Entropic Risk Measure in Policy Search

David Nass, Boris Belousov, and Jan Peters

***Abstract*— With the increasing pace of automation, modern robotic systems need to act in stochastic, non-stationary, partially observable environments. A range of algorithms for finding parameterized policies that optimize for long-term average performance have been proposed in the past. However, the majority of the proposed approaches does not explicitly take into account the variability of the performance metric, which may lead to finding policies that although performing well on average, can perform spectacularly bad in a particular run or over a period of time. To address this shortcoming, we study an approach to policy optimization that explicitly takes into account higher order statistics of the reward function. In this paper, we extend policy gradient methods to include the entropic risk measure in the objective function and evaluate their performance in simulation experiments and on a real-robot task of learning a hitting motion in robot badminton.**

I. Introduction

Applying reinforcement learning (RL) to robotics is notoriously hard due to the curse of dimensionality [1]. Robots operate in continuous state-action spaces and visiting every state quickly becomes infeasible. Therefore, function approximation has become essential to limit the number of parameters that need to be learned. Policy search methods, that employ pre-structured parameterized policies to deal with continuous action spaces, have been successfully applied in robotics [2]. These methods include policy gradient [3], [4], natural policy gradient [5], expectation maximization (EM) policy search [6], [7], and information theoretic approaches [8].

A common feature of the aforementioned policy search methods is that they all aim to maximize the expected reward. Therefore, they do not take into account the variability and uncertainty of the performance measure. However, robotic systems need to act in stochastic, non-stationary, partially observable environments. To account for these challenges, the objective function should include an additional variance related criteria. This paper contributes to the field of reinforcement learning for robotics by extending the range of applicability of policy search methods to problems with risk-sensitive optimization criteria, where risk is given by the entropic risk measure [9].

II. Related work

Howard and Matheson [10] along with Jacobson [11] were the first to consider risk-sensitivity in optimal control both in discrete and continuous settings. Jacobson attempted to solve the linear exponential quadratic Gaussian problem that is analogous to the linear quadratic Gaussian but with an exponentially transformed quadratic cost. Later, connections between risk-sensitive optimal control and H_∞ theory [12] as well as differential games [13] were found.

In the recent years, there have been some advances in risk-sensitive policy search using policy gradients. In [14], a policy gradient algorithm was developed that accounted for the variance in the objective either through a penalty or as a constraint. The Conditional value at risk criterion was combined with policy gradients in [15] and [16]. In this paper, we study properties of policy gradient methods with the entropic risk measure in the objective. Employing this particular type of risk measure reveals tight links to popular policy search algorithms, such as reward weighted regression (RWR) [7] and relative entropy policy search (REPS) [8].

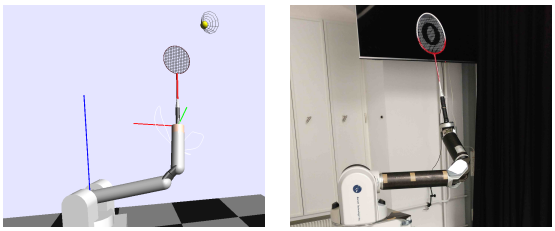
III. Background and Notation

In this section, we provide the necessary background information on risk measures and policy search methods.

A. Risk-Sensitive Objectives and Measures

The goal of an RL agent is to learn a mapping from states to actions that maximizes a performance measure [17]. In the episodic RL setting [2], the episode return R (also called reward) is a random variable, and the standard objective is to maximize the expected return. Our approach considers the case where the performance measure is risk-sensitive. That is, instead of optimizing for the long-term system performance we account for higher order moments of R by performing a risk-sensitive transformation of the return.

Risk-sensitivity can be described in terms of utility [18]. Namely, the utility function defines a transformation of the reward $U(R)$. The performance measure is then given by the expected utility $\mathbb{E}[U(R)]$. Clearly, depending on the choice of function $U(R)$, different behaviors will arise. A special



(a) Barrett WAM in a simulation environment

(b) Barrett WAM at IAS, TU Darmstadt

Fig. 1: Robot badminton evaluation task.

The authors are with Intelligent Autonomous Systems Lab, Department of Computer Science, Technische Universitt Darmstadt, Germany, surname@ias.tu-darmstadt.de

This project has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No 640554.

choice of the utility function, which attracted a lot of interest in various fields [18], is given by the exponential function

$$U(R) = \exp(-\gamma R) \tag{1}$$

with a risk-sensitivity factor $\gamma \in \mathbb{R}$. Depending on the sign of γ, the expected utility based on (1) needs to be either minimized or maximized [19], as explained in the following.

For a positive $\gamma > 0$, the expectation $\mathbb{E}[U(R)]$ is a convex decreasing function of R, therefore it needs to be minimized in order to maximize the reward. In this case, a certain expected reward with lower variance is favored, and the utility function is called *risk-averse*. On the other hand, when $\gamma < 0$, the expected utility needs to be maximized, which leads to favoring high-variance rewards. In this case the utility is called *risk-seeking*.

To avoid confusion, both cases $\gamma > 0$ and $\gamma < 0$ are often treated at once through the certainty-equivalent expectation [18] which always has to be maximized,

$$J_{\text{risk}}(R) = U^{-1}\,\mathbb{E}[U(R)] = -\frac{1}{\gamma}\log\mathbb{E}[\exp(-\gamma R)]. \tag{2}$$

When $\gamma > 0$, this quantity is called the *entropic risk measure* [9]. We slightly abuse terminology and refer to it as the entropic risk measure in the risk-seeking case $\gamma < 0$ too.

B. Policy Search

Consider the finite-horizon episodic RL setting [17]. At each time step t, an agent takes action $\mathbf{a}_t$ depending on the current state $\mathbf{s}_t$ by sampling it from a policy $\pi_{\boldsymbol{\theta}} = \pi(\mathbf{a}|\mathbf{s},\boldsymbol{\theta})$ parameterized by $\boldsymbol{\theta}$. Subsequently, the agent transitions into the next state $\mathbf{s}_{t+1}$ with probability $\mathscr{P}(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t)$. Real-valued reward $r(\mathbf{s}_t,\mathbf{a}_t)$ is collected at each time-step. The goal is to find a policy that maximizes the expected return

$$J_{\boldsymbol{\theta}} = \int p_{\boldsymbol{\theta}}(\tau) R(\tau)\,\mathrm{d}\tau \tag{3}$$

where $\tau = [\mathbf{s}_0,\mathbf{a}_0,\mathbf{s}_1,\mathbf{a}_1,\ldots]$ is a trajectory, $p_{\boldsymbol{\theta}}(\tau)$ is a distribution over trajectories induced by policy $\pi_{\boldsymbol{\theta}}$, and the cumulative reward is defined as $R(\tau) = \sum_{t=0}^{T} r(\mathbf{s}_t,\mathbf{a}_t)$.

Following common practice of policy search methods [2], we do not learn parameters $\boldsymbol{\theta}$ directly, but rather learn an upper-level policy $\pi_{\boldsymbol{\omega}}(\boldsymbol{\theta})$ that selects the parameters of the lower-level policy $\pi_{\boldsymbol{\theta}}$. Typically, the upper-level policy is modeled as a Gaussian distribution $\boldsymbol{\theta} \sim \mathscr{N}(\boldsymbol{\mu}_{\boldsymbol{\theta}},\Sigma_{\boldsymbol{\theta}})$. By defining the distribution over $\boldsymbol{\theta}$, we can explore directly in the parameter space. The resulting optimization problem for learning the upper-level policy is given by

$$\begin{aligned}\underset{\boldsymbol{\omega}}{\text{maximize}} \quad J(\boldsymbol{\omega}) &= \int \pi_{\boldsymbol{\omega}}(\boldsymbol{\theta}) \int p_{\boldsymbol{\theta}}(\tau) R(\tau)\,\mathrm{d}\tau\,\mathrm{d}\boldsymbol{\theta} \\ &= \int \pi_{\boldsymbol{\omega}}(\boldsymbol{\theta}) R(\boldsymbol{\theta})\,\mathrm{d}\boldsymbol{\theta} = \mathbb{E}_{\boldsymbol{\theta}\sim\pi_{\boldsymbol{\omega}}}[R(\boldsymbol{\theta})].\end{aligned} \tag{4}$$

Mind the common abuse of notation: $R(\boldsymbol{\theta})$ is not the same function as $R(\tau)$, although they are related. It is important to note that in the episodic scenario, the low-level policy return $R(\boldsymbol{\theta})$ is not limited to be the cumulative reward function but can be any function computed over rollouts [2].

IV. Risk-Sensitive Policy Search

Expected return (4) is the prime optimization objective for the majority of policy search methods [4], [5], [7], [8], [20]. To introduce risk-sensitivity into policy search, we propose to optimize the entropic risk-measure (2) instead. Rewriting it for the parameters $\boldsymbol{\omega}$ of the upper-level policy yields

$$J_\gamma(\boldsymbol{\omega}) = -\frac{1}{\gamma}\log\mathbb{E}_{\boldsymbol{\theta}\sim\pi_{\boldsymbol{\omega}}}[\exp(-\gamma R(\boldsymbol{\theta}))]. \tag{5}$$

In the following, policy search methods that maximize the objective (5) are described and studied. First, a risk-sensitive PG algorithm is derived in Sec. IV-A. After that, a connection to the REPS algorithm [8] is established in Sec. IV-C.

A. Risk-Sensitive Policy Gradient

As the name suggests [2], policy gradient methods aim to maximize the objective $J(\boldsymbol{\omega})$ by gradient ascent on the policy parameters

$$\boldsymbol{\omega}_{k+1} = \boldsymbol{\omega}_k + \alpha\nabla J(\boldsymbol{\omega}_k).$$

The likelihood ratio trick is commonly invoked to derive an estimate of the gradient. For the risk-sensitive objective (5), the likelihood ratio gradient yields

$$\nabla J_\gamma = \mathbb{E}_{\boldsymbol{\theta}\sim\pi_{\boldsymbol{\omega}}}\left[\nabla\log\pi_{\boldsymbol{\omega}}(\boldsymbol{\theta})\left\{-\frac{1}{\gamma}\mathrm{e}^{-\gamma(R(\boldsymbol{\theta})-\psi_\gamma(\pi_{\boldsymbol{\omega}}))}\right\}\right] \tag{6}$$

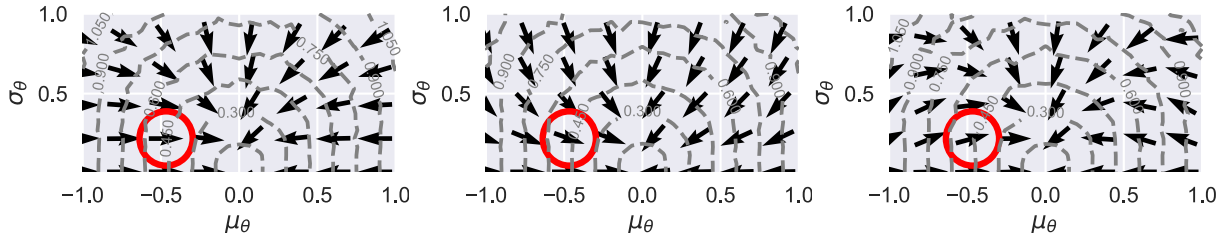
where $\psi_\gamma(\pi_{\boldsymbol{\omega}}) = -\gamma^{-1}\log\mathbb{E}_{\pi_{\boldsymbol{\omega}}}[\exp(-\gamma R)]$ is the log-partition function [21]. Expression (6) for the risk-sensitive gradient plays a fundamental role in our further discussion and we will often refer to it in the following.

The first point to make about (6) is the relation between the risk-sensitive policy gradient and the standard, risk-neutral one. Observe from (5) that the risk-sensitive objective $J_\gamma(\boldsymbol{\omega})$ becomes risk-neutral for $\gamma \to 0$. That is, by Taylor expansion, one can show that $J_\gamma \to J = \mathbb{E}[R]$. Surprisingly, however, *the gradient of the risk-sensitive objective does not correspond to the vanilla PG* $\nabla J = \mathbb{E}[\nabla\log\pi\cdot R]$ but instead to the PG with an average reward baseline

$$\nabla J_\gamma \xrightarrow[\gamma\to 0]{} \mathbb{E}[\nabla\log\pi\cdot(R-\mathbb{E}[R])]. \tag{7}$$

The log-partition function $\psi_\gamma(\pi_{\boldsymbol{\omega}})$ plays the role of the risk-sensitive baseline, since $\psi_\gamma \to \mathbb{E}[R]$ for $\gamma \to 0$. Therefore, risk-sensitive PG (6) automatically has lower variance compared to vanilla PG due to the presence of the baseline.

Regarding computational aspects, expectations in (6) can be estimated by averages $\mathbb{E}_{\boldsymbol{\theta}\sim\pi_{\boldsymbol{\omega}}}[f(\boldsymbol{\theta})] = N^{-1}\sum_{i=1}^{N} f(\boldsymbol{\theta}_i)$. Furthermore, we can view (6) as a risk-neutral PG for an exponentially transformed reward function given by the expression in curly braces in (6). Therefore, along with the multiplicative baseline $\psi_\gamma(\boldsymbol{\omega})$, the usual additive baseline can also be subtracted to further reduce variance. Moreover, standard algorithms, such as natural policy gradient (NPG) [5] and proximal policy optimization (PPO) [22], can be directly applied to optimize the risk-sensitive objective (5) thanks to the form (6) of the risk-sensitive policy gradient.



(a) PG, risk-neutral (b) PG, risk-averse (c) PG, risk-seeking

Fig. 2: Comparison of policy gradient directions on a simple linear system with risk-neutral, risk-averse, and risk-seeking objectives. Policy mean and variance are parameterized as $\mu_\theta = \omega_1$ and $\sigma_\theta = \exp(\omega_2)$. Interestingly, risk-seeking policy gradient (PG) may even point away from the optimum if it values uncertainty more than the expected return (see red circle).

B. Parameter Exploration with Risk-Sensitivity

Exploration noise and reward variability may come in conflict within the risk-sensitive optimization framework. In episodic policy search [2], exploration is achieved by sampling parameters θ of a lower-level policy π_θ from a stochastic upper-level policy $\pi_\omega = \mathcal{N}(\mu_\theta, \sigma_\theta^2)$. Variance of the upper-level policy determines the granularity at which parameter space is queried. At the same time, it directly affects variability of the observed rewards. Therefore, reward variability gets entangled with exploration noise.

We proceed to examine the following one-dimensional toy problem

$$\underset{\mu_\theta, \sigma_\theta^2}{\text{maximize}} \quad -\frac{1}{\gamma} \log \mathbb{E}_{\theta \sim \mathcal{N}(\mu_\theta, \sigma_\theta^2)} \left[\mathrm{e}^{\gamma |\theta|} \right]. \tag{8}$$

Policy gradients of the objective (8), evaluated on a grid of parameter values μ_θ and σ_θ, are displayed in Fig. 2. A risk-neutral ($\gamma = 0$), risk-averse ($\gamma > 0$), and risk-seeking ($\gamma < 0$) scenarios are shown. Covariant parameterization [23] of the Gaussian density was used, i.e., $\mu_\theta = \omega_1$ and $\sigma_\theta = \exp(\omega_2)$; therefore, the displayed gradient directions coincide with the natural gradient directions for this problem. A crucial observation from Fig. 2 is that a risk-seeking policy update may increase exploration variance, whereas a risk-averse update always decreases it. In practical terms, risk-aversion may lead to premature convergence due to insufficient exploration. Optimism, on the other hand, may result in a better coverage of the search space by fostering exploration.

C. Connection to Relative Entropy Policy Search

In Sec. IV-A, we established a remarkable fact that risk-sensitive PG (6) yields a baseline-corrected gradient estimator (7) in the risk-neutral limit $\gamma \to 0$. It turns out, another important property of the gradient estimator (6) can be revealed by recognizing it as the gradient of the maximum likelihood (ML) policy update in REPS [8]. This renders our risk-sensitive policy update optimal in a certain information-theoretic sense, made precise below.

REPS belongs to the category of information-theoretic policy search approaches [2]. This class of methods follows the idea of limiting the loss of information in-between policy updates. The Kullback-Leibler (KL) divergence is commonly used as the measure of information loss. REPS can be framed as an EM-like algorithm, with the parameter update step given by the weighted ML fit [8]. At each iteration, the following optimization problem gets solved

$$\begin{aligned} \underset{\pi}{\text{maximize}} \quad & \int \pi(\theta) R(\theta) \, \mathrm{d}\theta \\ \text{subject to} \quad & \mathrm{KL}(\pi(\theta) \| q(\theta)) \le \varepsilon, \\ & \int \pi(\theta) \, \mathrm{d}\theta = 1. \end{aligned} \tag{9}$$

Conveniently, a closed-form solution can be found

$$\pi(\theta) = q(\theta) \exp\left(\frac{R(\theta) - \psi_{-1/\eta}(q)}{\eta} \right) \tag{10}$$

as a function of the Lagrange multiplier $\eta > 0$, which corresponds to the KL-bound in (9). Note the appearance of the log-partition function again, $\psi_{-1/\eta}(q) = \eta \log \mathbb{E}_q[\exp(R/\eta)]$. The optimal value of η is found by dual optimization

$$\eta^\star = \arg\min_{\eta > 0} \left\{ \eta \varepsilon + \psi_{-1/\eta}(q) \right\}. \tag{11}$$

Since only black box access to function $R(\theta)$ is assumed, Eq. (10) does not yield the new policy π as an explicit function of θ but rather only provides samples from it.

A parametric policy $\pi_\omega(\theta)$ is fitted to the samples obtained from (10) by moment projection [2]

$$\begin{aligned} & \underset{\omega}{\text{minimize}} \ \ \mathrm{KL}(\pi(\theta) \| \pi_\omega(\theta)) \\ \propto \ & \underset{\omega}{\text{maximize}} \ \ \mathbb{E}_{\theta \sim q} \left[\log \pi_\omega(\theta) \exp\left(\frac{R(\theta) - \psi_{-1/\eta}(q)}{\eta} \right) \right]. \end{aligned} \tag{12}$$

The gradient of (12) serves as the link to the risk-sensitive policy gradient (6). Indeed, compare

$$\nabla_\omega \mathrm{KL} = \mathbb{E}_{\theta \sim q} \left[\nabla \log \pi_\omega(\theta) \left\{ \mathrm{e}^{\eta^{-1}(R(\theta) - \psi_{-1/\eta}(q))} \right\} \right] \tag{13}$$

to the risk-sensitive gradient (6). The correspondence is established by identifying $\gamma = -1/\eta$ and noting that the argument in the curly braces in (13) is proportional to the one in (6) up to a scaling factor η.

The key difference between (6) and (13) is the sampling distribution. Whereas the risk-sensitive policy gradient (6) requires samples from π_ω, an auxiliary distribution q is used in REPS. In theory, it means that REPS can perform several gradient update steps according to (13) with the same samples from q, while the risk-sensitive gradient (6) requires gathering new data after each parameter update. However, in practice, optimizing the ML objective (12) till

convergence is problematic due to the finite sample size and the associated overfitting problems [2]. That is why alternatives to the policy update objective (12) are often used, such as performing the information projection instead of the moment projection [24], or constraining the policy fitting step with another KL divergence [25], which can be viewed as a form of maximum a posteriori estimation [26].

Thus, the policy update of REPS (12) can be identified with the risk-sensitive update (6) under the assumption that the information loss bound ε is small, such that $q \approx \pi_\omega$ and one step in the direction of the gradient (13) solves (12). Importantly, though, the temperature parameter $\eta = -1/\gamma$ gets optimized in REPS and thus changes with iterations, whereas when applying (6), it has to be scheduled manually.

Another interesting distinction between risk-sensitive optimization and REPS stems from the fact that the temperature parameter η must be positive in REPS. This means $\gamma < 0$, or risk-seeking optimization. Thus, REPS is risk-seeking by construction, unlike the risk-sensitive PG (6) which can also be risk-averse.

V. Experiments

To analyze the properties of the risk-sensitive policy gradient algorithm of Sec. IV-A, we first consider a prototypical risk-sensitive portfolio optimization problem to establish the validity of our approach, then we proceed to apply the risk-sensitive policy gradient method to a toy dynamical system that models a part of our robot badminton setup, and finally, we report the results obtained by applying the algorithm to a real-robot task of learning to return a shuttlecock in the game of badminton with the Barrett WAM robot.

A. Risk-Sensitive Portfolio Management

A basic problem of portfolio optimization can be described as follows [27]. An individual wants to invest a unit of capital in N assets with the goal of making profit. The distribution of capital over assets $\mathbf{x}$ is called portfolio; by definition, portfolio is normalized, $\sum_{i=1}^{N} x_i = 1$. The returns of various assets $\mathbf{r}$ are random variables and are assumed to be Gaussian distributed $\mathbf{r} \sim \mathcal{N}(\mu_\mathbf{r}, \Sigma_\mathbf{r})$. Then, return of a portfolio $\mathbf{x}$ is a random variable $R \sim \mathcal{N}(\mu_\mathbf{r}^\mathrm{T}\mathbf{x}, \mathbf{x}^\mathrm{T}\Sigma_\mathbf{r}\mathbf{x})$. Depending on the definition of 'making profit', different objective functions can be constructed. We explore the notion of optimality with respect to the exponential risk measure (2), which allows for controlling the mean-variance trade-off (and higher moments) by varying the risk-sensitivity factor γ.

To apply the risk-sensitive policy gradient approach from Sec. IV-A, a suitable policy must be defined. We let the lower-level policy output a portfolio $\mathbf{x}$, parameterized by the softmax $\mathbf{x} = \exp(\theta - \log(1^T \exp(\theta)))$. Parameters $\theta \in \mathbb{R}^N$ of the lower-level policy are sampled from a Gaussian upper-level policy $\pi_\omega(\theta) = \mathcal{N}(\theta|\mu_\theta, \Sigma_\theta)$ with $\omega = \{\mu_\theta, \Sigma_\theta\}$.

In simulation, the number of assets is set to $N = 30$. Parameters of the asset return distribution are sampled evenly in the interval $\mu_\mathbf{r} \in [4, 0.5]$, $\sigma_\mathbf{r} \in [2, 0.01]$, with $\Sigma_\mathbf{r} = \mathrm{diag}(\sigma_\mathbf{r}^2)$. This distribution of parameters can be interpreted as follows. Returns with a high expected value are accompanied with higher risks, whereas lower risk returns yield lower mean reward. When comparing two policies π_1 and π_2 corresponding to risk factors $\gamma_1 > \gamma_2$, policy π_1 will prefer lower risk assets and yield lower return on average than π_2.

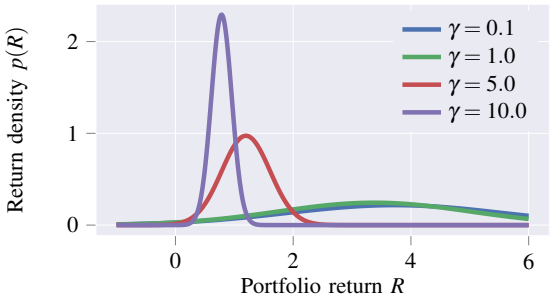


Fig. 3: Return distributions of optimal portfolios found with risk-sensitive policy gradient (6) for $\gamma \in \{0.1, 1, 5, 10\}$. The higher the risk-aversion factor $\gamma > 0$, the lower the variance of the returns; however, the mean is also lower in this case.

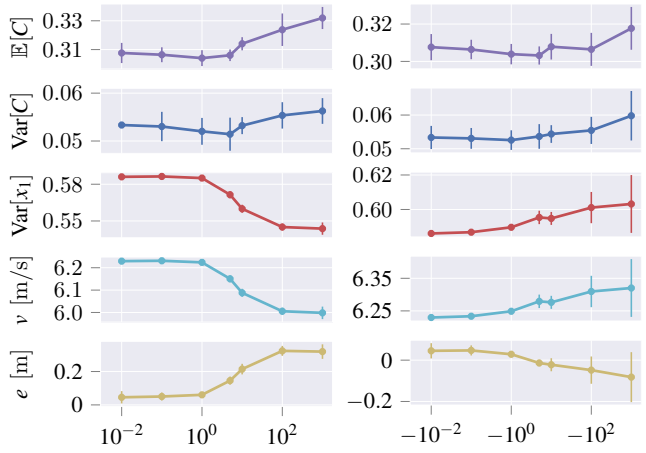
Return distributions for various values of the risk-aversion factor γ are shown in Fir. 3. Simulation was run over 10 random seeds with 1000 samples per trial. Results confirm the theory. A more pessimistic objective (higher γ) leads to a narrow reward distribution with lower mean, implying a smaller but more consistent reward. Smaller values of γ, and therefore more risk-neutral objectives, on the other hand, lead to an asset distribution that almost entirely aims to maximize the expected reward not taking the variance into account.

B. Toy Badminton System

We consider a simplified scenario of a robot learning to return a shuttlecock in the game of badminton. We assume a two dimensional world and a ball following a parabolic flight trajectory. The goal is to determine the hitting velocity of the racket which results in the ball arriving at a desired target location. The hypothesis is that for different values of the risk-aversion factor γ, the agent will learn different strategies: either aggressive hits but with high variability, or safe returns however with smaller expected reward. The problem is specified as follows

$$\begin{aligned} \underset{\omega}{\text{minimize}} \quad & \frac{1}{\gamma} \log \mathbb{E}\left[\exp(\gamma |x_\mathrm{des} - x_1)|)\right] \\ \text{subject to} \quad & x_1 = x_0 + v_{x,0} \left(\frac{v_{y,0}}{g} + \sqrt{\frac{v_{y,0}^2}{g^2} + \frac{2y_0}{g}} \right). \end{aligned} \tag{14}$$

The initial position of the ball (x_0, y_0) is assumed known. Due to a perfectly elastic collision, the initial velocity of the ball equals the velocity of the racket. Therefore, we treat the initial ball velocity as the control variable. Keeping in mind that this model should resemble the real-robot setup considered later, we add a bit of noise to the initial ball velocity, such that $(v_{x,0} \; v_{y,0}) = \mathbf{v}_0 \sim \mathcal{N}(\mathbf{u}, \Sigma_{\mathbf{v}_0})$ with $\mathbf{u}$ being our control variable. Constant g is the gravitational constant, and the equality constraint in (14) is derived from the equations of motion. Optimization variable $\omega = \{\mu_\mathbf{u}, \Sigma_\mathbf{u}\}$



(a) risk-averse, $\gamma > 0$ (b) risk-seeking, $\gamma < 0$

Fig. 4: Simulation results on the toy badminton system with a risk-sensitive objective optimized for varying risk factor values $\gamma \in \pm\{0.01, 0.1, 1, 5, 10, 100, 1000\}$. System noises are fixed $\sigma_{v_{x,0}} = \sigma_{v_{x,0}} = 0.6$ and the initial position is $x_0 = y_0 = 0$.

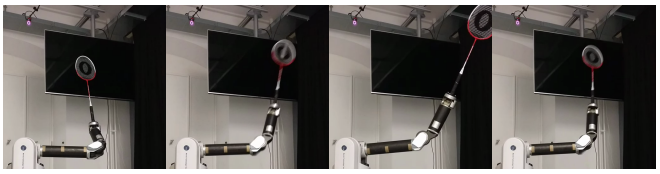
contains the parameters of the higher-level policy. As usually, we employ a Gaussian policy $\mathbf{u} \sim \pi_{\boldsymbol{\omega}}(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mu_{\mathbf{u}}, \Sigma_{\mathbf{u}})$.

Evaluation results on this simulated problem are shown in Fig. 4. Optimization was run over a range of values of γ with 1000 samples per trial, averaged over 10 random seeds. Error e is defined as $e = x_{\text{des}} - x_1$, the cost is $C = |e|$ and v is the initial speed. Several trends can be observed in Fig. 4. First, variance in the landing location x_1 is inversely proportional to risk-aversion: when risk-aversion increases, variance in the landing location decreases. However, such a clear trend cannot be observed in the variance of the cost function. Second, from the plot of the final position error e, we can read that both risk-seeking and risk-averse policies corresponding to extreme values of γ fail at returning the ball to the desired target. This effect is due to the dual nature of the objective function which trades mean performance agains variability. Extreme risk-averse policies tend to undershoot the target, while extreme risk-seeking ones tend to overshoot it. The same conclusion can be made based on the plot of velocities v. Risk-averse, pessimistic policies favor smaller initial velocities. In contrast, risk-seeking policies chose larger initial velocities. Third, variance bars are larger for large negative values of γ. This effect is due to objective (14) becoming very sharp, close to a delta function, which negatively affects optimization.

C. Robot Badminton

Finally, we proceed to apply the risk-sensitive policy gradient on a real robotic system consisting of a Barrett WAM supplied with an optical tracking setup and equipped with a badminton racket (see Fig. 1). The goal is to learn movement primitives of different levels of riskiness: on the scale between an aggressive smash and a defensive backhand.

To represent movements, we encode them using probabilistic movement primitives (ProMPs) [28]. Since a ProMP is given by a distribution over trajectories, generalization is



(a) rest (b) hit (c) swing (d) retract

Fig. 5: Phases of the hitting movement of the Barrett WAM.

accomplished through probabilistic conditioning, and trajectories can be shaped as desired by including via-points. We utilize these properties of ProMPs to learn hitting movements encoded by a small set of meta-parameters [20].

The meta parameters in our case encode a via-point defined by joint positions and velocities of the robot arm at a desired hitting time, $\boldsymbol{\theta} = [\mathbf{q}_{t,\text{hit}}^{\mathrm{T}} \quad \dot{\mathbf{q}}_{t,\text{hit}}^{\mathrm{T}}]^{\mathrm{T}}$. Using the extended Kalman filter (EKF), we process shuttlecock observations and predict its future state $\mathbf{s} = [\mathbf{x}_b^{\mathrm{T}} \quad \dot{\mathbf{x}}_b^{\mathrm{T}}]^{\mathrm{T}}$ at the interception plane. Control policy $\pi_{\boldsymbol{\omega}}(\boldsymbol{\theta}|\mathbf{s}) = \mathcal{N}(\boldsymbol{\theta}|\mathbf{M}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{s}), \Sigma_{\boldsymbol{\theta}})$ maps state $\mathbf{s}$ to meta-parameters $\boldsymbol{\theta}$, where random Fourier features $\boldsymbol{\phi}(\mathbf{s})$ are tuned as described in [29]. Reward function $R = -\sum_{i \in x,y,z} |x_{i,\text{ball}} - x_{i,\text{racket}}| - r_{\text{target}}$ is based on two terms: it encourages contact between the racket and the shuttlecock and it provides a bonus r_{target} if the shuttlecock reaches the desired target afterwards.

This problem falls into the realm of contextual policy search [2]. Therefore, we state the optimization objective as follows

$$\underset{\boldsymbol{\omega}}{\text{maximize}} \quad -\frac{1}{\gamma} \log \int \mu(\mathbf{s}) \int \pi_{\boldsymbol{\omega}}(\boldsymbol{\theta}|\mathbf{s}) e^{-\gamma R(\boldsymbol{\theta},\mathbf{s})} \, \mathrm{d}\boldsymbol{\theta} \, \mathrm{d}\mathbf{s}, \quad (15)$$

where $\mu(\mathbf{s})$ is the distribution over contexts $\mathbf{s}$, and $R(\boldsymbol{\theta}, \mathbf{s})$ is the reward given for the combination of $\mathbf{s}$ and $\boldsymbol{\theta}$.

Returning a shuttlecock in badminton to a desired location requires a high degree of precision. In our experiments, we had to relax the requirements due to constraints of the hardware. We only optimized for returning the shuttlecock at all and forced the projectile to follow the same trajectory for every iteration and always hit the same point at the interception plane.

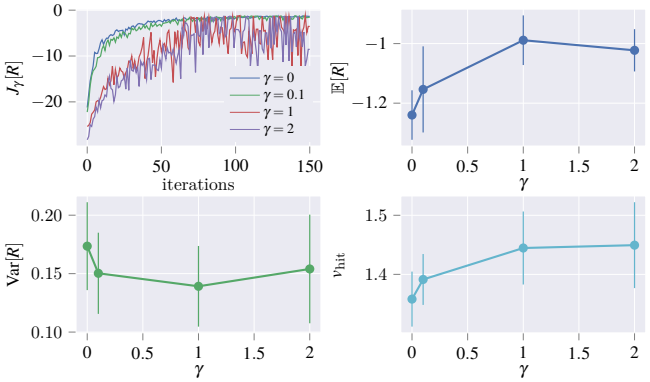


Fig. 6: The upper left plot shows the convergence curves for different values of $\gamma \geq 0$. The other three plots compare performance of the final learned policies in terms of expected return $\mathbb{E}[R]$, variance $\mathrm{Var}[R]$, and hitting velocity v_{hit}.

Evaluation results are shown in Fig. 6. Policies are optimized in simulation using 35 roll-outs per iteration over 150 iterations and experiments are repeated over 5 random seeds. Only risk-sensitive policies are shown, but it is noted that risk-seeking also yielded converging results. Although the convergence plots for positive values of γ look noisier (upper left plot), the final performance achieved by the policies trained with non-zero risk aversion (e.g., $\gamma = 1$ or $\gamma = 2$) is higher in terms of the expected reward (upper right plot). It is hard to make a conclusive judgement about the variance (lower left plot) due to high variability in the results. Surprisingly, the hitting velocity v_{hit} is actually higher for more risk-averse policies. This observation stands in contrast to what we had in the toy badminton model, where less aggressive policies favored smaller velocities.

Unfortunately, with our current setup, we were not able to achieve the goal of training skills of varying degree of riskiness. Nevertheless, we wanted to test the limits of achievable performance in the badminton task following a risk-neutral objective. The interception of the shuttlecock and hitting plane was enlarged to cover an area of approximately 1m^2. We carried out an extended learning trial in simulation with 100 roll-outs per iteration over 800 iterations. The best risk-neutral controller could return 95% of the served balls. The learned policy could be transferred to the real robot and was able to successfully return a shuttle cock. An example hitting movement is shown in Fig. 5.

VI. Conclusion

The entropic risk measure was considered as the optimization objective for policy gradient methods. By analyzing the exact form of its gradient, we found that it is related to the standard policy gradient but inherently incorporates a baseline. Furthermore, risk-sensitive policy update was shown to correspond to a certain limiting case of the policy update in REPS. Exploring this connection to information-theoretic methods appears to be a fruitful direction for future work. Entanglement between exploration variance and inherent system variability was found to be a strong limiting factor. Approaches for separating these two sources of uncertainty need to be searched for.

To reveal strengths and weaknesses of risk-sensitive optimization in a real robotic context, we applied our policy gradient method to the problem of learning risk-sensitive movement primitives in a badminton task. In a simplified model, we observed that policies optimized for different values of risk aversion demonstrate qualitatively different behaviors. Namely, risk-averse policies hit the shuttlecock with smaller velocity and tended to undershoot, whereas risk-seeking policies favored higher velocities and typically overshot the target. Finally, we carried out experiments on the real robot, which showed that moderate values of risk aversion can help finding better solutions for the original, risk-neutral problem. However, our attempt at learning risk-sensitive movement primitives on the real robot had limited success due to limitations of the hardware platform and the entanglement of sources of variability.

References

[1] R. E. Bellman, *Dynamic programming*. Princeton Univ. Press, 1957.

[2] M. P. Deisenroth, G. Neumann, J. Peters, *et al.*, "A survey on policy search for robotics," *Foundations and Trends® in Robotics*, vol. 2, no. 1–2, pp. 1–142, 2013.

[3] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3-4, pp. 229–256, 1992.

[4] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *NIPS*, 2000, pp. 1057–1063.

[5] S. M. Kakade, "A natural policy gradient," in *Advances in neural information processing systems*, 2002, pp. 1531–1538.

[6] J. Kober and J. Peters, "Learning motor primitives for robotics," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 2112–2118.

[7] J. Peters and S. Schaal, "Reinforcement learning by reward-weighted regression for operational space control," in *ICML*, 2007, pp. 745–750.

[8] J. Peters, K. Mülling, and Y. Altun, "Relative entropy policy search." in *AAAI*. Atlanta, 2010, pp. 1607–1612.

[9] H. Föllmer and T. Knispel, "Entropic risk measures: Coherence vs. convexity, model ambiguity and robust large deviations," *Stochastics and Dynamics*, vol. 11, no. 02n03, pp. 333–351, 2011.

[10] R. A. Howard and J. E. Matheson, "Risk-sensitive markov decision processes," *Management science*, vol. 18, no. 7, pp. 356–369, 1972.

[11] D. Jacobson, "Optimal stochastic linear systems with exponential performance criteria and their relation to deterministic differential games," *IEEE Transactions on Automatic control*, vol. 18, no. 2, pp. 124–131, 1973.

[12] P. Whittle, "A risk-sensitive maximum principle," *Systems & Control Letters*, vol. 15, no. 3, pp. 183–192, 1990.

[13] W. H. Fleming and W. M. McEneaney, "Risk sensitive optimal control and differential games," in *Stochastic theory and adaptive control*. Springer, 1992, pp. 185–197.

[14] A. Tamar, D. Di Castro, and S. Mannor, "Policy gradients with variance related risk criteria," in *ICML*, 2012, pp. 1651–1658.

[15] L. Prashanth, "Policy gradients for cvar-constrained mdps," in *ALT*, 2014, pp. 155–169.

[16] A. Tamar, "Risk-sensitive and efficient reinforcement learning algorithms," Ph.D. dissertation, Israel Institute of Technology, 2015.

[17] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[18] P. Whittle, "Risk sensitivity, a strangely pervasive concept," *Macroeconomic Dynamics*, vol. 6, no. 1, pp. 5–18, 2002.

[19] ——, "Risk-sensitive linear/quadratic/gaussian control," *Advances in Applied Probability*, vol. 13, no. 4, pp. 764–777, 1981.

[20] J. Kober, A. Wilhelm, E. Oztop, and J. Peters, "Reinforcement learning to adjust parametrized motor primitives to new situations," *Autonomous Robots*, vol. 33, no. 4, pp. 361–379, 2012.

[21] M. J. Wainwright, M. I. Jordan, *et al.*, "Graphical models, exponential families, and variational inference," *Foundations and Trends® in Machine Learning*, vol. 1, no. 1–2, pp. 1–305, 2008.

[22] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[23] D. Wierstra, T. Schaul, J. Peters, and J. Schmidhuber, "Natural evolution strategies," in *Congress on Evolutionary Computation*. IEEE, 2008, pp. 3381–3387.

[24] G. Neumann, "Variational inference for policy search in changing situations," in *ICML*, 2011, pp. 817–824.

[25] A. Abdolmaleki, B. Price, N. Lau, L. P. Reis, and G. Neumann, "Deriving and improving cma-es with information geometric trust regions," in *GECCO*, 2017, pp. 657–664.

[26] A. Abdolmaleki, J. T. Springenberg, Y. Tassa, R. Munos, N. Heess, and M. Riedmiller, "Maximum a posteriori policy optimisation," *arXiv preprint arXiv:1806.06920*, 2018.

[27] S. Boyd, E. Busseti, S. Diamond, R. N. Kahn, K. Koh, P. Nystrup, J. Speth, *et al.*, "Multi-period trading via convex optimization," *Foundations and Trends® in Optimization*, vol. 3, no. 1, pp. 1–76, 2017.

[28] A. Paraschos, C. Daniel, J. R. Peters, and G. Neumann, "Probabilistic movement primitives," in *NIPS*, 2013, pp. 2616–2624.

[29] A. Rajeswaran, K. Lowrey, E. V. Todorov, and S. M. Kakade, "Towards generalization and simplicity in continuous control," in *NIPS*, 2017, pp. 6550–6561.