Direct Loss Minimization Inverse Optimal Control

Andreas Doerr*, Nathan Ratliff*†, Jeannette Bohg†, Marc Toussaint* and Stefan Schaal†‡
*Machine Learning & Robotics Lab, University Stuttgart, Germany
†Max Planck Institute, Autonomous Motion Departement, Tuebingen, Germany
‡University of Southern California, Computational Learning and Motor Control Lab, Los Angeles, CA, USA
andreasdoerr@gmx.net, {ratliffn, jbohg, sschaal}@tue.mpg.de, marc.toussaint@ipvs.uni-stuttgart.de, sschaal@usc.edu

***Abstract*—Inverse Optimal Control (IOC) has strongly impacted the systems engineering process, enabling automated planner tuning through straightforward and intuitive demonstration. The most successful and established applications, though, have been in lower dimensional problems such as navigation planning where exact optimal planning or control is feasible. In higher dimensional systems, such as humanoid robots, research has made substantial progress toward generalizing the ideas to model free or locally optimal settings, but these systems are complicated to the point where demonstration itself can be difficult. Typically, real-world applications are restricted to at best noisy or even partial or incomplete demonstrations that prove cumbersome in existing frameworks. This work derives a very flexible method of IOC based on a form of Structured Prediction known as Direct Loss Minimization. The resulting algorithm is essentially Policy Search on a reward function that rewards similarity to demonstrated behavior (using Covariance Matrix Adaptation (CMA) in our experiments). Our framework blurs the distinction between IOC, other forms of Imitation Learning, and Reinforcement Learning, enabling us to derive simple, versatile, and practical algorithms that blend imitation and reinforcement signals into a unified framework. Our experiments analyze various aspects of its performance and demonstrate its efficacy on conveying preferences for motion shaping and combined reach and grasp quality optimization.**

I. Introduction

Implementing versatile and generalizable robot behavior can take hundreds if not thousands of person-hours. Typical systems integrate sensor processing, state estimation, multiple layers of planning, low level control, and even reactive behaviors to induce successful and generalizable actions. With that complexity comes huge parameter spaces that take experts typically days of tuning to get right. Still, performance may be suboptimal, and any change to the system requires additional calibration. Much of how experts tweak systems remains an art, but recent machine learning advances have led to some very powerful learning from demonstration tools that have significantly simplified the process [1, 33].

One method of choice for learning from demonstration, especially in practical high-performance applications, is Inverse Optimal Control (IOC) [18, 34, 3, 13]. Since most planning-based systems are designed to transform sensor readings into cost functions interpretable by planners, they already produce generalizable behavior by design. Tapping into this architecture automates the time-consuming tweaking process needed to make this mapping from features to costs reliable. Where applicable, IOC has become an invaluable tool for real-world applications [25]. IOC has been very successful in lower dimensional problems, but interestingly has struggled to make the leap to higher dimensional systems. Two factors contribute to making high-dimensional systems hard.

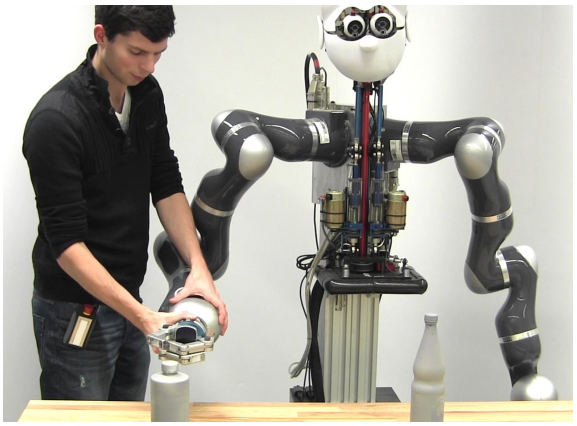


Fig. 1: Learning from demonstrated behavior. A humanoid robot is used as a running example in the experimental section of this work for learning of motion policies.

First, the Optimal Control problem itself, is intractable in most high-dimensional, especially continuous, domains (e.g. as found in humanoids). Recent advances in motion optimization, though, have made significant progress on that problem. Algorithms like CHOMP [21], STOMP [10], iTOMP [4], TrajOpt [24], KOMO [30], and RIEMO [22] have incrementally improved motion optimization to the point where now it is often a central tool for high-dimensional motion generation.

The second issue though, is more fundamental: it is very difficult to provide accurate, full policy demonstrations for high-dimensional systems. This problem is often overlooked or ignored in existing approaches (cf. Section III).

This work narrows the gap between Imitation Learning (specifically, Inverse Optimal Control) and Reinforcement Learning (Policy Search) by tracing the development of IOC through its connections to Structured Prediction and relating further advances in Structured Prediction back to the policy learning problem. What results is a unified algorithm that blends naturally between Inverse Optimal Control and Policy Search Reinforcement Learning enabling both learning from noisy partial demonstrations and optimization of high-level reward functions to join in tuning a high-performance optimal planner. The fundamental connection we draw, which

Section II discusses in detail, is that if we apply an advanced form of Structured Prediction, known as Direct Loss Minimization [15], to the IOC problem, what results is effectively a Policy Search algorithm that optimizes a reward

promoting similarity to expert demonstrations. This connection of Policy Search RL through Direct Loss Minimization to IOC suggests that the problem's difficulty results mostly from the shape of the reward landscape itself, and less from the problem formulation. Reward functions that reward similarity to expert demonstrations are naturally more discriminating than high-level success-oriented rewards. This work closes the gap between Reinforcement Learning and Imitation Learning by straightforwardly posing them both as blackbox optimization problems, letting the reward/loss functions distinguish between whether the problem is Inverse Optimal Control, Reinforcement Learning, or some combination of the two. Section V presents applications of this hybrid learning methodology using both noisy complete demonstrations and partial information to address learning motion styles.

Our high-level goal is to create tools to help engineers navigate the complicated high-dimensional parameter spaces that arise in sophisticated real-world systems.

II. Methodology

Inverse Optimal Control is strongly related to Structured Predition. Maximum Margin Planning [18], for instance, reduces the problem directly to Maximum Margin Structured Classification [28, 31], and Maximum Entropy IOC [34] develops a Bayesian framework strongly related to Conditional Random Fields [12]. These two bodies of literature have been driving significant algorithmic advances from both sides [20, 17, 32]. In many ways, we can view IOC explicitly as a form of Structured Prediction where the set of all policies is the structured set of labels and the underlying learning problem is to predict the correct policy given expert demonstrations. Advances in Structured Prediction usually lead to advances in IOC.

Structured Prediction has gone through a number of incarnations, but one prominent formalization of the problem is Maximum Margin Structured Classification (MMSC) [28], a generalization of the Support Vector Machine (SVM). Just as the hinge loss is a simple piecewise linear upper bound to the 0-1 binary loss function in an SVM, the generalized "hinge-loss" for MMSC is also a piecewise linear upper bound to the structured loss function of the Structured Prediction problem. For binary classification, directly optimizing the 0-1 loss is nearly impossible because of its discontinuity; the convexity of the hinge-loss upper bound is, therefore, critical. For much of the early development of Structured Prediction and MMSC, the same upper bound proxy requirement was generally assumed to be just as critical for the structured case.

This structured loss, for IOC, may be the squared loss between an arbitrary trajectory ξ and a demonstration ξ_i: $\mathcal{L}(\xi_i, \xi) = \mathcal{L}_i(\xi) = \frac{1}{2}\|\xi_i - \xi\|^2$ (i indexes the ith example). We would generally need to differentiate ξ as a function of our parameters $\boldsymbol{w}$, but MMSC and related methods instead proposed a convex upper bound (the structured hinge loss)

$$\mathcal{F}(\boldsymbol{w}) = \sum_{i=1}^{N} \left(\boldsymbol{w}^T \boldsymbol{f}_i(\xi) - \min_{\xi \in \Xi} (\boldsymbol{w}^T \boldsymbol{f}_i(\xi_i) - \mathcal{L}_i(\xi)) \right) + \frac{\lambda}{2}\|\boldsymbol{w}\|^2,$$

where $\boldsymbol{w} \in \mathbb{R}^d$ is a weight vector for d features $\boldsymbol{f}(\xi_i, \xi) = \boldsymbol{f}_i(\xi)$, and $\lambda \in \mathbb{R}_+$. A simple subgradient method for optimizing this objective suggests an update rule of the form

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta_t \sum_i (\boldsymbol{f}_i(\xi_i) - \boldsymbol{f}_i(\xi_i^*)), \tag{1}$$

where $\xi_i^* = \operatorname{argmin}_{\xi \in \Xi}(\boldsymbol{w}^T \boldsymbol{f}_i(\xi) - \mathcal{L}_i(\xi))$ (see [28] and [18] for details). This proxy objective and update rule have also been used in graphics (see [14]) under a margin-less "Perceptron" form [5] where the authors learned directly from motion capture data. In all cases (Structured Prediction, IOC, and graphics), formulating the convex proxy and resulting subgradient update rule was critical since it was unclear how to efficiently optimize the loss function directly.

But in 2010, [15] demonstrated that an algorithm very similar in nature to a subgradient approach to MMSC, but which directly estimated the gradient of the underlying structured loss function, not only worked, but often performed better than approaches leveraging this convex MMSC objective. The authors demonstrated empirically and theoretically that the shape of $\mathcal{L}(\xi_i, \xi)$, itself, was sufficiently structured to admit direct optimization. Parameterizing ξ by a weight vector $\boldsymbol{w}$ and problem context $\boldsymbol{\gamma}_i$, they showed that they could directly optimize $\psi(\boldsymbol{w}) = \sum_i \mathcal{L}(\xi_i, \xi(\boldsymbol{w}, \boldsymbol{\gamma}_i))$. Interestingly, the update rule that approximated that gradient bears a striking resemblance to that given in Equation 1

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \eta_t \sum_i \left(\boldsymbol{f}_i(\xi_i) - \boldsymbol{f}_i(\xi^*_{i\,\text{direct}}) \right), \tag{2}$$

where $\boldsymbol{f}_i(\xi^*_{i\,\text{direct}}) = \operatorname{argmin}_{\xi \in \Xi} \left(\boldsymbol{w}^T \boldsymbol{f}_i(\xi) + \epsilon \mathcal{L}_i(\xi) \right)$. In the parlance of policy learning, this update rule compares the features see by the example policy ξ_i to those see by a policy coaxed downhill slightly by increasing the cost of high-loss states.

Turning back to the policy learning problem these observations suggest that a strong form of IOC would be to directly optimize this loss function $\psi(\boldsymbol{w})$. The above direct loss gradient estimate gives one update rule that's particularly applicable to general Structured Prediction, but on the policy side, this sort of direct optimization problem has been studied for a while and there are a number of well-understood competing approaches. Denoting $R_i(\xi) = -\mathcal{L}(\xi_i, \xi)$ to relate negative losses to rewards and using $\boldsymbol{\gamma}$ to denote the dependence of the loss on current problem context, we see this problem is effectively a form of Policy Search on a deterministic policy:

$$\psi(\boldsymbol{w}) = \sum_{i=1}^{N} R_i(\xi(\boldsymbol{w}, \boldsymbol{\gamma}_i)). \tag{3}$$

In other words, an effective form of Inverse Optimal Control is simply Policy Search using a reward function that rewards similarity to demonstrated behavior. Here $\xi(\boldsymbol{w}, \boldsymbol{\gamma}_i)$ denotes the policy or trajectory produced under parametrization $\boldsymbol{w}$ for problem context (environment) $\boldsymbol{\gamma}_i$.

Given these observations, the question now is largely experimental. This paper discusses the implications of this framework for IOC and presents a series of experiments analyzing

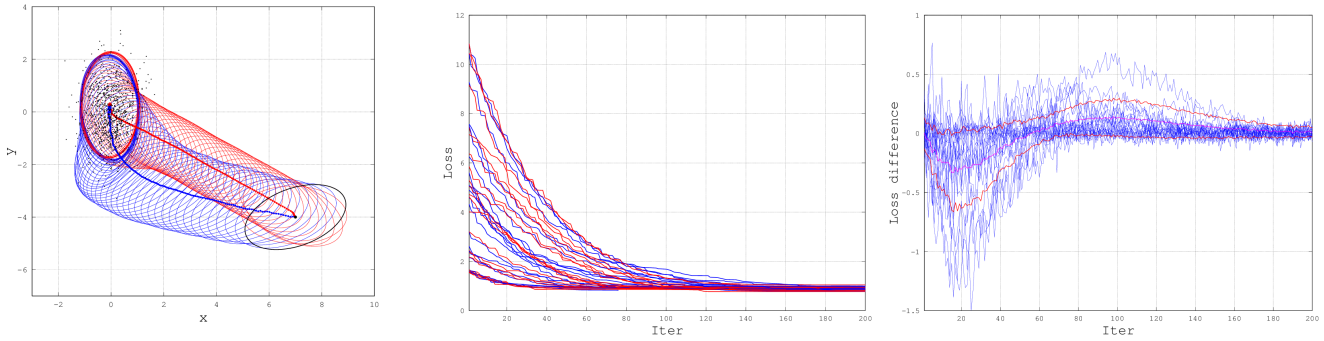


Fig. 2: **Left:** The hypothesis progressing toward the optimal policy. Black oval is the initial hypothesis, and the black points are expert data samples. Path differences in the hypothesis progressions largely stem from differences in objective. **Middle:** Comparison of loss progressions over time with different starting points and expert samples. Note that the loss progressions are typically grouped in pairs; the primary variation results from changing starting location and expert samples. **Right:** Plot showing the progression of loss-differences between the two algorithms across each learning trial. Positive values indicate Maximum Likelihood is smaller and vice versa. The mean and one-standard-deviation is shown as magenta and red lines, respectively. Notice, that Direct Loss Minimization tends to outperform Maximum Likelihood early on, but Maximum Likelihood converges slightly faster on average. In all figures, the Maximum Likelihood progression is depicted in red and the Direct Loss Minimization progression is depicted in blue.

the performance of this methodology using a generic policy search framework based on the black box Covariance Matrix Adaptation (CMA) optimizer. CMA is quickly becoming a goto tool for complex nonlinear policy search problems [6] for its combined efficacy, simplicity, and strong theoretical connections to a very successful form of policy search known as PI^2 [26] which has been shown to perform well in real-world robotics applications [29].

A. Similarities Between Direct Loss Minimization and Traditional IOC

Direct Loss Minimization IOC at first glance seems substantially different from traditional IOC methods. In this section, we demonstrate that both the update equations and the resulting behavior of the algorithms can be quite similar. Consider a simple policy of the form $p(\xi; \boldsymbol{w}) \propto e^{-C(\xi, \boldsymbol{w})}$, where $C(\xi, \boldsymbol{w})$ is a family of cost functions defined over possible trajectories ξ with parameters $\boldsymbol{w} \in \mathbb{R}^d$. Given samples $\mathcal{D} = \{\xi_i\}_{i=1}^N$ from an expert distribution $p_{\mathcal{T}}(\xi)$, one traditional form of IOC simply fits the distribution using Maximum Likelihood [34]. The gradient of the log-likelihood $\mathcal{F}_{\text{ml}}(\boldsymbol{w})$ is

$$\boldsymbol{g}_{\text{ml}} = -\text{E}_{\mathcal{D}}[\nabla_{\boldsymbol{w}} C(\xi, \boldsymbol{w})] + \text{E}_{p_{\boldsymbol{w}}}[\nabla_{\boldsymbol{w}} C(\xi, \boldsymbol{w})], \qquad (4)$$

where $\text{E}_{\mathcal{D}}[\cdot]$ denotes the empirical expectation taken over the data and $\text{E}_{p_{\boldsymbol{w}}}[\cdot]$ denotes the expectation taken with respect to the current policy $p_{\boldsymbol{w}}(\boldsymbol{x}) = p(\boldsymbol{x}; \boldsymbol{w})$.

Now consider running Policy Search as gradient descent on the expected reward

$$\mathcal{F}_{\text{ps}}(\boldsymbol{w}) = \int \mathcal{R}(\xi) p(\xi; \boldsymbol{w}) d\xi. \qquad (5)$$

The gradient of $\mathcal{F}_{\text{ps}}(\boldsymbol{w})$ is

$$\boldsymbol{g}_{\text{ps}} = -\text{E}_{p_{\boldsymbol{w}}^{\mathcal{R}}}[\nabla_{\boldsymbol{w}} C(\xi, \boldsymbol{w})] + \text{E}_{p_{\boldsymbol{w}}}[\nabla_{\boldsymbol{w}} C(\xi, \boldsymbol{w})], \qquad (6)$$

where $p_{\boldsymbol{w}}^{\mathcal{R}}(\xi) \propto \mathcal{R}(\xi) e^{-C(\xi; \boldsymbol{w})}$ denotes the reward-weighted policy distribution. The Supplementary Material derives both of these gradients. Here, we just note that both $\boldsymbol{g}_{\text{ml}}$ and $\boldsymbol{g}_{\text{ps}}$ are very similar in structure. They differ only in the first term, which defines how the algorithm makes policy modifications toward "better" policies. Maximum Likelihood forms this term directly using the demonstrations, while Direct Loss Minimization forms the term effectively using the gradient of the chosen loss function.

Figure 2 compares optimizations under these two algorithms for a simple two-dimensional Gaussian policy distribution parameterized by the mean (x, y) and an angle θ giving the angle of the covariance's principle axis. The variances along the major and minor axes are fixed at 1 and .3, respectively (see the Supplementary Material for details). The expert distribution is a zero-centered Gaussian with primary axis aligned with the y-axis and the same major and minor variances. Rather than maximizing an expected reward, we minimize an expected loss, which is effectively the same but more natural when viewing the algorithm as Direct Loss Minimization IOC. For this simple example, we use the following loss function $\mathcal{L}(\boldsymbol{x}) = \frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_e)^T \boldsymbol{\Sigma}_e^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_e)$, where $\boldsymbol{\mu}_e$ and $\boldsymbol{\Sigma}_e$ are the empirical mean and covariance of the expert data, respectively. Both algorithms used the same inverse linear step size sequence ($\eta_t = \frac{\eta}{t+c}$ for $\eta, c \in \mathbb{R}_+$ and t is the iteration index) with normalized gradients in order to emphasize the relative quality of the calculated gradient information.

The plots compare the behavior of the two algorithms for this problem. Both optimize well and differences in the hypothesis path taken by the two algorithms result largely from differences in the shape of the objective and how that interacts with gradient-based methods. Notice that the maximum likelihood method tends to (needlessly) align the principle axis of the Gaussian in the direction of steepest objective decrease up front. This optimization artifact slows its progress early on allowing direct loss minimization to optimize faster during initial stages. The logarithm in the log-likelihood objective, though, induces better conditioning in the final stages of optimization allowing maximum likelihood to ultimately catch up and, in some cases, even overtake direct loss minimization. Overall convergences are not strongly affected by these differences, but these progressions demonstrate the added flexibility under direct loss methodologies to shape the optimization behavior through careful choice of the loss function.

B. Combined Imitation and Reinforcement

Framing IOC as blackbox optimization of a relevant loss brings with it the flexibility of generic optimization. This section discusses a number of ways this flexibility manifests for policy learning as described below.

a) Blending with Reinforcement learning: At its core, this methodology suggests that there is no algorithmic difference between Policy Search and Imitation Learning of this form (given matching assumptions on policy parametrization). The difference is primarily in how the reward (or loss) function is constructed. Generally, a natural notion of reward measuring the quality and success of a policy is available (e.g. the robot achieves its goal without expending too much energy). Imitation rewards can easily be layered on top of that to help coax the learner into a good regions of the policy space before they are slowly downweighted and ultimately removed. Effectively, imitation learning in this context acts as an interesting regularization term to help the learner navigate the tricky reinforcement reward landscape.

b) Partial and noisy demonstrations: Past IOC approaches required full trajectory demonstrations from an expert policy. Generating these demonstrations is difficult for systems having many degrees of freedom. This lead to either built in policy assumptions or noisy demonstrations. Manually moving a robot arm in gravity compensation is awkward and leads to jerky, inexact motions. Kinesthetic teaching of the end-effector motion in contrast requires Inverse Kinematics (IK) to determine the remaining joint states. The resulting Null space motions certainly are not optimal with respect to velocities, accelerations, and the underlying dynamics. In some cases, it makes sense, as described in Section V-E, to demonstrate only the final configuration which encodes only posture and grasp point information. In all of these cases, the expert may supply important information, but it is incomplete at best, and usually very noisy. Even so, it is straightforward still to write down a loss function measuring how well the behavior of a policy matches the demonstrated behavior. These imitation loss terms again supplement the high-level reward to encourage similarity to the demonstrations.

c) Life-long learning: Many Policy Search methods are amenable to online execution, which enables life-long learning in robots. Modeling imitation learning as loss-optimization using generic Policy Search optimizers makes any of these tools available to imitation as well. Experts can advise a robot system simply by adding temporary imitation terms to their reward function to coax the learner toward a good solution. This paper does not explore this avenue explicitly, but this property has consistently been a strong motivating factor for this work.

d) Practical tool for tuning and calibrating the system: At the end of the day, regardless of theoretical connections between IOC and Structured Prediction, we want a tool that practically enables engineers to better navigate the high-dimensional space of parameters that comes with complex behavior generation systems such as the motion optimizer described here. Modeling imitation as loss-shaping of black box Policy Search is a simple and effective methodology for simplifying the design and implementation of sophisticated robot behaviors. The grasping experiments in Section V-E give examples of how these tools address the recalibration problem that arises when the underlying motion optimization policy representation is modified.

C. Motion Optimization

We use a Motion Optimization toolbox called RIEmannian Motion Optimization (RIEMO) [22] as our underlying optimization framework. RIEMO solves constrained motion optimization problems of the form

$$\begin{aligned} \min_{\xi} \sum_{t=1}^{T} c_t(\boldsymbol{q}_t, \dot{\boldsymbol{q}}_t, \ddot{\boldsymbol{q}}_t) \\ \text{s.t. } \boldsymbol{g}_t(\boldsymbol{q}_t, \dot{\boldsymbol{q}}_t, \ddot{\boldsymbol{q}}_t) \leq \mathbf{0} \quad \text{for all } t = 1, \ldots, T \\ \boldsymbol{h}_t(\boldsymbol{q}_t, \dot{\boldsymbol{q}}_t, \ddot{\boldsymbol{q}}_t) = \mathbf{0} \quad \text{for all } t = 1, \ldots, T, \end{aligned} \tag{7}$$

where $\xi = (\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_T)$ with $\boldsymbol{q}_t \in \mathbb{R}^n$ denoting the tth configuration along the trajectory. c_t, $\boldsymbol{g}_t$, and $\boldsymbol{h}_t$ are the tth objective term, inequality constraint function(s), and equality constraint function(s), respectively. RIEMO exploits the 2nd order Markov structure of this network of terms to efficiently implement an Augmented Lagrangian method with an inner loop unconstrained Newton optimizer leveraging Gauss-Newton like approximations that account for first-order geometry of differentiable (e.g. kinematic) maps. Typically, each c_t, $\boldsymbol{g}_t$, and $\boldsymbol{h}_t$ is a weighted collection of sub-terms. These weights enter into the policy parametrization, as Section IV describes in detail.

III. Related work

IOC for high-dimensional continuous spaces has appeared a handful of times in the literature. For instance, [11] develop an algorithm for IOC designed around a motion optimizer named STOMP based on a Path Integral reformulation of Stochastic Optimal Control named PI^2. They demonstrate learning objective functions for both IK problems and reaching behaviors. Additionally, [3] presented an extension of the Maximum Entropy IOC ideas from [34] to model-free learning in continuous spaces which they demonstrate on ball-in-cup and race trace driving problems; [13] emphasizes the difficulty of Nonlinear Optimal Control in their work, developing methods that work well with local optimizers.

Those papers make significant progress toward getting IOC to work in higher-dimensional continuous spaces, but they generally assume demonstrations are available and they are careful to ensure the demonstrations they use are good enough. This paper emphasizes the practical difficulties of data generation and introduces a pragmatic framework for leveraging even partial or noisy demonstrations by blending imitation with reinforcement learning methodologies.

Tuning planners (specifically, Rapidly-exploring Randomized Trees (RRTs)) using Policy Search Reinforcement Learning has previously been explored by [35] as a principled approach to navigating the high-dimensional space of parameters

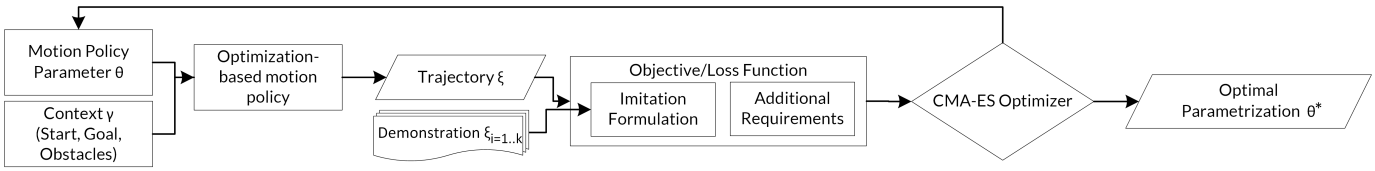


Fig. 3: Components of the high-level optimization setup for direct loss minimization policy search. The blackbox optimizer CMA-ES minimizes a loss function which can express the proximity to demonstrated behavior but also additional higher level goals. The policy is represented as an optimization program based on a given objective function parametrization and the test context.

associated with complex heuristics. Here we emphasize the theoretical connection between this approach and Structured Prediction forms of IOC, as well as the broader scope of the framework as discussed in Section II-B.

IV. Experimental Setup

The experiments presented in this work focus on motions executed on a humanoid robot arm (cf. Figure 1). Barrett manipulator hands are attached to KUKA lightweight robot arms summing up to 11 DoF per arm (7 arm joints and 4 hand joints).

The framework is built on top of the RIEMO based motion policy as described in Section II-C. Up to $d = 25$ parametrized cost features have been utilized in the following experiments depending on the motion type's characteristics. The cost terms trade off dynamics, kinematic smoothness, and posture criteria, while the constraints prevent joint limit violations and obstacle penetration, while enforcing goal success. Specifically, we use the following terms:

- Configuration derivatives penalties. $c(\dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) = \alpha_1 \|\dot{\boldsymbol{q}}\|^2 + \alpha_2 \|\ddot{\boldsymbol{q}}\|^2$.
- Task space derivatives penalties. $c(\boldsymbol{x}, \dot{\boldsymbol{x}}) = \frac{1}{2} \| \frac{d}{dt} \phi(\boldsymbol{x}) \|^2$, where $\phi : \mathbb{R}^3 \to \mathbb{R}^n$ is a mapping of key points on the robot's body to a higher-dimensional workspace representation.
- Joint limit proximity penalties. $c_i(q_i) = (\max\{0, q_i - (q_{\max} - \epsilon), (q_{\min} + \epsilon) - q_i\})^2$, where $\epsilon > 0$ is a joint limit margin.
- Posture potentials. $c(\boldsymbol{q}) = \frac{1}{2} \|\boldsymbol{q} - \boldsymbol{q}_{\text{default}}\|^2$ Bias towards a natural robot position $\boldsymbol{q}_{\text{default}}$.
- Orientation potentials. $c(\boldsymbol{q}) = \frac{1}{2} \|\mathbf{n} - \psi(\boldsymbol{x})\|^2$ Quadratic potential pulling orientations of robot parts into a given direction. E.g. horizontal hand posture.

And we also use the following constraints

- Joint limit constraints. Explicit constraints to prevent joint limit violations in the final trajectory.
- Obstacle constraints. Analytic representation of surrounding obstacles to prevent end-effector or other key points from penetrating the obstacle surfaces by the use of distance margins.
- Goal constraint. Reaching the goal is enforced as a zero distance constraint on the goal proximity distance function. This strategy generalize the goal set ideas described in [7].

In the presence of obstacles, we additionally exposed the radial and angular scaling factors of the cylindrical workspace Riemannian metric in the presence of obstacles. See [22] for details of the basic parametrization. Note that since we build upon the CMA optimizer, we don not require this parametrization to be linear.

We denote the motion policy parameter vector by $\boldsymbol{\theta} \in \mathbb{R}^d$; Figure 3 depicts the entire training framework.

Our pipeline is predicated on a definition of the imitation reward/loss function of Equation 3. We experimented with a collection of intuitive objectives, and found a simple metric measuring the average deviation from the demonstration between key points on the robot's arm and hand to work the best:

$$\mathcal{L}(\xi_i, \xi) = \sum_{t=1}^{T} \sum_{k=1}^{K} \left\| \phi_k(\boldsymbol{q}_i^{(t)}) - \phi_k(\boldsymbol{q}^{(t)}) \right\|^2, \qquad (8)$$

where $\phi_k(\boldsymbol{q})$ denotes the k^{th} key point on the body for configuration $\boldsymbol{q}$, $\xi_i = (\boldsymbol{q}_i^{(1)}, \ldots, \boldsymbol{q}_i^{T})$ is the demonstrated trajectory, and $\xi = (\boldsymbol{q}^{(1)}, \ldots, \boldsymbol{q}^{(T)})$ is the trajectory under consideration. Arbitrary additional higher level requirements can be expressed in the loss function since only the function evaluation for a given generated trajectory is necessary and no gradient information has to be computed.

We use a generic implementation of Covariance Matrix Adaptation (CMA-ES) [8] as our Policy Search blackbox optimizer. The learning parameters of the CMA-ES algorithm have been set according to Hansen's default implementation. Only the initial solution point $\boldsymbol{\theta}_0$ and the scalar initial step size σ_0 have to be chosen in a problem-specific way. To achieve the optimal search performance, all parameters (i.e. all search dimensions) should have similar sensitivities. Since the ranges of the raw parameters incorporated in the given cost terms cover more than six magnitudes, the dimensions have been rescaled to $[0, 10]$ to achieve a reasonable parameter encoding. The normalization factors have been determined experimentally by manually weighting the cost features in order to achieve a smooth motion in unobstructed space. The initial solution point is chosen uniformely from the assumed parameter range $(0, 10)$. In [27], CMA-ES has been shown to successfully adapt the individual exploration noise of each dimension to explore up to several magnitudes from the initial starting point and CMA-ES is furthermore capable of handling sensitivity differences between dimensions in the range of some magnitudes. The initial covariance matrix shape is therefore given by the identity matrix $\Sigma_0 = \mathbf{I}_d$. In accordance to [8], the initial exploration step size is set to a fifth of the domain size which proved to be reasonable in all our experiments.

To get around the need to actively constrain the search space to the parameter space since most cost terms are only defined on positive parameter values, the objective function is evaluated on the absolute parameter, i.e. $c_t(|\boldsymbol{\theta}|)$. Other methods to constrain the search space by a lower and/or upper bound (e.g. quadratic or exponential replacement) have been proposed in [8] but performed poorly for the problems investigated in this work.

V. Experimental Evaluation

We will present several experiments to analyze the behavior of Direct Loss Minimization IOC using CMA-based Policy Search. Section V-A addresses learning motion shaping behaviors from full trajectory demonstrations while Section V-B addresses learning related behaviors from partial demonstrations. Section V-C examines imposing underlying biasing reward terms to resolve ambiguity in the partial demonstrations, whereas Section V-D demonstrates the robustness of the CMA optimizer on these problems, and Section V-E explores applications to combined reaching and grasping optimization.

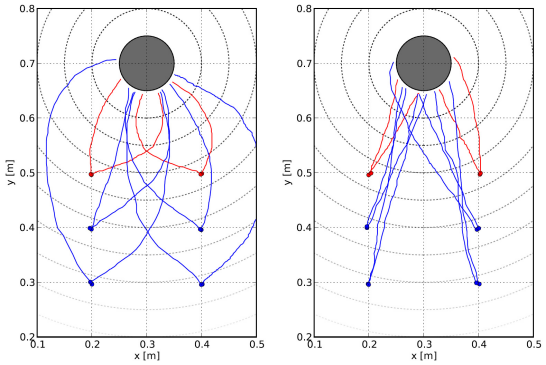
A. Noisy Joint State Demonstrations

For this experiment, we recorded demonstrations by kinesthetic teaching having the robot's arm in gravity compensation mode. The resulting motions are downsampled equidistantly in joint space to match the horizon length of the motion optimizer. Our demonstrations were grouped into the following behaviors: straight end-effector motions, sliding on a tabletop in front of the robot, motions that primarily use a subset of the robot's degrees of freedom (e.g. shoulder rotations or forearm motions), reaching motions maintaining a certain hand orientation (e.g. carrying a glass of water). Some trajectories produced by the learned optimal policies are visualized in Figure 4 as they are executed on the actual robot. The learned characteristics of each motion type are clearly reflected in each parameter vector. Predominant joints have significantly lower velocity penalty weights while straight motions in task space are expressed in the ratio of task to joint space velocity penalties. The characteristic parts of these motions have been successfully learned to imitate the demonstrated behavior and also to generalize the observed behavior to untrained scenarios.

B. Sketched End-Effector Demonstrations

Full joint state demonstrations as seen in the last experiments tend to be imprecise and jerky even in situations where kinesthetic teaching is available. Especially behaviors that require certain particular velocity or acceleration profiles are hard to demonstrate. The following experiment will explore the ability of this approach to learn from intentionally incomplete demonstrations that focus on some special aspect of a motion. In the given example, the behavior of the end-effector when approaching and operating around obstacles is therefore learned from sketched demonstration of the desired end-effector trajectories as shown in Figures 5(a) and 5(b). For both motion types, 24 sketched end-effector demonstrations are given, partitioned into a training and a test set to cross-validate the resulting optimal policy. The optimal policy is learned from all training demonstrations by using the average distance between the end-effector's trace and the sketched trajectory as loss function.

For this scenario we used the non-Euclidean representation of the workspace geometry introduced in [22] combining a cylindrical coordinate system with an ambient Euclidean system. The weight on the cylindrical system increases with proximity to the obstacle. The intensity of the concentric rings in Figure 5 depicts this Riemannian workspace geometry. The relative weights on radial and angular velocity components in the cylindrical coordinate system of the obstacle are exposed as parameters such that the Direct Loss Minimization IOC learning system can shape the relative tendency of the optimizer to circle the obstacle vs heading directly toward the goal point. The average fitness for both learned motion policies and one initial random policy evaluated for all 24 demonstrated scenarios is shown in Figure 6 (A = motion type I, C = motion type II). The next section explains the results in greater detail along with results from imposing additional objectives.



(a) Motion Type I (b) Motion Type II

Fig. 5: The desired behavior is partially demonstrated by sketching the desired end-effector path. A subset of the demonstrations given for training (red) and test (blue), is visualized for two types of motions together with one solution trajectory as produced by the learned policy (green).

C. Imposing Additional Objectives

Especially in case of incomplete demonstrations and to resolve redundancies, additional objectives can be introduced. These objectives can affect arbitrary aspects of the motion since the chosen black box optimizer requires only the evaluation of the objective function. The loss function of reinforcement learning, previously formulated to imitate a demonstration, is therefore augmented by an additional term. Here, we introduce the average height of the elbow as a new loss contribution in order to maintain a 'natural', low elbow position. The same scenario as presented in Experiment V-B is used here. For all possible combinations of the two motion types and two loss functions, a total of four policies (A,B,C,D) is learned whose fitness is visualized in Figure 6. Additionally, the fitness of an initial random policy (orange) is shown for comparison. For each policy, the contribution of the end-effector distance (yellow) between demonstration and solution trajectory as well as the contribution of the elbow loss term (red) to the overall loss function is plotted. The resulting elbow loss is also displayed for the policies that have been learned using only the pure imitation loss function which does not consider the elbow height (policies A and C, the elbow loss is shown in light red). This experiment demonstrates several of the capabilities we discussed in Section II:

a) Imitation of Observed Behavior: A policy learned on a given set of training demonstrations is able to reproduce the observed behavior. On the same set of problems, the optimal

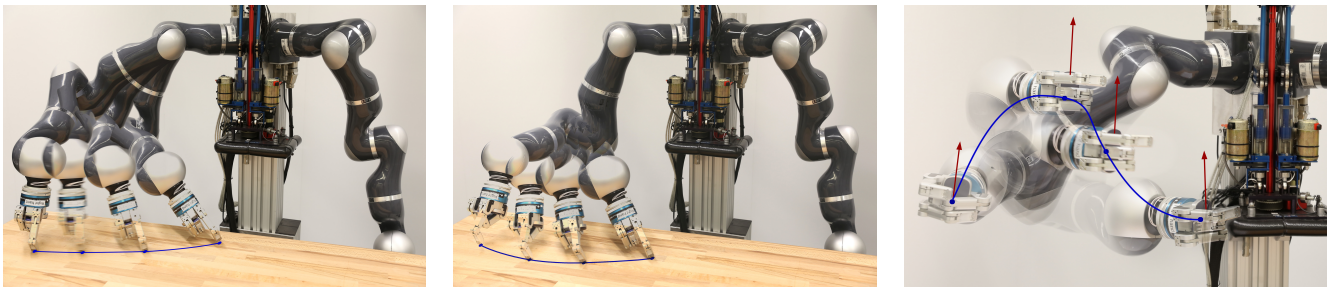


Fig. 4: Motion policies learned to favor specific degrees of freedom. Pointing motions, sliding from one point on the table to another are visualized as executed on the actual robot. On the left image shoulder motions are favored whilst on the image in the middle forearm motions are. In the right image, a policy is learned which maintains a horizontal hand alignment.

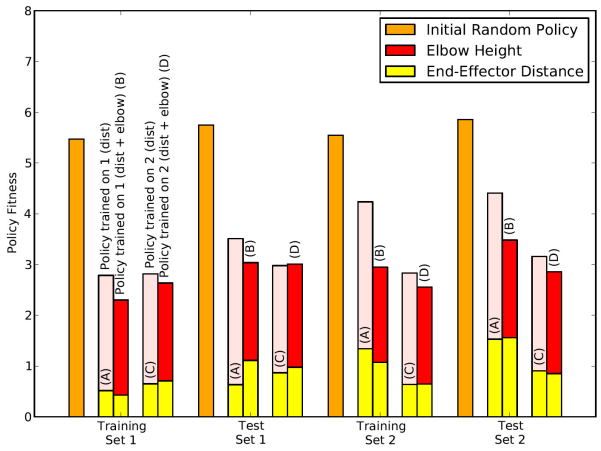


Fig. 6: Cross-validation of the fitness of policies learned on training sets for two motion types: pointing to a cylindrical obstacle with high (set 1) and low (set 2) obstacle avoidance. For both motion types, one policy minimizing the imitation loss and another minimizing the combined imitation and elbow height loss have been learned, summing up to 4 different policies (A,B,C,D).

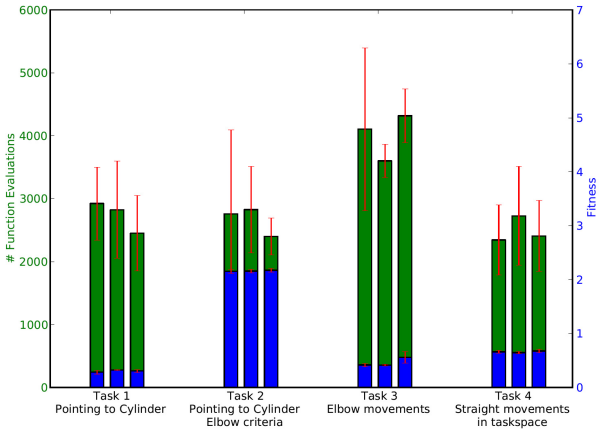


Fig. 7: Effects of domain size and problem dimensionality on the optimization speed and quality. The average number of necessary function evaluations and the resulting fitness is evaluated on four motion problems. Three different initial configurations (step size and initial parametrization) are shown for each task.

policy clearly outperforms the initial random policy but also the policy which learned a different type of motion. This is true for both the policy learned using the pure imitation loss function (A and C) and the policy learned using the combined combined loss function (B and D).

b) Generalization to Similar Scenarios: Concerning the fitness of the learned policies on the corresponding test problems, the policy learned for this type of motion clearly performs better than the random policy and the policies learned for the other motion type. This is especially the case for the policies learned for motion type 2 which perform significantly better on their own test set compared to the performance of the motion type 1 policies on this test set (cf. policies B and D clearly outperform policies A and C on their own training set 2).

c) Optimizing Additional Objectives: The policies learned using the augmented loss functions can be directly compared to the ones which have been learned using the pure imitation loss functions (e.g. policy A and B). The policies learned based on the augmented loss function clearly minimize the combined loss. In particular, the contribution of the elbow loss is significantly reduced for policies learned from the combined loss formulation. This result is not only visible in the fitness of the policies on the training set but it also generalizes to the policies' application to the according test set.

D. Effects of Optimizer Settings

The effects of the domain size on the optimizer's performance (e.g. in terms of convergence and best fitness) have been evaluated on the experiments as presented above. The range of the domain has been varied for all those experiments from $[0, 1]$ and $[0, 5]$ up to $[0, 10]$. The initial guess $\boldsymbol{\theta}_0$ has been chosen randomly within the domain and the initial step size has been set to $\sigma_0 = (\max - \min)/5$.

The results of evaluating the performance for those experiments are shown in Figure 7. The optimal policy is computed based on the three different domain ranges. For each task, the bars represent from left to right the results of the experiment carried out for increasing domain ranges. The results are averages over at least 4 independent runs. The error bars display the 1σ standard deviation. Neither the number of function evaluations (shown in green) nor the resulting fitness (shown in blue) is significantly influenced by the initial domain size. CMA turns out to be robust against slightly inappropriate initial configurations. The automatic adaptation of exploration direction and magnitude is capable to find a similar solution in all evaluated cases. Similar results have been achieved by employing the BiPop version of CMA-ES [2], striving for more global optimization. No evident improvement of the global fitness could be achieved. In case of a reasonable parameter encoding, the pure CMA-ES setup is sufficient to explore the required parameter space.

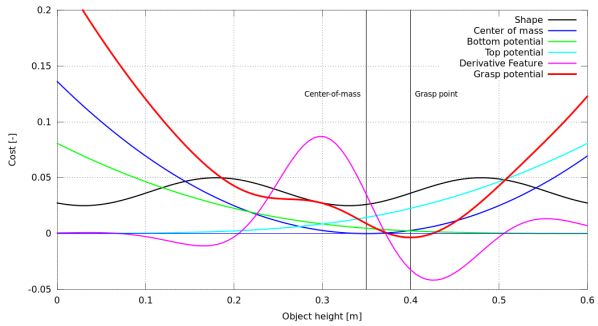


Fig. 8: Contribution of cost feature terms to the grasp metric along the test object's main axis. The object's shape (black, only one half of the 0.6 m object is shown) is visualized together with top (cyan) and bottom (green) potential, center of mass potential (blue) and the combined surface width feature (magenta). The learned grasp potential (red) is shown for center-of-mass=0.35 m and the grasp point demonstrated to be at 0.40 m.

E. Combined Reach and Grasp Motion

Grasping problems are usually divided into separate approach planning and grasp planning problems [16]. Imitation approaches to the latter have successfully learned grasp or grasp point metrics from demonstration [23, 9, 19], but these two stage strategies make simultaneously optimizing both parts difficult. Traditionally, IOC methods for learning such combined approach and grasp motions required full approach and grasp demonstrations, and generating such demonstrations is hard. This section describes some experiments leveraging Direct Loss Minimization IOC to learn full approach motions from partial data specifying only the desired grasp point. We experiment both with learning the combined objective, and recalibrating that function to work with new approach motion parametrization.

Our motion parametrization matches that of the above experiments for all terms except for the terminal potential. For this experiment, we assume a simple grasping setup in which an object, having a given rotationally symmetric shape, is placed upright in front of the robot as shown in Figure 9. We constructed a terminal potential consisting of a term penalizing quadratic deviations from the object's vertical axis along (x, y) and a term $c_g(z; \boldsymbol{\theta}_g)$ measuring the grasp point cost along the object's height $z \in \mathbb{R}$. Figure 8 depicts the grasp point features along with one of the learned combined grasp point metrics. Our features included biases away from the top and bottom of the object, a quadratic potential pulling toward the center-of-mass, and three surface width features measuring the width value $f_w(z)$, first-derivative $f_{w'}(z)$, and second-derivative $f_{w''}(z)$. We denote the vector of all three of these features as $\boldsymbol{f}_w(z)$. The three surface width features are combined nonlinearly as $c_s(z; \boldsymbol{\alpha}) = (e^{\boldsymbol{\alpha}^T \boldsymbol{f}_w(z)} - 1)e^{-\frac{\beta}{2}\|z - z_c\|^2}$, where z_c is the center of mass and $\beta > 0$ is a fixed scaling parameter. CMA is able to optimize the problem despite the nonlinearity in the three parameters $\boldsymbol{\alpha} \in \mathbb{R}^3$. This parameterization builds off an explicit surface model for illustration purposes, but features designed from perceptual inputs can be naturally integrated into this model as well.

Three sample objects are defined and demonstrations of reasonable grasping points are given embodying a grasp concept favors grasp points close to the object's bulge in proximity to the center-of-mass. The learned policy is able to integrate the correct grasp point into the approaching motions and reproduces the demonstrated grasp within some millimeters (object height: 60 cm). Given the grasp demonstrations, we can use our Direct Loss Minimization IOC framework to easily recalibrate the combined approach and grasp motion optimization in case of changes to the system. In our experiments, the recalibrated systems consistently produced similar imitation loss accuracies. Figure 9 depicts the resulting grasping motions on our robot platform. Two of the demonstrations, specified by the center-of-mass position (red point) and demonstrated grasp position (yellow point), are visualized. The learned policy executes the correct grasp in case of the initial system setup (yellow and red trajectory) as well as in case of the system disturbed by an upward potential (green trajectory).

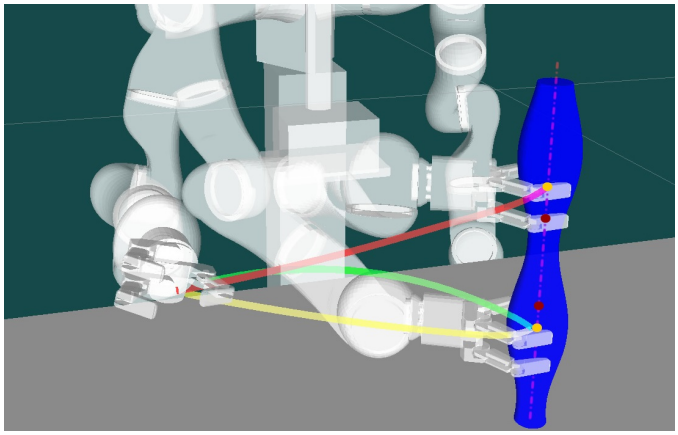


Fig. 9: Two grasp postures and corresponding approach trajectories (red and yellow) generated by the learned grasp motion policy are visualized for two configurations of the blue grasp object. For two of three demonstrations, the center-of-mass (red point) and the demonstrated grasp point (yellow point) are shown. After modifications to the system (additional upward potential), the newly learned grasp policy compensates changes in the approach trajectory automatically and matches the demonstrated grasp point (green trajectory).

VI. Conclusion

This work capitalizes on the strong connection between Inverse Optimal Control and Structured Prediction to pose a new form of IOC modeled after Direct Loss Minimization Structured Prediction. The resulting behavior learning framework, Direct Loss Minimization IOC, may be viewed as Policy Search Reinforcement Learning using rewards that promote behavior that best mimics an expert. Unified algorithms that blend between the two extremes of pure reinforcement and pure imitation arise naturally within this framework, and handling partial or noisy demonstrations is straightforward. We demonstrate our learning strategy using generic CMA policy search algorithms on a collection of problems ranging from motion shaping to combined motion optimization and grasp metric calibration. Direct Loss Minimization IOC overcomes many of the practical limitations that can make traditional IOC methodologies cumbersome in high-dimensional continuous spaces. Direct Loss Minimization IOC is a very flexible and practical tool for designing, tuning, updating, and refining complicated planning systems with increasingly large and unintuitive parameter space.

REFERENCES

[1] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.

[2] A. Auger and N. Hansen. A restart CMA evolution strategy with increasing population size. In B. McKay et al., editors, *The 2005 IEEE International Congress on Evolutionary Computation (CEC'05)*, volume 2, pages 1769–1776, 2005.

[3] A. Boularias, J. Kober, and J. Peters. Relative entropy inverse reinforcement learning. In *Proceedings of Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2011)*, 2011. URL http://jmlr.csail.mit.edu/proceedings/papers/v15/boularias11a/boularias11a.pdf.

[4] Jia Pan Chonhyon Park and Dinesh Manocha. ITOMP: Incremental trajectory optimization for real-time replanning in dynamic environments. In *International Conference on Automated Planning and Scheduling (ICAPS)*, 2012.

[5] Michael Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*. Association for Computational Linguistics, 2002.

[6] Mike Depinet, Patrick MacAlpine, and Peter Stone. Keyframe sampling, optimization, and behavior integration: Towards long-distance kicking in the robocup 3d simulation league. In H. Levent Akin, Reinaldo A. C. Bianchi, Subramanian Ramamoorthy, and Komei Sugiura, editors, *RoboCup-2014: Robot Soccer World Cup XVIII*, Lecture Notes in Artificial Intelligence. Springer Verlag, Berlin, 2015.

[7] Anca Dragan, Nathan Ratliff, and Siddhartha Srinivasa. Manipulation planning with goal sets using constrained trajectory optimization. In *2011 IEEE International Conference on Robotics and Automation*, May 2011.

[8] Nikolaus Hansen. The cma evolution strategy: A tutorial. *Vu le*, 29, 2005.

[9] A. Herzog, P. Pastor, M. Kalakrishnan, L. Righetti, T. Asfour, and S. Schaal. Template-based learning of grasp selection. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 2379–2384, 2012. URL http://www-clmc.usc.edu/publications/H/herzog-ICRA2012.pdf.

[10] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal. STOMP: Stochastic trajectory optimization for motion planning. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, 2011. URL http://www-clmc.usc.edu/publications/K/kalakrishnan-ICRA2011.pdf.

[11] M. Kalakrishnan, P. Pastor, L. Righetti, and S. Schaal. Learning objective functions for manipulation. In *IEEE International Conference on Robotics and Automation*, 2013. URL /publications/K/kalakrishnan-ICRA2013.pdf.

[12] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289, 2001.

[13] Sergey Levine and Vladlen Koltun. Continuous inverse optimal control with locally optimal examples. In *ICML 2012*, 2012.

[14] C. Karen Liu, Aaron Hertzmann, and Zoran Popovic. Learning physics-based motion style with nonlinear inverse optimization. *ACM Trans. Graph*, 24:1071–1081, 2005.

[15] D. McAllester, T. Hazan, and J. Keshet. Direct loss minimization for structured prediction. In *Neural Information and Processing Systems (NIPS)*, 2010.

[16] Andrew Miller and Peter K. Allen. Graspit!: A Versatile Simulator for Robotic Grasping. *IEEE Robotics and Automation Magazine*, 11(4):110–122, 2004.

[17] Daniel Munoz, J. Andrew Bagnell, Nicolas Vandapel, and Martial Hebert. Contextual classification with functional max-margin markov networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[18] Nathan Ratliff, J. Andrew (Drew) Bagnell, and Martin Zinkevich. Maximum margin planning. In *International Conference on Machine Learning*, July 2006.

[19] Nathan Ratliff, J. Andrew (Drew) Bagnell, and Siddhartha Srinivasa. Imitation learning for locomotion and manipulation. In *IEEE-RAS International Conference on Humanoid Robots*, November 2007.

[20] Nathan Ratliff, J. Andrew (Drew) Bagnell, and Martin Zinkevich. (online) subgradient methods for structured prediction. In *Eleventh International Conference on Artificial Intelligence and Statistics (AIStats)*, March 2007.

[21] Nathan Ratliff, Matthew Zucker, J. Andrew (Drew) Bagnell, and Siddhartha Srinivasa. CHOMP: Gradient optimization techniques for efficient motion planning. In *IEEE International Conference on Robotics and Automation (ICRA)*, May 2009.

[22] Nathan Ratliff, Marc Toussaint, and Stefan Schaal. Riemannian motion optimization. Technical report, Max Planck Institute for Intelligent Systems, 2015.

[23] A. Saxena, J. Driemeyer, J. Kearns, and A.Y. Ng. Robotic grasping of novel objects. In *Neural Information Processing Systems*, 2006.

[24] John D. Schulman, Jonathan Ho, Alex Lee, Ibrahim Awwal, Henry Bradlow, and Pieter Abbeel. Finding locally optimal, collision-free trajectories with sequential convex optimization. In *In the proceedings of Robotics: Science and Systems (RSS)*, 2013.

[25] David Silver, J. Andrew (Drew) Bagnell, and Anthony (Tony) Stentz. Learning from demonstration for autonomous navigation in complex unstructured terrain. *International Journal of Robotics Research*, 29(12):1565 – 1592, October 2010.

[26] Freek Stulp and Olivier Sigaud. Path integral policy improvement with covariance matrix adaptation. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*, 2012.

[27] Freek Stulp and Olivier Sigaud. Path integral policy improvement with covariance matrix adaptation. *arXiv preprint arXiv:1206.4621*, 2012.

[28] B. Taskar, V. Chatalbashev, D. Koller, and C. Guestrin. Learning structured prediction models: A large margin approach. In *Twenty Second International Conference on Machine Learning (ICML05)*, Bonn, Germany, August 2005.

[29] Evangelos A. Theodorou, Jonas Buchli, and Daniel Lee. A generalized path integral control approach to reinforcement learning. *The Journal of Machine Learning Research*, pages 3137–3181, 2010.

[30] Marc Toussaint. Newton methods for k-order Markov constrained motion problems. *CoRR*, abs/1407.0414, 2014. URL http://arxiv.org/abs/1407.0414.

[31] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *International Conference on Machine Learning (ICML)*, pages 104–112, 2004.

[32] Paul Vernaza and Daniel D Lee. Scalable real-time object recognition and segmentation via cascaded, discriminative markov random fields. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 3102–3107. IEEE, 2010.

[33] Shao Zhifei and Er Meng Joo. A survey of inverse reinforcement learning techniques. *International Journal of Intelligent Computing and Cybernetics*, 5(3):293–311, 2012.

[34] Brian D. Ziebart, Andrew Maas, J. Andrew (Drew) Bagnell, and Anind Dey. Maximum entropy inverse reinforcement learning. In *Proceeding of AAAI 2008*, July 2008.

[35] Matthew Zucker, James Kuffner, and J. Andrew (Drew) Bagnell. Adaptive workspace biasing for sampling based planners. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, May 2008.