Differentiable Physics and Stable Modes for Tool-Use and Manipulation Planning – Extended Abstract*

Marc Toussaint[1,2†], **Kelsey R. Allen**[3], **Kevin A. Smith**[3] and **Joshua B. Tenenbaum**[3]
[1]Machine Learning & Robotics Lab, University of Stuttgart, Germany
[2]Max Planck Institute for Intelligent Systems, Stuttgart, Germany
[3]Massachusetts Institute of Technology, Cambridge, MA 02139
marc.toussaint@informatik.uni-stuttgart.de, {krallen, k2smith, jbt}@mit.edu

Abstract

We propose to formulate physical reasoning and manipulation planning as an optimization problem that integrates first order logic, which we call Logic-Geometric Programming.

1 Introduction

Physical reasoning is a corner stone of intelligence in humans and animals. A large body of animal behavior studies that aim at characterizing intelligence in animals focus on experiments related to physical reasoning in novel situations, such as tool use, means-end sequential object manipulation, an the anticipation of physical effects [Köhler, 1917; Wimpenny *et al.*, 2009]. Fundamentally, we are interested in AI representations and methods that would enable such general physical reasoning.

Evidence from cognitive science suggests that people have an "intuitive physics engine" [Battaglia *et al.*, 2013] which can be used to simulate the outcome of an action or tool manipulation [Osiurak and Badets, 2016], and have dedicated neural architectures near the motor cortex for implementing this capability [Fischer *et al.*, 2016]. But a typical physics engine only predicts outcomes for an action, not how to choose those actions.

Physical reasoning can be viewed as a problem of *inverting physics*, which means that we can formulate any objectives or constraints on the end result or trajectory of the simulation and solve for the inputs (control signals of an embedded robot, or parameters of the scene or kinematics) that render the desired constraints true. Fully (auto-) differentiable physical simulations are currently discussed as candidates for inverting physics [Todorov, 2011; Filipe de Avila Belbute-Peres and J. Zico Kolter, 2017], and could be embedded in end-to-end trainable systems [Tamar *et al.*, 2016].

However, we argue that gradients alone cannot solve the problem of inverting physics, for several reasons. From the NLP theory we know that their solutions are only piece-wise differentiable, where pieces are defined by stable constraint activity (discussed below). From the theory on path planning we know that our problem is NP complete [LaValle, 2006]. Finally, the problem is highly non-unimodal, where different contact sequences imply different local optima. Overall the structure inverse problem includes a combinatorics of local optimal and discontinuities in physical effects.

Our approach is to use logic to explicitly represent the combinatorics of possible physical interactions and respective local optima and differentiable modes. This follows the paradigm of Mixed-Integer Program (MIP) formulations in hybrid control synthesis [Deits and Tedrake, 2014]. However, it extends this to 1st-order logic, leveraging the strong generalization over objects of classical AI formulations. It also follows the standard task and motion planning (TAMP) approach of using logic to describe the task level, but now describes the combinatorics of possible physical interactions.

Given this general approach, the core of our method is to introduce explicit predicates and a PDDL-style decision rules that describe possible sequences of modes, as well as grounding these modes in differentiable constraints on the resulting path optimization problem.

In this extended abstract we focus on aspects that could not be discussed thoroughly in the original presentation [Toussaint *et al.*, 2018]. In particular, we first discuss why physics is only piece-wise differentiable, which we believe is key to understand the limitations of purely gradient-based methods. We then summarize the technical approach, including Multi-Bound Tree Search (MBTS), which was previously only described in [Toussaint and Lopes, 2017]. More specifically, we present the simplified version of MBTS that reflects the current implementation of our Logic-Geometric Program (LGP) solver[3], which pinpoints that the key for efficiency currently is in the formulation and solver of the various optimization sub-problems (the bounds, or "heuristics"), rather than in logic search itself. We then briefly summarize our experimental results, and discuss in more depth limitations and challenges with the current solver.

2 Physics Is Only Piece-Wise Differentiable

Physics can be described by a differential equation $\dot{x} = f(x)$ (neglecting external controls u). One of the core challenges for physical simulation is to integrate physical dynamics when hard objects interact. There are roughly three approaches: (1) smoothly modeling contact interactions as

*The original paper "Differentiable Physics and Stable Modes for Tool-Use and Manipulation Planning" was first presented at the Robotics: Science and Systems (R:SS 2018) conference in 2018.

†Contact Author

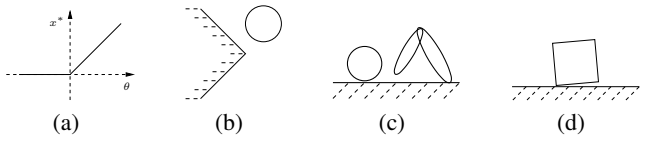

Figure 1: Illustrations for non-differentiable physics: (a) The solution map for $\min_x (x-\theta)^2$ s.t. $x \geq 0$, (b) a ball dropping close to a corner, (c) a ball might be touched or not by a robot, (d) jumping contact points for a cube.

spring-dampers and choosing very small time steps for integration; (2) detecting events (by backtracking penetrations) and applying impulse exchanges there (which leads to non-regular time stepping); and (3) choosing a fixed coarse time stepping scheme but formulating the next state $x_{t+1} = \operatorname{argmin}_x \mathcal{P}(x, x_t)$ as the solution of a non-linear constrained mathematical program $\mathcal{P}$, typically a so-called linear complementary problem (LCP) [Posa *et al.*, 2014]. The latter is the standard in control and many physical simulations.

In all cases, physical simulation is a forward iteration of computing x_{t+1} as a function of x_t. If each iteration is differentiable, we can back-propagate and compute gradients $\frac{dx_T}{dx_t}$ of any configuration x_T w.r.t. any previous configuration x_t (and analogously, any control signal or configuration parameter). A series of recent works aim to embed such differentiable physics simulation in trainable computation graphs [Filipe de Avila Belbute-Peres and J. Zico Kolter, 2017].

However, first, physics is not differentiable; it is only piece-wise differentiable. And second, typical objective functions have a combinatorics of local optima—and unlike to what seems to be the case in neural networks, here the majority of local optima are likely to be infeasibility traps and only one of many local optima might be an acceptable solution to the problem.

2.1 Differentiability of NLP Solutions

Concerning differentiability, let us consider the case where we simulate physics by iteratively solving $x_{t+1} = \operatorname{argmin}_x \mathcal{P}(x, x_t)$, for some non-linear mathematical program (NLP) $\mathcal{P}$. Each x_{t+1} fulfills the (Karush-Kuhn-Tucker) KKT conditions of $\mathcal{P}$. The classical literature on sensitivity analysis of NLPs considers the quasi-solution map $S : \theta \mapsto \{x : \text{KKT hold for } \mathcal{P}(\theta)\}$ of a parametrized NLPs $\mathcal{P}(\theta)$, and investigates how $S(\theta)$ varies with a change in θ. E.g., [Levy and Rockafellar, 1995] show that "under a standard constraint qualification [...], S is differentiable in a generalized sense, and we present a formula for its derivative". Also earlier work provides differentiability results [Fiacco and Kyparisis, 1985], and later work generalizations [Izmailov, 2010]. This classical work gives more insight in the role of *constraint qualification* for differentiability than more recent rediscoveries [Filipe de Avila Belbute-Peres and J. Zico Kolter, 2017]. As a conclusion, NLPs are only *piece-wise* differentiable. They are not differentiable across a change of constraint activity, e.g., a change of contact configuration in a physical simulation.

It is straight-forward to give simple illustrations of the non-differentiability of physical interaction. As a first minimal example, consider the problem $\min_x (x-\theta)^2$ s.t. $x \geq 0$. It has the solution map $S : \theta \mapsto x^* = \max\{0, x\}$, which is the ReLu illustrated in Fig. 1(a). This principle translates to physical settings: Figure (b) illustrates a ball dropping close to a corner, where the end state will bifurcate depending on whether it hits the corner or not. Figure (c) is similar, where the ball's end-state depends on the robot joint angles, depending on whether the robot touched the ball or not. And in Figure (d) the contact points jump with the cube's angle, which leads to bifurcation and chaos, if iterated.

Our LGP approach is to use logic to chop the overall problem of inverting physics into its (smoothly) differentiable pieces, which we call modes. It uses logic to explicitly enumerate these modes.

2.2 The Zero Gradient Problem, Relaxation, and Local Optima

A physical world of hard objects has a very particular structure: objects interact only when they are in contact. Interaction should here be understood in terms of gradient propagation: The gradient of an object w.r.t. any aspect of another object at any previous time is zero, unless there was some (chain of) contacts over time. Therefore, even if a configuration may have thousands of degrees of freedom, gradient propagation between these is rather sparse.

Purely gradient-based planning for physical interaction planning has a serious problem with zero gradients. If the initial configuration, or rather, the configuration path initialization, does not already have an appropriate chain of contact interactions to influence certain target objects, the gradient of goal objective is exactly zero. Therefore, the gradient does not help to decide on which objects should interact in a plan.

Previous work has proposed to relax the interaction of objects and proposed force exchange models that smoothly decay with the distance between objects. This relaxation can be scheduled to be very smooth in the initial phase of optimization, which might allow the optimizer to find a rough interaction plan, and tighten it during optimization to the limit of the exact hard contact model [Mordatch *et al.*, 2012]. However, we should be clear that this approach still is greedy: it will fixate on the first interaction schedule that gradient descend in the initial relaxed phase converged to, without any mechanisms to escape local optima. The problem of deciding on an interaction sequence falls into the category of classical PDDL planning, and it should be clear that gradient descent on a relaxation cannot solve this inherently NP hard problem.

In LGP we employ classical search over possible mode sequences and then constrain the mathematical program to converge to a solution within this mode sequence, exploiting the smooth differentiability of this mode even in the early iterations of optimization that are not yet within the mode. The key for efficiency for this approach are good heuristics for search, where we propose using a hierarchy of simpler mathematical problems that define lower bounds.

3 Logic-Geometric Programming

We consider the configuration space $\mathcal{X} = \mathbb{R}^n \times SE(3)^m$ of an n-dimensional robot and m rigid objects, in an initial con-

figuration $x_0 \in \mathcal{X}$. We aim to find a path $x : [0, T] \to \mathcal{X}$

$$\begin{aligned} \min_{x} \quad & \int_0^T f_{\text{path}}(\bar{x}(t))\, dt + f_{\text{goal}}(x(T)) \\ \text{s.t.} \quad & x(0) = x_0,\ h_{\text{goal}}(x(T)) = 0,\ g_{\text{goal}}(x(T)) \le 0\ , \\ & \forall t \in [0, T] :\ h_{\text{path}}(\bar{x}(t)) = 0,\ g_{\text{path}}(\bar{x}(t)) \le 0\ , \end{aligned} \tag{1}$$

where $(h, g)_{\text{path}}$ define path constraints which depend on $\bar{x}(t) = (x(t), \dot{x}(t), \ddot{x}(t))$ and describe what is physically and kinematically feasible. The function f_{path} defines control costs, which in our experiments we choose as sum-of-squares of joint accelerations. And $(f, h, g)_{\text{goal}}$ specify arbitrary objectives or constraints on the final configuration. In our experiments this will be a single equality constraint expressing contact between an object and a target.

We augment this formulation with additional logic decision variables $a_{1:K}, s_{1:K}$, so that we may assume that the path constraints $(h, g)_{\text{path}}$ are *smooth* for constant logical state s_k, and we have smooth constraints for transitioning between logical states. We call this a Logic-Geometric Program [Toussaint, 2015; Toussaint and Lopes, 2017; Toussaint *et al.*, 2018],

$$\begin{aligned} \min_{x, a_{1:K}, s_{1:K}} \quad & \int_0^T f_{\text{path}}(\bar{x}(t))\, dt + f_{\text{goal}}(x(T)) \\ \text{s.t.} \quad & x(0) = x_0,\ h_{\text{goal}}(x(T)) = 0,\ g_{\text{goal}}(x(T)) \le 0, \\ & \forall t \in [0, T] :\ h_{\text{path}}(\bar{x}(t), s_{k(t)}) = 0, \\ & \qquad\qquad g_{\text{path}}(\bar{x}(t), s_{k(t)}) \le 0 \\ & \forall k \in \{1, .., K\} :\ h_{\text{switch}}(\hat{x}(t_k), a_k) = 0, \\ & \qquad\qquad g_{\text{switch}}(\hat{x}(t_k), a_k) \le 0, \\ & \qquad\qquad s_k \in \text{succ}(s_{k\text{-}1}, a_k)\ . \end{aligned} \tag{2}$$

Here, $\hat{x} = (x, \dot{x}, \dot{x}')$ captures the configuration velocity $\dot{x}$ before and after ($\dot{x}'$) an instantaneous impulse exchange at a state transition.

In practice, feasible logical decision sequences $a_{1:K}$ are described in PDDL. We call the sequence $a_{1:K}$ a *skeleton* [Lozano-Pérez and Kaelbling, 2014], which uniquely defines $s_{1:K}$, and use the notation $\mathcal{P}(a_{1:K})$ to denote the path optimization problem (2) for a given skeleton. As $(h, g)_{\text{path}}$ and $(h, g)_{\text{switch}}$ are smooth conditional to the logic decisions, all objectives and constraints in $\mathcal{P}(a_{1:K})$ are smooth and our implementations are differentiable to provide constraint Jacobians and pseudo-Hessians when the costs $f_{\text{path,goal}}$ are sum-of-squares terms. Solving $\mathcal{P}(a_{1:K})$ implicitly solves for all action parameters jointly and optimally: E.g. when the sequence involves a grasp first, a hit second, and a placement third, then all parameters of these actions (grasp pose, hitting angle, placement pose) are jointly optimized to yield the overall optimal manipulation path.

In [Toussaint *et al.*, 2018] we detail the concrete predicates used to represent geometric constraints (touch, inside, above) and constraints on the system dynamics (e.g., impulse, stableFree, dynamicFree). For instance, dynamicFree imposes the Newton-Euler equation on a particular object, impulse imposes instantaneous impulse exchange between two objects, and touch imposes the distance between two convex shapes to be zero. In this way, logic predicates are grounded as differentiable constraints on the resulting path optimization problem.

4 Solver

We assume we have a generic NLP solver[1] which, for any NLP $\mathcal{P}$ returns either *infeasible* or a feasible solution x and cost $f_{\mathcal{P}}(x)$. Note that the NLP solver does not return further insight in which constraints rendered a solution infeasible, what would be a minimal constrained removal for feasibility, or what are the origin of costs. The higher-level algorithm can only send $\mathcal{P}$-queries to the NLP solver and check feasibility and costs.

As higher-level algorithm we use Multi-Bound Tree Search (MBTS) [Toussaint and Lopes, 2017], which leverages a hierarchy of bounds to prioritize search and NLP computations, thereby integrating the concepts of branch-and-bound from MIP. We sketch the method here briefly and defer to the original publication for more details.

Assume that purely symbolic tree search (based on the given PDDL) has found a node that represent a potential solution skeleton $a_{1:K}$. Trying to solve the full path problem $\mathcal{P}(a_{1:K})$ is in the order of roughly ~ 10 seconds and far too expensive as a heuristic to guide search. We therefore define a hierarchy of lower bounds of $\mathcal{P}(a_{1:K})$, each of which is a simplification of the full path problem, which essentially drops constraints, cost terms and decision variables and is straight-forward to prove to be a lower bound. Specifically, the lower bound $\mathcal{P}_1(a_{1:k})$ optimizes only over a single world configuration $x(t_k)$ for $k \in \{1, .., K\}$, a generalized form of inverse kinematics, and we can compute this efficiently for all $k \in \{1, .., K\}$ to check feasibility of intermediate configurations. If all these are feasible, we then evaluate feasibility of $\mathcal{P}_2(a_{1:K})$, which jointly optimizes over all intermediate configurations (keyframes) $(x(t_1), .., x(t_K))$, but neglects the motion paths connecting them. Only if $\mathcal{P}_2(a_{1:K})$ is also feasible, we eventually try solving for the full path $\mathcal{P}(a_{1:K})$.

MBTS maintains best-first queues for tree expansion as well as for computations of poses ($\mathcal{P}_1$), sequences ($\mathcal{P}_2$), and paths ($\mathcal{P}$), and schedules computations following the above principles. Importantly, it also prunes large parts of the tree if some pose computations ($\mathcal{P}_1$) turn out infeasible: Similar to the infeasible in [Srivastava *et al.*, 2014] we add a predicate to the PDDL description that rules out the corresponding action also in other branches of the tree.

5 Summary of Experiments

We point the reader to[2] for a collection of videos demonstrating the approach, and here[3] for the available source code.

We investigated our method on 6 problems as illustrated and described in Fig. 2. The problems were designed to

[1] We use KOMO [Toussaint, 2017], an Augmented Lagrangian quasi-Newton method.

[2] Videos: http://ipvs.informatik.uni-stuttgart.de/mlr/lgp/ https://www.youtube.com/watch?v=-L4tCIGXKBE

[3] Source code: https://github.com/MarcToussaint/rai-python https://github.com/MarcToussaint/18-RSS-PhysicalManipulation

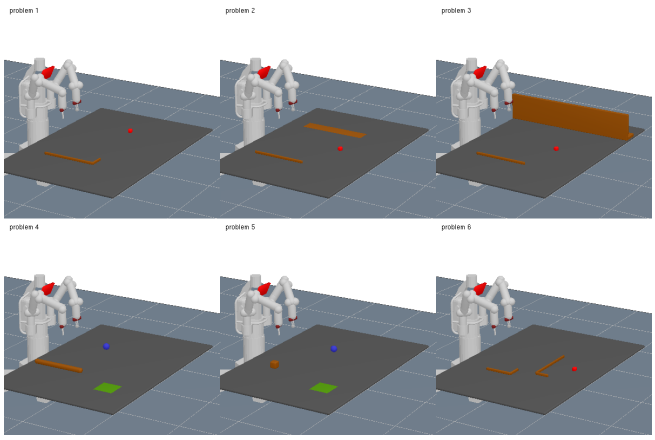

Figure 2: In the 6 investigated problems the goal is to reach for the red ball or get the blue ball in the green area. Solutions involve using a hook to pull a desired object, push-sliding a ball along a wall, pushing a ball onto a strip of paper to then pull it closer, hitting a ball with a stick, throwing a box at a ball, and using a hook to reach for another hook to reach a ball.

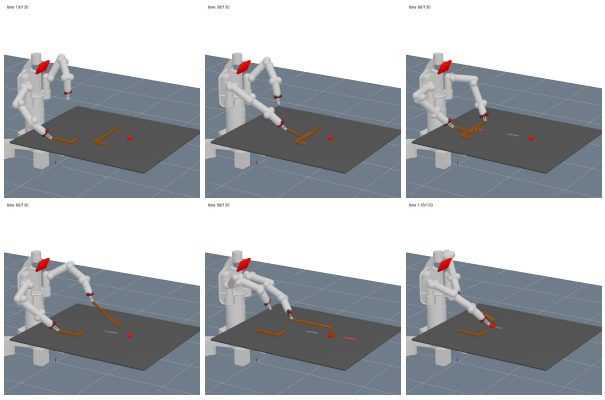

Figure 3: Example sequence of key frames (subject to $\mathcal{P}_2$) for the double hook problem.

cover a spectrum of types of interactions, including the need to use tools, hit objects, or throw objects in order to reach the goal. Fig. 3 displays a sequences of key frames for the double hook problem; it is these keyframes that are optimized in $\mathcal{P}_2$. The accompanying videos illustrate solutions to the problems. The source code includes the precise scene descriptions. Note that the videos display the computed paths x, not a "simulation" of their execution. The fact that they "look like a simulation" shows that the constraints we impose on paths result in physically plausible paths.

The solver surprised us in finding much larger varieties of solutions than anticipated. For instance, in problem 1 a natural solution is grasping the hook, pulling the ball, and grasping it. Our method also found solutions that involve hitting the ball, or sliding the hook to the ball to hit it. Handovers are much more frequent than anticipated. The solver exploited the combinatorics of manipulations that are possible with the given primitives beyond what we had in mind when designing the problems.

The solver is an anytime method. For each problem we ran it until 12 solutions (with different skeletons) were found or 400 seconds were exceeded. The solver then presents and ranks solutions by their cost. First reasonable solutions are roughly found after 20-200 seconds, depending on depth of the simplest solution. Interestingly, a major amount of computation time is spend on trying to solve NLPs that eventually turn out infeasible. Although the sub-problems are not guaranteed to be convex, we empirically tested how often restarts converge to the same optimum and found a high degree of consistency of convergence. See [Toussaint *et al.*, 2018] for more details.

6 Discussion

Our approach is, to our knowledge, the first to embed dynamic physical manipulations in a task and motion planning (TAMP) framework, combining a discrete logic level for sequences of possible interaction modes with a continuous path optimization level. We tackled sequential manipulation and tool-use planning problems, a hallmark of intelligent behavior. But the approach also raises a series of fundamental questions for future research, which we want to discuss here.

The current approach focuses on planning against a deterministic model of physics. It therefore neither constructs a reactive controller for execution, nor estimates the robustness of plans under stochasticity. Extending our path optimization approach to account for probabilities, e.g., to a stochastic optimal control setting, is therefore a core challenge. How to effectively represent beliefs, that is, represent possible path distributions, including the combinatorics implicit in the logic as well as the complementarity condition of the KKT conditions seems a key question.

In the present study we defined a concrete set of possible modes by hand, including specific modes for stable grasps and placements. We experimented with using only a single and fundamental mode predicate which only represents contact between objects and lets the NLP solver decide on exchanged forces (using complementarity formulations similar to [Posa *et al.*, 2014]). The videos[2] demonstrate simple results. However, while being very general, it scales much worse than the hand-designed interaction modes. The potential to learn efficient interaction modes is an interesting avenue for future research.

The present MBTS solver uses trivial tree search instead of efficient PDDL solvers to find potential skeletons. This can easily be replaced (assuming that PDDL solvers can be made to enumerate all possible symbolic solutions), but computation time spent on logic search is marginal in the current applications. Scaling to larger domains is limited much more by the large NLPs to be solved. Our approach can be viewed as an extreme, where the eventual solution jointly optimizes over all decision variables, whereas all previously existing TAMP solvers construct solutions in a very decomposed manner. Scaling to larger domains will require finding a compromise between joint optimization and full decomposition.

References

[Battaglia *et al.*, 2013] Peter Battaglia, Jessica Hamrick, and Joshua B Tenenbaum. Simulation as an engine of physical scene understanding. *Proceedings of the National Academy of Sciences*, 110:18327–18332, 2013.

[Deits and Tedrake, 2014] Robin Deits and Russ Tedrake. Footstep planning on uneven terrain with mixed-integer convex optimization. In *RAS Int. Conf. on Humanoid Robots*. IEEE, 2014.

[Fiacco and Kyparisis, 1985] Anthony V. Fiacco and Jerzy Kyparisis. Sensitivity analysis in nonlinear programming under second order assumptions. In *Systems and Optimization*, pages 74–97. Springer, 1985.

[Filipe de Avila Belbute-Peres and J. Zico Kolter, 2017] Filipe de Avila Belbute-Peres and J. Zico Kolter. A Modular Differentiable Rigid Body Physics Engine. In *Neural Information Processing Systems (NIPS'17)*, 2017.

[Fischer *et al.*, 2016] Jason Fischer, John G Mikhael, Joshua B Tenenbaum, and Nancy Kanwisher. Functional neuroanatomy of intuitive physical inference. *Proceedings of the national academy of sciences*, 113:E5072–E5081, 2016.

[Izmailov, 2010] Alexey F Izmailov. Solution sensitivity for Karush–Kuhn–Tucker systems with non-unique Lagrange multipliers. *Optimization*, 59(5):747–775, 2010.

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

[LaValle, 2006] Steven M. LaValle. *Planning Algorithms*. Cambridge university press, 2006.

[Levy and Rockafellar, 1995] Adam B Levy and R. Tyrrell Rockafellar. Sensitivity of Solutions in Nonlinear Programming Problems with Nonunique Multipliers. In *Recent Advances in Nonsmooth Optimization*, pages 215–223. WORLD SCIENTIFIC, September 1995.

[Lozano-Pérez and Kaelbling, 2014] Tomás Lozano-Pérez and Leslie Pack Kaelbling. A constraint-based method for solving sequential manipulation planning problems. In *Int. Conf. on Intelligent Robots and Systems (IROS'14)*, pages 3684–3691. IEEE, 2014.

[Mordatch *et al.*, 2012] Igor Mordatch, Emanuel Todorov, and Zoran Popović. Discovery of complex behaviors through contact-invariant optimization. *ACM Transactions on Graphics (TOG)*, 31(4):43, 2012.

[Osiurak and Badets, 2016] François Osiurak and Arnaud Badets. Tool use and affordance: Manipulation-based versus reasoning-based approaches. *Psychological Review*, 123:534, 2016.

[Posa *et al.*, 2014] Michael Posa, Cecilia Cantu, and Russ Tedrake. A direct method for trajectory optimization of rigid bodies through contact. *The International Journal of Robotics Research*, 33(1):69–81, 2014.

[Srivastava *et al.*, 2014] Siddharth Srivastava, Eugene Fang, Lorenzo Riano, Rohan Chitnis, Stuart Russell, and Pieter Abbeel. Combined task and motion planning through an extensible planner-independent interface layer. In *Int. Conf. on Robotics and Automation (ICRA'14)*, pages 639–646. IEEE, 2014.

[Tamar *et al.*, 2016] Aviv Tamar, Yi Wu, Garrett Thomas, Sergey Levine, and Pieter Abbeel. Value iteration networks. In *Advances in Neural Information Processing Systems*, pages 2154–2162, 2016.

[Todorov, 2011] Emanuel Todorov. A convex, smooth and invertible contact model for trajectory optimization. In *Int. Conf. on Robotics and Automation (ICRA'11)*, pages 1071–1076. IEEE, 2011.

[Toussaint and Lopes, 2017] Marc Toussaint and Manuel 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.

[Toussaint *et al.*, 2018] Marc Toussaint, Kelsey R Allen, Kevin A Smith, and Josh B Tenenbaum. Differentiable physics and stable modes for tool-use and manipulation planning. In *Proc. of Robotics: Science and Systems (R:SS 2018)*, 2018.

[Toussaint, 2015] Marc Toussaint. 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.

[Toussaint, 2017] Marc Toussaint. A tutorial on Newton methods for constrained trajectory optimization and relations to SLAM, Gaussian Process smoothing, optimal control, and probabilistic inference. In Jean-Paul Laumond, editor, *Geometric and Numerical Foundations of Movements*. Springer, 2017.

[Wimpenny *et al.*, 2009] Joanna H Wimpenny, Alex AS Weir, Lisa Clayton, Christian Rutz, and Alex Kacelnik. Cognitive processes associated with sequential tool use in new caledonian crows. *PLoS One*, 4:e6471, 2009.