Inverse Optimal Control for Humanoid Locomotion

Taesung Park and Sergey Levine
Department of Computer Science, Stanford University
E-mail: taesung@graphics.stanford.edu, svlevine@stanford.edu

***Abstract*—In this paper, we present a method for learning the reward function for humanoid locomotion from motion-captured demonstrations of human running. We show how an approximate, local inverse optimal control algorithm can be used to learn the reward function for this high dimensional domain, and demonstrate how trajectory optimization can then be used to recreate dynamic, naturalistic running behaviors in new environments. Results are presented in simulation on a 29-DoF humanoid model, and include running on flat ground, rough terrain, and under strong lateral perturbation.**

I. Introduction

Bipedal locomotion is an essential ingredient for enabling humanoid robots to navigate around their environment [6]. Legged locomotion is also an important problem in computer graphics, where the goal is to animate lifelike virtual characters [11]. Previous successful locomotion methods often focus on hand-crafting a set of control laws that, often with the aid of optimization, can produce robust, efficient, and naturalistic locomotion behaviors [11, 6]. However, such methods rely on meticulous engineering, and are limited in their capacity to generalize to new situations, because they capture the essence of the locomotion behavior in terms of its mechanisms (such as balance feedback) rather than its objectives. If we can capture the objectives that underlie bipedal locomotion, we could create robust and dynamic locomotion behaviors in new situations simply by optimizing the trajectory with respect to these objectives.

Inverse optimal control (IOC), also called inverse reinforcement learning, is a tool can be used to learn an unknown reward function from example demonstrations of the desired behavior. This reward function then serves as a portable representation of the behavior, allowing it to be generalized to new domains [7, 1]. However, the extreme computational complexity of IOC has limited previous applications to lower-dimensional tasks, such as navigation of road networks [13] or footstep planning on 2D height maps [2]. In this paper, we show that a recently developed approximate IOC algorithm can overcome these computational challenges and learn reward functions directly from demonstrations on high-dimensional humanoid models. The reward is learned from motion-captured human demonstrations, and captures not only salient features of the desired behavior, but also the fine details that are necessary to produce fluid, dynamic locomotion that resembles the original human demonstration.

We employ an approximate local IOC algorithm that only examines the dynamics and reward around the example trajectory [3], making it possible to scale the method to high-dimensional humanoid tasks. The final behavior is produced with trajectory optimization, using a differential dynamic programming (DDP) algorithm that is analogous to the local IOC method. Besides demonstrating that the learned reward function allows us to reproduce smooth, realistic, and highly dynamic running, we also show that it provides a generalizable representation of the behavior that allows it to respond to perturbations and changes in the terrain. We also present preliminary results that suggest that not only does the learned reward allow us to create a more dynamic and realistic running behavior, but that it also shapes the objective and allows the local DDP optimizer to more effectively avoid poor local optima that might be present in a naïve, hand-engineered objective. In this way, IOC serves to not only learn a reward that explains the task, but also to shape the reward, making subsequent forward optimization easier.

The main contribution of this paper is a method for learning locomotion behaviors directly from human demonstration using inverse optimal control. To our knowledge, this is the first attempt to learn reward functions for humanoid behaviors with IOC directly from motion capture of real humans. We show that a local, approximate IOC algorithm can scale to the high dimensional space of humanoid poses, and present a set of reward features that are suitable for learning running behaviors. We demonstrate that the learned reward can be used to reproduce running behaviors with trajectory optimization and present preliminary results for generalization to perturbations and rough terrain. We also show that the learned reward is better suited for continuous local trajectory optimization than a naïve, hand-engineered objective.

II. Inverse Optimal Control

We model the locomotion behavior as a deterministic, fixed-horizon control task with continuous states $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)^{\mathrm{T}}$, continuous actions $\mathbf{u} = (\mathbf{u}_1, \ldots, \mathbf{u}_T)^{\mathrm{T}}$, and discrete time. The state vector contains the position of the pelvis, the angle at each joint, and all of the corresponding velocities, while the actions correspond to the torques at each joint. Such tasks are characterized by a dynamics function $f(\mathbf{x}_t, \mathbf{u}_t) = \mathbf{x}_{t+1}$, as well as a reward function $r(\mathbf{x}_t, \mathbf{u}_t)$. Given the initial state $\mathbf{x}_1$, the optimal actions are given by

$$\mathbf{u} = \arg\max_{\mathbf{u}} \sum_t r(\mathbf{x}_t, \mathbf{u}_t).$$

IOC aims to find a reward function r under which the optimal actions match the expert's demonstrations, given by $\mathcal{D} = \{(\mathbf{x}_0^{(1)}, \mathbf{u}^{(1)}), \ldots, (\mathbf{x}_0^{(n)}, \mathbf{u}^{(n)})\}$. The algorithm is presented with reward features $\mathbf{f} : (\mathbf{x}_t, \mathbf{u}_t) \rightarrow \mathbb{R}$ that can be used

to represent the unknown reward as

$$r(\mathbf{x}_t, \mathbf{u}_t) = \sum_j \theta_j \mathbf{f}_j(\mathbf{x}_t, \mathbf{u}_t).$$

We employ a probabilistic model for the expert's behavior that represents the probability of a sequence of actions as being proportional to the exponential of their total reward [13]:[1]

$$P(\mathbf{u}|\mathbf{x}_1) = \frac{1}{Z} \exp\left(\sum_t r(\mathbf{x}_t, \mathbf{u}_t)\right), \tag{1}$$

where Z is the partition function. In this model, the expert follows a stochastic policy that becomes more deterministic when the stakes are high, and more random when all choices have similar value. The reward can then be learned by maximizing the log probability of the examples with respect to the reward weights θ.

Computing the partition function Z typically requires an iterative algorithm similar to value iteration, which scales exponentially with state-space dimensionality [13]. Levine and Koltun proposed to instead approximate the reward log likelihood by using the Laplace approximation with locally linearized dynamics and locally quadratic rewards [3]. In this approach, the log of the path probability in Equation 1 is approximated as

$$\mathcal{L} = \frac{1}{2}\mathbf{g}^{\mathrm{T}}\mathbf{H}^{-1}\mathbf{g} + \frac{1}{2}\log|-\mathbf{H}| - \frac{d_{\mathbf{u}}}{2}\log 2\pi, \tag{2}$$

where $\mathbf{g}$ is the gradient of the total reward with respect to the actions, and $\mathbf{H}$ is the Hessian. This objective, as well as its gradients with respect to θ, can be computed efficiently with one of several dynamic programming algorithms, including an algorithm that is virtually identical to the linear quadratic regulator (LQR) [3]. In the latter case, the objective can be derived by assuming linear dynamics and quadratic rewards, and iteratively computing the Q-function from the last time step to the first. In the case of linear dynamics and quadratic rewards, it is known that the value function updates in the maximum entropy model have the same form as in the standard model [12], leading to the following dynamic programming equations for the Q-function and value function:

$$\begin{aligned}
Q_{\mathbf{xx}t} &= r_{\mathbf{xx}t} + f_{\mathbf{x}t}^{\mathrm{T}} V_{\mathbf{xx}t+1} f_{\mathbf{x}t} & Q_{\mathbf{x}t} &= r_{\mathbf{x}t} + f_{\mathbf{x}t}^{\mathrm{T}} V_{\mathbf{x}t+1} \\
Q_{\mathbf{uu}t} &= r_{\mathbf{uu}t} + f_{\mathbf{u}t}^{\mathrm{T}} V_{\mathbf{xx}t+1} f_{\mathbf{u}t} & Q_{\mathbf{u}t} &= r_{\mathbf{u}t} + f_{\mathbf{u}t}^{\mathrm{T}} V_{\mathbf{x}t+1} \\
Q_{\mathbf{ux}t} &= r_{\mathbf{ux}t} + f_{\mathbf{u}t}^{\mathrm{T}} V_{\mathbf{xx}t+1} f_{\mathbf{x}t} & V_{\mathbf{x}t} &= Q_{\mathbf{x}t} - Q_{\mathbf{ux}t}^{\mathrm{T}} Q_{\mathbf{uu}t}^{-1} Q_{\mathbf{u}} \\
V_{\mathbf{xx}t} &= Q_{\mathbf{xx}t} - Q_{\mathbf{ux}t}^{\mathrm{T}} Q_{\mathbf{uu}t}^{-1} Q_{\mathbf{ux}}.
\end{aligned} \tag{3}$$

The quantities $r_{\star\star}$ indicate derivatives of the reward function with respect to $\mathbf{x}$ and $\mathbf{u}$, while $f_{\star\star}$ indicate derivatives of the dynamics. Under the maximum entropy model [12], the probability of the demonstrated action at time t is given by

$$P(\mathbf{u}_t|\mathbf{x}_t) = \exp\left(Q(\mathbf{x}_t, \mathbf{u}_t) - V(\mathbf{x}_t)\right), \tag{4}$$

[1]In the case of multiple example trajectories, their probabilities are simply multiplied together to obtain $P(\mathcal{D})$.

where the value function is given by

$$V(\mathbf{x}_t) = \log \int \exp\left(Q(\mathbf{x}_t, \tilde{\mathbf{u}}_t)\right) \tilde{\mathbf{u}}_t. \tag{5}$$

Assuming that the Taylor expansion is done around the example state and action $\mathbf{x}_t$ and $\mathbf{u}_t$, Equation 4 can be evaluated by using the second order approximation for Q from Equation 3. The integral in Equation 5 can then be evaluated in closed form, since the exponential of a quadratic is a Gaussian. This produces the following action probability at each time step:

$$\log P(\mathbf{u}_t|\mathbf{x}_t) = \frac{1}{2} Q_{\mathbf{u}t} Q_{\mathbf{uu}t}^{-1} Q_{\mathbf{u}t} + \frac{1}{2}\log|-Q_{\mathbf{uu}t}|.$$

The approximate objective in Equation 2 can then be evaluated by adding up these log likelihoods over all time steps, and gradients can also be computed by using the chain rule and maintaining the derivative of each $Q_{\star\star}$ quantity with respect to each reward weight. Further details on this algorithm are provided in previous work [3].

III. Trajectory Optimization

Once we compute the reward weights θ that maximize the approximate log likelihood for the demonstrated locomotion behavior, we can reconstruct the behavior under new conditions (such as rough terrain) by using a variant of differential dynamic programming called iterative LQR [5]. This algorithm repeatedly reestimates the optimal linear feedback policy under linear dynamics and quadratic rewards, expanded around the current trajectory, and then follows this policy to create a new trajectory. The dynamic programming equations for this algorithm are identical to the IOC dynamic programming step in Equation 3, and the linear feedback policy is given by

$$\mathbf{u}'_t = \mathbf{u}_t + \mathbf{k}_t + \mathbf{K}_t(\mathbf{x}'_t - \mathbf{x}_t),$$

where $\mathbf{x}_t$ and $\mathbf{u}_t$ are the state and action along the previous trajectory, $\mathbf{x}'_t$ and $\mathbf{u}'_t$ are the state and action along the new trajectory, and $\mathbf{k}$ and $\mathbf{K}$ are given by

$$\mathbf{k}_t = -Q_{\mathbf{uu}t}^{-1} Q_{\mathbf{u}t} \qquad \mathbf{K}_t = -Q_{\mathbf{uu}t}^{-1} Q_{\mathbf{ux}t}.$$

In order to ensure that the total reward along the trajectory improves at each iteration, we follow Tassa et al. [8] and employ a linesearch on $\mathbf{k}$.

Trajectory optimization in the presence of contacts is known to be quite difficult, since the contacts introduce discontinuities in the system dynamics. To alleviate this problem, we also use a softened contact model described by Tassa et al. [8], as implemented in the MuJoCo simulation engine [9].

IV. Learning Locomotion Rewards

To apply the IOC algorithm described in Section II to humanoid locomotion, we must first obtain an example of the desired locomotion behavior. To this end, we use joint angles recorded with motion capture from a real human run. These joint angles are then tracked by using DDP with a simple reward that minimizes quadratic deviation from the target angles. It should be noted that while this simple reward is sufficient to obtain torques that realize the demonstrated run,

it is often not sufficient to generalize to new situations (such as strong perturbations), as shown in Section V.

In order to learn a reward function that captures a portable and generalizable representation of the desired locomotion behavior, we must provide the algorithm with a set of features that is sufficient for representing the behavior, but not so specific that they do not generalize to new environments. For example, the quadratic deviation reward used to track the motion capture could also be used as a reward feature, but would be a poor choice, since it would not improve generalization. In this section, we describe the features we created for learning humanoid locomotion. While this choice of features is specific to locomotion, the general principles described in this section may also be informative when seeking to construct appropriate features for new behaviors.

A. Torque Minimization

Minimization of joint torques is a common regularizer in reward functions for dynamic humanoid behaviors. We therefore employ a learned quadratic penalty on joint torques at each time step. We found it useful to learn a seperate penalty weight on the torque at each joint, in order to allow the reward to express a preference for some joints over others. Recent work has shown that direct total torque minimization is a poor explanation for real human locomotion [10]. The per-joint penalty allows our model to capture more fine-grained joint preferences that may be caused by the dynamics of the underlying muscles or idiosyncracies of individual gaits. The joint torque feature for joint i has the following form:

$$\mathbf{f}_i(\mathbf{x}_t, \mathbf{u}_t) = -\frac{1}{2}u_{ti}^2,$$

where u_{ti} is the i^{th} component of $\mathbf{u}_t$.

B. Root Position and Velocity

In order to make forward progress and maintain balance, we use features that captures the quadratic deviation of the root (pelvis) position and velocity from a setpoint. Specifically, we use a single feature for the medial component of the root velocity, a single feature for the height of the root, and an additional feature for the position of the root along the transverse axis, in order to control lateral deviations. In all cases, the feature has the form

$$\mathbf{f}_j(\mathbf{x}_t, \mathbf{u}_t) = \frac{1}{2}(x_{tj} - x_j^\star)^2, \tag{6}$$

where $x_j^\star$ is the desired setpoint. This setpoint is set to the mean value of x_{tj} in the example sequence. In the future, it would be straightforward to also learn x_{tj} as part of the IOC optimization, by employing both a linear and a quadratic feature.

C. Joint Angle Regularization

In principle, the root position and velocity, together with torque minimization, sufficiently constrain the task to specify a forward movement with minimal effort. However, practical local trajectory optimization techniques typically fail to create a plausible locomotion behavior from such abstract objectives, as shown in the following section. To further shape the reward and capture the unique style of the demonstrated gait, we also employ a set of joint angle features that control the degree to which each joint is allowed to deviate from a setpoint. The form of these features is identical to the root position and velocity features in Equation 6, except the index j refers to particular joint angles. The setpoints $x_j^\star$ are set to zero for all joints except the knees, corresponding to an upright standing pose. The knees are set to 0.5 radians, to encourage flexion. As with the root position and velocity, it would be straightforward to learn the target angles in the future using a set of linear and quadratic features.

D. Periodic Foot Motion

A crucial component of bipedal locomotion is the periodic, alternating motion of the feet. We capture this aspect of the behavior by learning Cartesian attractor positions for the feet in the sagittal plane. This is accomplished by using a set of linear and quadratic features for the medial and longitudal position of each foot. By learning the weights on the linear and quadratic features, we effectively learn a target position and weight for a quadratic deviation. To create periodic motion of the feet, a different set of targets is learned for each quarter of the gait cycle, using four equal-sized intervals. This results in a total of 32 features: 4 intervals, 2 feet, 2 axes, and 2 features each (linear and quadratic).

It should be noted that although these features can be interpreted as specifying a target position for the feet, they do not act as footstep constraints. Instead, they act as attractors for the feet, and the learned setpoints are often located several meters above the ground, and thus are never reached even in the example motion, as discussed in the following section. In future work, it would be interesting to see if the phase and frequency of the gait could also be learned, to avoid the need for specifying four equal-length intervals.

E. Arm Swing Features

To produce naturalistic arm swinging motion, we use an additional set of features that expresses the quadratic deviation of the shoulder and elbow from the reference example at each time step. Because the arm motion does not need to change drastically to adapt to new situations, we did not find that this feature degrades the generalization of our method. However, it would be worthwhile in the future to explore more abstract features for capturing periodic arm motion, perhaps in a similar spirit to the foot target features.

V. Results

Figure 1 shows plots of the original running demonstration, a running sequence generated with trajectory optimization from the learned reward function, and the result of running trajectory optimization with a naïve objective that only seeks to maintain balance and forward velocity while minimizing torques. The new trajectory was initialized with a simple standing policy to produce the initial trajectory for iterative LQR.

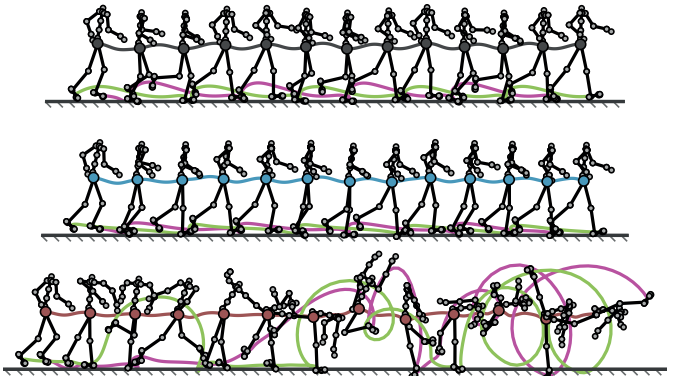

Fig. 1. Plots of running trajectories: original human motion capture (top), optimized result using our learned reward (middle), and the result with naïve baseline reward (bottom). Colored lines indicate pelvis and foot trajectories. The learned reward produces a run that is qualitatively similar to the original motion capture.

Although the reoptimized run is not identical, it exhibits many of the same overall properties, including a similar forward velocity and similar stride length. On the other hand, the naïve objective is insufficiently detailed to produce a plausible locomotion behavior, resulting in an unstable tumbling gait. The superior performance of the learned reward over this naïve baseline suggests that IOC has a shaping effect on the reward, effectively producing an objective that is more condusive to local optimization.

Table I shows the learned weights on each reward feature. From these reward weights, we can visualize the foot target features at each interval. These targets are visualized in Figure 2, by using an arrow to indicate the direction of the attractor, with length corresponding to the product of the distance to the target and the quadratic feature weight. Since the targets themselves were often several meters above the ground, they are not shown. Note that the foot targets do not act as "footstep constraints," but rather as attraction fields that guide the feet over the course of the gait.

To test generalization, we optimized a running sequence in the presence of a 5000 Newton, 100 ms lateral push, as well as on rough terrain. The results for the push are shown in Figure 3, and the rough terrain results are shown in Figure 4. In both tests, the results are compared to a simple baseline that minimizes the quadratic deviation from the original unperturbed, flat ground motion capture example. In the case of the lateral perturbation, we observe that the learned reward affords a more flexible and realistic recovery strategy, taking a large sideways step to avoid a fall. The baseline relies on razor-edge balance and remains too close to the example demonstration, producing an implausible recovery that uses very large torques to slide laterally while matching the example poses. The baseline fares better on the rough terrain, where both reward functions show reasonable generalization.

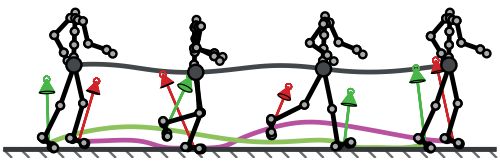

Fig. 2. Visualization of learned foot position attractors in each of the four phases, overlaid on poses from the original example.

Feature Class	Feature	Weight
joint torque penalty	waist rotation (x, y, z)	23.7,15.6,21.1
	left shoulder rotation (x, y, z)	24.9, 24.2, 24.8
	left elbow	24.8
	right shoulder rotation (x, y, z)	24.9, 24.4, 24.9
	right elbow	24.9
	left thigh rotation (x, y, z)	25.0, 22.0, 21.5
	left knee	21.8
	left ankle rotation (y, z)	21.2, 21.2
	right thigh rotation (x, y, z)	25.0, 22.1, 21.5
	right knee	21.6
	right ankle rotation (y, z)	21.2, 21.2
root motion	forward velocity	4.93
	lateral deviation	0.0141
	height deviation	1.81
joint angle	pelvis rotation (x, y, z)	20.6, 46.0, 702
	torso rotation (x, y, z)	178, 251, 1.89
	thigh rotation (x, y, z)	24.3, 0.712, 171
	left knee	1.25
	right knee	0.463
	left ankle rotation (y, z)	3.94, 435
	right ankle rotation (y, z)	1.83, 676
foot motion	phase 1	
	left foot (x^2, x)	69.4, -87.3
	left foot (z^2, z)	-10.2, 62.7
	right foot (x^2, x)	41.6, 104
	right foot (z^2, z)	3.84, 54.9
	phase 2	
	left foot (x^2, x)	396, 127
	left foot (z^2, z)	-6.48, 7.35
	right foot (x^2, x)	77.6, 21.2
	right foot (z^2, z)	34.3, 16.9
	phase 3	
	left foot (x^2, x)	60.5, 135
	left foot (z^2, z)	29.1, 69.0
	right foot (x^2, x)	48.1, -104
	right foot (z^2, z)	13.3, 87.7
	phase 4	
	left foot (x^2, x)	123, 51.3
	left foot (z^2, z)	1.92, -18.0
	right foot (x^2, x)	562, 127
	right foot (z^2, z)	-8.71, 11.9
arm swing	shoulder and elbow deviation	2180

TABLE I
WEIGHTS ASSIGNED TO EACH FEATURE BY THE LEARNING ALGORITHM.

These results indicate that the learned reward provides sufficient detail to reproduce the example behavior, while providing a sufficient abstract representation to allow for meaningful generalization. Further experiments are necessary to establish the amount of generalization that is possible, as well as the degree to which the learned reward outperforms simple quadratic tracking of the example motion.

VI. DISCUSSION

We presented a method for learning a generalizable reward function for humanoid locomotion that can be used to recreate a demonstrated locomotion behavior under new conditions, such as strong perturbations and rough terrain. We employed

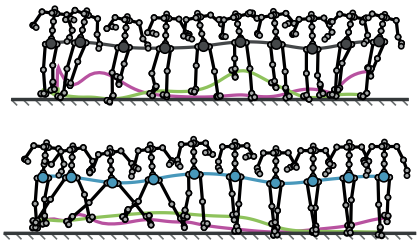

Fig. 3. Results under lateral perturbation: quadratic tracking of the unperturbed example (top) and the learned reward function (bottom), shown from the front. A 5000 Newton push to the right is applied at the first time step. When optimizing the learned reward, the trajectory exhibits a more plausible recovery strategy, which is abscent when rigidly tracking the example motion.

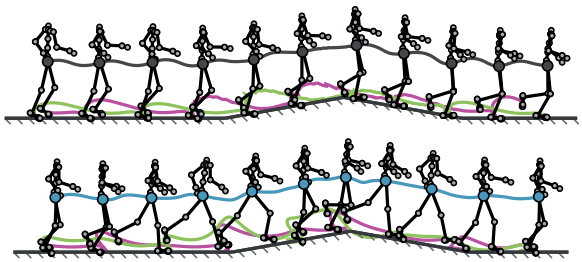

Fig. 4. Results on rough terrain: quadratic tracking of the flat ground example (top) and the learned reward function (bottom). The learned reward function allows plausible generalization to new terrains.

an approximate local IOC algorithm to extract the reward function from a human demonstration, and then used a local trajectory optimization algorithm to reconstruct the behavior under new conditions. Our preliminary results suggest that the IOC algorithm was able to learn a reward function that generalized successfully, and also suggest that the learned reward function was better suited for avoiding the local optima that often plague local trajectory optimization under more naïve objectives.

While our current trajectory optimization process is offline, it would be straightforward to use the learned reward function in a model predictive control (MPC) setting, where the new trajectory is constructed online in real time. Recent work has shown remarkable progress in applying MPC to high-dimensional systems with contacts [8], suggesting that such a method might be suitable for real-time control of humanoid locomotion in the future.

Although this paper only demonstrates inverse optimal control for locomotion, the methods employed are not specific to locomotion behaviors. An interesting avenue for future work would be to expand this approach to a range of other humanoid behaviors, especially highly dynamic behaviors, which are quite challenging to control using traditional methods. In order to adapt IOC to humanoid locomotion, we engineered a set of features that can be used to capture a flexible but generalizable representation of the reward function. A number of IOC methods have been proposed that alleviate the burden of feature engineering by learning nonlinear reward functions [4, 3], and applying such methods to humanoid behaviors would also be an exciting avenue for future work.

References

[1] Abbeel, P. and Ng, A. Apprenticeship learning via inverse reinforcement learning. In *ICML*, 2004.

[2] Kolter, J. Zico, Abbeel, Pieter, and Ng, Andrew Y. Hierarchical apprenticeship learning with application to quadruped locomotion. In *Advances in Neural Information Processing Systems (NIPS 20)*, 2007.

[3] Levine, S. and Koltun, V. Inverse optimal control with locally optimal examples. In *International Conference on Machine Learning (ICML)*, 2012.

[4] Levine, S., Popović, Z., and Koltun, V. Nonlinear inverse reinforcement learning with gaussian processes. In *Advances in Neural Information Processing Systems (NIPS 24)*, 2011.

[5] Li, W. and Todorov, E. Iterative linear quadratic regulator design for nonlinear biological movement systems. In *ICINCO (1)*, pp. 222–229, 2004.

[6] Manchester, Ian R., Mettin, Uwe, Iida, Fumiya, and Tedrake, Russ. Stable dynamic walking over uneven terrain. *International Journal Robotic Research*, 30(3): 265–279, 2011.

[7] Ng, Andrew Y. and Russell, Stuart J. Algorithms for inverse reinforcement learning. In *Proceedings of ICML*, 2000.

[8] Tassa, Y., Erez, T., and Todorov, E. Synthesis and stabilization of complex behaviors through online trajectory optimization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012.

[9] Todorov, E., Erez, T., and Tassa, Y. MuJoCo: A physics engine for model-based control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012.

[10] Wang, Jack M., Hamner, Samuel R., Delp, Scott L., and Koltun, Vladlen. Optimizing locomotion controllers using biologically-based actuators and objectives. *ACM Transactions on Graphics*, 31(4):25, 2012.

[11] Yin, K., Loken, K., and van de Panne, M. SIMBICON: simple biped locomotion control. *ACM Transactions Graphics*, 26(3), 2007.

[12] Ziebart, B. *Modeling purposeful adaptive behavior with the principle of maximum causal entropy*. PhD thesis, Carnegie Mellon University, 2010.

[13] Ziebart, Brian D., Maas, Andrew, Bagnell, J. Andrew, and Dey, Anind K. Maximum entropy inverse reinforcement learning. In *AAAI Conference on Artificial Intelligence*, 2008.