests where there exists no robot or actuators, but path optimization is merely used to compute a physically feasible path. This tests whether our descriptions of physical interactions are appropriate to also perform ordinary physical simulation using path optimization.[4] While the tests are trivial scenarios, they give interesting insights in the method.

[2] https://youtu.be/tVFkKIIODaM

[3] https://github.com/MarcToussaint/20-IROS-ForceBased

[4] In fact, a physical simulator could be implemented using MPC based on our formulation, repeating path optimization of a short receding horizon. This would enable features such as co-optimizing the time stepping τ for the sake of simulation precision, or equality-constraining the simulation additionally on precise long-term energy conservation. We haven't explored further in this direction.

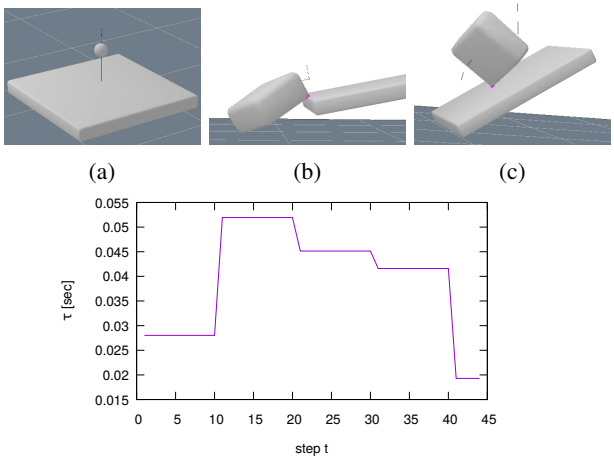


Fig. 1: Three passive simulation tests. (a) Bounding ball, (b) sliding over the edge, (c) tumbling box. (d) shows the time stepping optimization for (a).

1) Ball bouncing: We drop a ball onto a table (Fig. 1(a)), letting it bounce 4 times with elasticity coefficient $\beta = 0.9$ (where the outgoing velocity is constrained to be 90% of the incoming velocity). The accompanying video shows the simple behavior. The optimizer robustly converges within about $0.26s$ to standard precision, and within about 1 second to ultimate floating point precision in the constraints.

An insight we gain from this is that, in order to compute correct physical bounces, it is essential to include co-optimization of the time stepping τ. Fig. 1(d) shows τ_t for time steps $t = 1, .., T$, for $T = 45$. We see that the optimizer found different time scalings during each bounce interval. This is essential as the duration of the bounce is determined by physics and must be aligned with the imposed bounce schedule. This also means that those configurations at which bouncing contact is imposed are optimized to be at exactly the real times where the ball hits the table, allowing all constraints to be fulfilled to arbitrary precision even though we choose a very coarse time discretization. This addresses the typical issue in physical simulations of choosing efficient but imprecise fixed time steps versus adaptive step-sizes. Disabling stepping optimization makes our approach fail to find a correct solution for this simple bouncing problem.

2) Slide-falling and tumbling block: We present two passive examples that highlight the POA mechanism, a box sliding from a tilted table (solver time $2.41s$, Fig. 1(b)) and a box tumbling with sticky contact on a tilted table (solver time $0.49s$, Fig. 1(c)). In both cases we used general complementary contacts, and it was essential to allow the optimizer to find a suitable POA. Without the POA decision variable (when inserting the central witness point of the current configuration between the current shapes, computed with GJK or MPR, instead of p in all constraints), the solver was unable to find solutions in both scenarios. Using the POA, the optimizer finds (mostly smooth) POA movements on the surfaces.

B. Physical Manipulations

In the remainder we consider sequential robot manipulation scenarios. In all cases the manipulator model is a Franka Emika Panda. However, we abstracted the gripper's fingers as spheres. In the context of stable grasping, this is motivated by our experience that modeling the actual gripper closing with the actual finger geometries is hardly indicative of grasp success in real-world execution. Instead, ensuring a central opposing positioning to normal surfaces is simpler and transfers well. Abstracting the two fingers as spheres allows us to constrain that the nearest distance vector from the left finger-sphere to the object should exactly oppose the nearest distance vector from the right finger-sphere to the object. To this end we constrain the sum of both vectors to be zero, which describes a central opposition and has nice gradients to pull the gripper towards an opposing grasp.

1) Quasi-static pushing with a picked stick: In this scenario (Fig. 2(a)) the robot picks up a stick in order to push the box to the green target pose. The box motion is modeled as quasi-static table sliding. The pre-defined skeleton is

```
(oppose finger1 finger2 stick) (stable gripper stick)
(quasiStaticOn table box) (contact_slide stick box)
(poseEq box target)
```

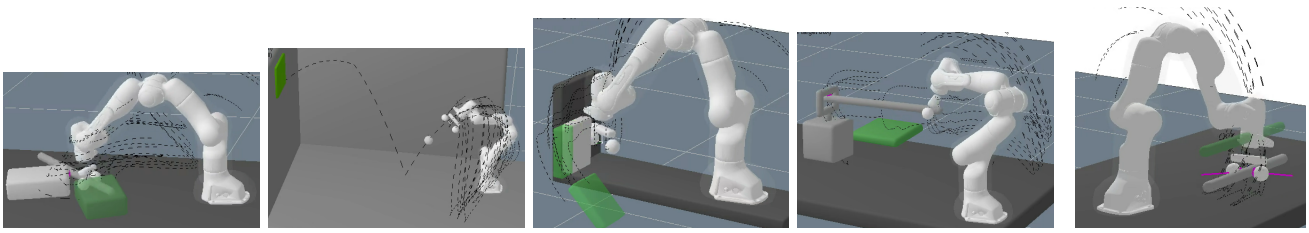
where each line corresponds to one phase step, the predicates stable and quasiStaticOn describe our mode switches, contact_slide the creation of a forced sliding contact, and oppose and poseEq are geometric constraints.

The solver finds (in $31.47s$) a rather involved pushing maneuver where the POA between the stick and the box is controlled to places that allow pushing the box into different directions. The video displays several additional pushing sequences, some with a free floating gripper, to show the variety of solutions found by the solvers. This scenario and the following two are cases where the solver benefits from mixing physics descriptions of varying abstraction: the stable grasp abstraction for the interaction with the stick, and force-based modeling for the interaction between box and stick, and quasi-static dynamics for the box. Our last experiment will investigate the gained efficiency of a stable grasp abstraction vs. a force-based grasp.

2) Dynamic ball throwing and bouncing to a target: This scenario (Fig. 2(b)) is an extension of the passive bouncing test discussed above. A robot picks up a ball to throw it onto the floor so that it bounces back against a wall, and then bounces to a given target. This highlights the ability to implicitly propagate back target constraints through force-based contacts to yield a correct throwing strategy. The pre-defined skeleton is

```
(oppose finger1 finger2 ball) (stable gripper ball)
(dynamic ball)
(bounce ball table)
(bounce ball wall)
(touch target ball)
```

which states that the first phase ends with grasping the ball, the ball becomes free and dynamic (Newton-Euler equations) after the second phase, the ball bounces with the table after the third, with the wall after the fourth, and touches (zero distance) the green target after the fifth.



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

Fig. 2: Physical robot manipulation demonstrations: (a) Picking a stick to push-rotate a box to a target pose; (b) picking a ball to throw it onto the floor so that it bounces back against a wall, and then bounces to a given target; (c) pushing a book forward to become graspable, pick it and move it to a target; (d) picking a stick to lift a box through a ring in order to place it onto a target; (e) basic force-based lifting a stick.

The solver finds a solution (in $16.3s$) where the robot, after picking up the ball, nicely accelerates and releases the ball to bounce to the target, as in 3D billiards. The velocity of the full sequence has to be rather fast as the free ball flight is governed by physics. Therefore, the control costs of this path are highly significant in this optimization problem. As seen in the video, the found solution varies drastically depending on the scaling of control costs.

3) Using a stick to lifting a weight at a ring to place it on a target: This scenario (Fig. 2(c)) aims to highlight the ability to create the needed contact points to achieve long term targets. A robot grasps a stick in order to insert it into a ring at the top of a weight. Thereby it can lift it and transport it onto a given target. The given skeleton is

```
(oppose finger1 finger2 stick) (stable gripper stick)
(dynamic box) (contact_slide stick ring)
(stableOn target box) (above target box)
```

where dynamic switches the box to free flying mode (Newton-Euler equations with contact forces as input), while stableOn then switches the box to reside stably on the target. above geometrically constraints the box center of mass to be within the target support.

The solver finds (in $10.0s$) a solution, where the robot finds the right spot for the stick to touch the ring so as to lift it. The box swings slightly during the dynamic transport to the target. The transition from its initial resting on the table to the dynamic phase is not perfectly smooth, which would require more careful regularization of accelerations at mode switches.

4) First pushing then grasping a book from a shelf: This scenario (Fig. 2(d)) is inspired from Fig. 1 in [29] and considers a book on a shelf that is initially too close to a wall to be grasped. So it first has to be pushed forward in order to allow to grasp it. The pre-defined skeleton is

```
(contact finger1 book) (quasiStaticOn shelf book)
(poseEq book subTarget)
(oppose finger1 finger2 book) (stable gripper book)
(poseEq book target)
```

Note that in this skeleton we predefined an intermediate target pose for the book, the first green pose seen in the video, as discussed in detail below.

Given this skeleton, the solver finds (in $16.4s$) a solution where the robot places the finger nicely to the right of the book to push it to the sub-target, then in a minimal motion transitions to the opposing grasp to lift the book and carry it to the final target.

As a negative result, if we remove the sub-target (2nd line) from the skeleton, the solver fails to find a feasible solution and typically converges to an infeasible solution that cheats when picking up the object, squeezing the finger between wall and book in a penetrating and book-jumping manner (see video). We considered extensively how we could fix this deficit of our method. However, we concluded that without cheating by redesigning the scenario to become less symmetric and have an intrinsic bias towards the first book slide, there is no way for our approach to solve this problem without introducing the sub-goal or some similar bias. The path optimization process has no implicit gradient towards paths that have consistent book slides in one or another direction. A random initialization is too unsystematic to pit optimization towards such slides. Instead, due to symmetry, path optimization is most likely to converge to the local optimum that corresponds to the shown infeasible solution.

We believe this scenario is highly insightful. Local optima are a fundamental issue for optimization and source of complexity for planning. The scenario shows that stronger biases would have to pre-exist, perhaps have been learned, to solve complex manipulation problems.

5) Force-based vs. stable grasping: In the previous experiments we imposed stable grasps. We can also solve for force-based grasping. In the last scenario (Fig. 2(e)) the robot only needs to lift the stick to a target pose. For the pre-defined skeleton with force-based contacts

```
(oppose fing1 fing2 stick) (contactStick fing1 stick)
  .. (contactStick finger2 stick) (dynamic stick)
(poseEq stick target)
```

it takes the solver $5.4s$ to find a lift. However, for the skeleton with stable grasp

```
(oppose finger1 finger2 stick) (stable gripper stick)
(poseEq stick target)
```

the solver requires only $0.24s$.

VI. Conclusions

In this work we propose concrete models for physical reasoning and robot manipulation planning which allow the solver to mix different abstractions for different objects and

phases of the solution, and integrate this in a path optimization framework to solve sequential physical manipulation problems over a wide range of scenarios. We call this a *multi-physics* model for reasoning. Our solver is based on a path description of physics that directly allows us to leverage constrained optimization methods.

A limitation of the approach is the still significant computation time needed to solve complex sequential physical interaction scenarios ($10 - 40s$ in our examples). This makes the naive integration into the full symbolic search of LGP unattractive. A promising alternative is to use our solver to generate large-scale data to learn a heuristic that can drastically accelerate search over potential interaction skeletons [33]. Further, this work only considers the problem of reasoning about possible manipulation sequences, not controller synthesis for a robust execution of such plans. Translating the framework to stochastic optimal control is yet subject to research [34].

ACKNOWLEDGMENTS

We would like to thank Alberto Rodriguez for inspiring us to work on some of the problems.

REFERENCES

[1] M. Anitescu and F. A. Potra, "Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems," *Nonlinear Dynamics*, vol. 14, no. 3, pp. 231–247, 1997.

[2] N. Fazeli, R. Kolbert, R. Tedrake, and A. Rodriguez, "Parameter and contact force estimation of planar rigid-bodies undergoing frictional contact," *The International Journal of Robotics Research*, vol. 36, no. 13-14, pp. 1437–1454, Dec. 2017.

[3] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference On*. IEEE, 2012, pp. 5026–5033.

[4] A. V. Fiacco and J. Kyparisis, "Sensitivity analysis in nonlinear programming under second order assumptions," in *Systems and Optimization*. Springer, 1985, pp. 74–97.

[5] A. B. Levy and R. T. Rockafellar, "Sensitivity of Solutions in Nonlinear Programming Problems with Nonunique Multipliers," in *Recent Advances in Nonsmooth Optimization*. WORLD SCIENTIFIC, Sept. 1995, pp. 215–223.

[6] A. F. Izmailov, "Solution sensitivity for Karush–Kuhn–Tucker systems with non-unique Lagrange multipliers," *Optimization*, vol. 59, no. 5, pp. 747–775, 2010.

[7] M. Posa and R. Tedrake, "Direct trajectory optimization of rigid body dynamical systems through contact," in *Algorithmic Foundations of Robotics X*. Springer, 2013, pp. 527–542.

[8] K. A. Smith, P. W. Battaglia, and E. Vul, "Different physical intuitions exist between tasks, not domains," *Computational Brain & Behavior*, vol. 1, no. 2, pp. 101–118, 2018.

[9] M. Toussaint, K. R. Allen, K. A. Smith, and J. B. Tenenbaum, "Differentiable physics and stable modes for tool-use and manipulation planning," in *Proc. of Robotics: Science and Systems (R:SS 2018)*, 2018, *Best Paper Award*.

[10] M. Toussaint, "A tutorial on Newton methods for constrained trajectory optimization and relations to SLAM, Gaussian Process smoothing, optimal control, and probabilistic inference," in *Geometric and Numerical Foundations of Movements*, J.-P. Laumond, Ed. Springer, 2017.

[11] ——, "Logic-geometric programming: An optimization-based approach to combined task and motion planning," in *Proc. of the Int. Joint Conf. on Artificial Intelligence (IJCAI 2015)*, 2015.

[12] M. Toussaint and M. Lopes, "Multi-bound tree search for logic-geometric programming in cooperative manipulation domains," in *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA 2017)*, 2017.

[13] R. Deits and R. Tedrake, "Footstep planning on uneven terrain with mixed-integer convex optimization," in *Humanoid Robots (Humanoids), 2014 14th IEEE-RAS International Conference On*. IEEE, 2014, pp. 279–286.

[14] I. Mordatch, Z. Popović, and E. Todorov, "Contact-invariant optimization for hand manipulation," in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Eurographics Association, 2012, pp. 137–144.

[15] J. Xie and N. Chakraborty, "Rigid body dynamic simulation with line and surface contact," in *2016 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)*. IEEE, 2016, pp. 9–15.

[16] T. Lozano-Pérez and L. P. Kaelbling, "A constraint-based method for solving sequential manipulation planning problems," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference On*. IEEE, 2014, pp. 3684–3691.

[17] F. Lagriffoul, D. Dimitrov, A. Saffiotti, and L. Karlsson, "Constraint propagation on interval bounds for dealing with geometric backtracking," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference On*. IEEE, 2012, pp. 957–964.

[18] F. Lagriffoul, D. Dimitrov, J. Bidot, A. Saffiotti, and L. Karlsson, "Efficiently combining task and motion planning using geometric constraints," *The International Journal of Robotics Research*, 2014.

[19] 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 *Robotics and Automation (ICRA), 2014 IEEE International Conference On*. IEEE, 2014, pp. 639–646.

[20] L. de Silva, A. K. Pandey, M. Gharbi, and R. Alami, "Towards combining HTN planning and geometric task planning," *arXiv preprint arXiv:1307.1482*, 2013.

[21] S. Alili, A. K. Pandey, E. A. Sisbot, and R. Alami, "Interleaving symbolic and geometric reasoning for a robotic assistant," in *ICAPS Workshop on Combining Action and Motion Planning*, 2010.

[22] W. Köhler, *Intelligenzprüfungen am Menschenaffen*. Springer, Berlin (3rd edition, 1973), 1917, english version: Wolgang Köhler (1925): *The Mentality of Apes*. Harcourt & Brace, New York.

[23] J. H. Wimpenny, A. A. S. Weir, L. Clayton, C. Rutz, and A. Kacelnik, "Cognitive Processes Associated with Sequential Tool Use in New Caledonian Crows," *PLOS ONE*, vol. 4, no. 8, p. e6471, Aug. 2009.

[24] F. Osiurak and A. Badets, "Tool use and affordance: Manipulation-based versus reasoning-based approaches." *Psychological review*, vol. 123, no. 5, p. 534, 2016.

[25] J. B. Hamrick, "Physical reasoning in complex scenes is sensitive to mass," PhD Thesis, Massachusetts Institute of Technology, 2012.

[26] J. Vonk, D. Povinelli, and C. Castille, "Social and Physical Reasoning in Human-reared Chimpanzees Preliminary Studies," Apr. 2013.

[27] P. W. Battaglia, J. B. Hamrick, and J. B. Tenenbaum, "Simulation as an engine of physical scene understanding," *Proceedings of the National Academy of Sciences*, vol. 110, no. 45, pp. 18 327–18 332, 2013.

[28] A. M. Seed and J. Call, "Space or physics? Children use physical reasoning to solve the trap problem from 2.5 years of age." *Developmental Psychology*, vol. 50, no. 7, pp. 1951–1962, 2014.

[29] F. R. Hogan, E. R. Grau, and A. Rodriguez, "Reactive Planar Manipulation with Convex Hybrid MPC," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 247–253.

[30] D. Stewart and J. C. Trinkle, "An implicit time-stepping scheme for rigid body dynamics with coulomb friction," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 1. IEEE, 2000, pp. 162–169.

[31] J. Zhou, M. T. Mason, R. Paolini, and D. Bagnell, "A convex polynomial model for planar sliding mechanics: theory, application, and experimental validation," *The International Journal of Robotics Research*, vol. 37, no. 2-3, pp. 249–265, 2018.

[32] M. Halm and M. Posa, "A quasi-static model and simulation approach for pushing, grasping, and jamming," in *Workshop on the Algorithmic Foundations of Robotics*, 2018.

[33] D. Driess, O. Oguz, J.-S. Ha, and M. Toussaint, "Deep visual heuristics: Learning feasibility of mixed-integer programs for manipulation planning," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2020.

[34] J.-S. Ha, D. Driess, and M. Toussaint, "Probabilistic framework for constrained manipulations and task and motion planning under uncertainty," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2020.