Preference-balancing Motion Planning under Stochastic Disturbances

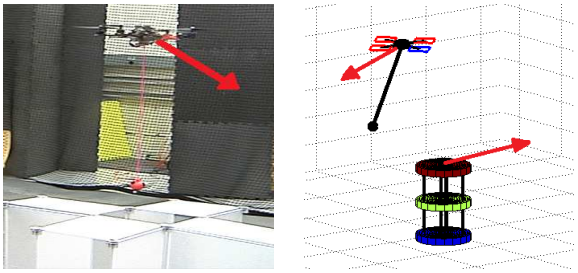
Aleksandra Faust[1,2], Nick Malone[1], and Lydia Tapia[1]

***Abstract*— Physical stochastic disturbances, such as wind, often affect the motion of robots that perform complex tasks in real-world conditions. These disturbances pose a control challenge because resulting drift induces uncertainty and changes in the robot's speed and direction. This paper presents an online control policy based on supervised machine learning, *Least Squares Axial Sum Policy Approximation* (LSAPA), that generates trajectories for robotic *preference-balancing tasks* under stochastic disturbances. The task is learned offline with reinforcement learning, assuming no disturbances, and then trajectories are planned online in the presence of disturbances using the current observed information. We model the robot as a stochastic control-affine system with unknown dynamics impacted by a Gaussian process, and the task as a continuous Markov Decision Process. Replacing a traditional greedy policy, LSAPA works for high-dimensional control-affine systems impacted by stochastic disturbances and is linear in the input dimensionality. We verify the method for Swing-free Aerial Cargo Delivery and Rendezvous tasks. Results show that LSAPA selects an input an order of magnitude faster than comparative methods, rejecting a range of stochastic disturbances. Further, experiments on a quadrotor demonstrate that LSAPA trajectories that are suitable for physical systems.**

I. INTRODUCTION

Real-world conditions pose many challenges to physical robots. One such challenge is the introduction of stochastic disturbances that can cause positional drift. These disturbances externally excite the system with a normally distributed intensity and direction, and their impact to the system varies between consecutive observations. For example, atmospheric changes, i.e. wind, are possible sources of stochastic disturbances [3]. Stochastic disturbances, along with complex nonlinear system dynamics, make traditional solutions (e.g., adaptive and robust control modeling), which solve this problem by explicitly solving the optimal control problem, difficult or intractable [15]. We are interested in a trajectory generation method that solves a particular class of robotic motion planning tasks, rejects stochastic disturbances, and is computationally efficient enough to be used on a high-dimensional system that requires frequent input selection (50 Hz).

We consider *preference-balancing tasks* [8], a class of robotic motion planning tasks. Preference-balancing tasks are robotic tasks characterized with a single goal state and a set of often-opposing preferences, such as speed and quality, that the robot should balance while meeting task conditions. There are many examples of preference-balancing tasks: swing-free aerial cargo delivery (Fig. 1a), balancing an inverted pendulum [9], coordinated meeting of two robots (Fig. 1b), etc. In each instance, there is a defined, possibly unknown, and recognizable goal state. Yet, it is difficult to manually identify a trajectory that solves the task without violating its preferences. An interesting characteristic of preference-balancing tasks is that humans can easily describe them and judge their quality, but they are difficult for humans to perform. To learn preference-balancing tasks, we model robots as a kinematic, control-affine systems (nonlinear dynamics, but linear in the input), that are controlled through acceleration. In our experience, this level of abstraction is enough for the trajectory generation, and a lower-level controller can be used to track the kinematic trajectory.



(a) Swing-free delivery (b) Rendezvous

Fig. 1. Preference-balancing task examples. Red arrows on robots are examples of stochastic external input disturbances at time step k after learning without external disturbances.

Our previous work, PrEference Appraisal Reinforcement Learning (PEARL) solves preference-balancing tasks on deterministic control-affine systems [8]. PEARL uses a batch reinforcement learning (RL) framework, which means that the learning and planning phases are separated. In the learning phase, PEARL uses Continuous Action Fitted Value Iteration (CAFVI) [11] with a state-value function approximated with a linear combination of features, a specifically selected functions of state. The state-value function is a discounted cumulative reward that can be obtained from a given state. The features are given as squared preferences (Section III). Interacting with a system simulator, CAFVI appraises the preferences and produces the feature weights. In the planning phase, PEARL generates a trajectory in real-time, selecting an input one step at the time with the Deterministic Axial Sum (DAS) policy approximation [11]. The closed loop control policy, DAS, takes the current state

[1]Computer Science Department, University of New Mexico, USA {afaust,nmalone,tapia}@cs.unm.edu

[2]Sandia National Laboratories, Albuquerque, New Mexico, USA afaust@sandia.gov

observation and relying on the learned weights, features, and simulator, produces the next input to be applied on the system. DAS scales linearly with the input dimensionality, and under verifiable conditions is guaranteed to progress the system to the goal [11], but it only works for deterministic systems.

In this paper, we extend DAS to work under stochastic input disturbances. To that end, we propose Least Squares Axial Sum Policy Approximation (LSAPA). LSAPA, used in the PEARL framework, enables learning preference-balancing tasks *without* disturbances, and preforming tasks *with* stochastic input disturbances. This is achieved by using supervised machine learning to find the best input with respect to the current disturbance. A Gaussian process, defined with its mean and variance, is used to model the disturbance of the input [3]. We assume that its probability distribution can be measured and estimated outside of LSAPA [30]. The estimation can be done, for example, equipping the system with an accelerometer, measuring the true acceleration of the system, and estimating the error between the observed and the LSAPA produced acceleration. The key extension from [11] is the use of least squares linear regression in lieu of interpolation to estimate near-optimal input on each axis. This extension allows us to apply the method to non-zero mean stochastic disturbances limited only by the system's physical limits.

This novel method, LSAPA, is applied in simulation to the Swing-free Aerial Cargo Delivery and multi-robot Rendezvous tasks (Fig. 1), and experimentally verified on Swing-free Aerial Cargo Delivery. The method is evaluated for computational efficiency, and trajectory quality by comparing it to DAS, Model Predictive Control (MPC), and *Hierarchical optimistic optimization applied to trees* (HOOT) [21]. A preliminary version of LSAPA was presented in an unpublished workshop paper [9]. Here, we extend that work with: 1) proof that the objective function is quadratic and has a single global maxima, 2) computational-efficiency analysis, 3) experiments on Swing-free Aerial Cargo-delivery, 4) comparison with MPC and HOOT, and 5) evaluation on the Rendezvous task.

II. RELATED WORK

Robot motion control under stochastic disturbance has been studied on a number of problems. For instance, quadrotor trajectory tracking under wind-gust disturbances was solved using piecewise linearization [1]. Path planning and obstacle avoidance in the presence of stochastic wind for a blimp was solved using dynamic programming and augmented MDPs [14]. In another example, methods to handle motion planning and trajectory generation under uncertainty use low-level controllers for stabilization of trajectories within reach tubes [7], or trajectory libraries [20]. Other approaches to UAV control in environments with a drift field explicitly solve the system dynamics [22], [26], or use iterative learning to estimate repetitive disturbance [25], [23]. While these solutions are aimed at particular systems or repetitive disturbances, our method works for a class of problems and systems influenced by the stochastic disturbance.

LSAPA, PEARL's control policy, shares both similarities and notable differences with Model Predictive Control (MPC) [13]. MPC takes a cost function, set points, and the system model, and generates a trajectory that minimizes cumulative cost over the receding horizon. Both PEARL and MPC are interested in selecting input that minimizes the cumulative cost. MPC solves the optimization problem numerically, while, PEARL constructs state-value function, a discounted, infinite-horizon, cumulative cost (reward) prior to task execution. PEARL's control policy, LSAPA, uses the state-value function during planning and only needs to solve a one-step optimization problem with respect to the state-value function. We use Nonlinear Model Predictive Control (NMPC) [13] to track a deterministic trajectory, and compare the results with the proposed LSAPA method.

DAS [11] solves input selection in linear time using the divide and conquer. It finds optimal input for each of the input axes independently with Lagrangian interpolation, and then combines the single-axis selections. Although we showed that DAS can compensate for some levels of zero-mean noise [11], [12], the method stops working in the presence of stochastic disturbances. This is because the external disturbance induces unpredictable drift onto the system. The method presented here, LSAPA, uses polynomial least squares regression instead of interpolation to compensate for the stochastic disturbances.

RL is well-suited for solving stochastic Markov decision processes (MDPs) and the current state of the art is expanding the methods to domains with large and continuous spaces, such as robotics [17]. Gradient descent methods for policy approximation work well in some convex cases, but require an estimate of the gradient, and can take a long time to converge. Some other methods in use are Gibbs sampling [16], Monte Carlo methods [18], and sample averages [2]. Finally, a new class of sampling methods optimistically narrows the search space [4], [5], [21], [31]. Of this class we compare LSAPA against *Hierarchical optimistic optimization applied to trees* (HOOT), [21] a derivative-free optimization that hierarchically discretizes the space into progressively smaller cells.

We use a quadrotor with a suspended load as our benchmarking platform because it is a popular research platform leading to solutions for multiple robots [27], hybrid system [6], differentially-flat approach [28], and load trajectory tracking [24] among others.

III. PROBLEM FORMULATION

Our goal is to plan preference-balancing tasks on a control-affine system in the presence of an external stochastic disturbance. Fig. 2 describes the planner's flow. We assume that a RL method provides a feature vector, $\boldsymbol{F}$, and weights, $\boldsymbol{\theta}$, learned with a *method without disturbances*, such as CAFVI. During the planning, we assume that we have a black-box simulator of the system, which receives mean and variance

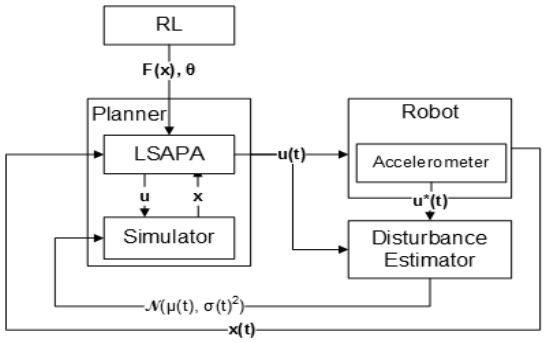


Fig. 2. Flow diagram for learning and planning preference-balancing tasks.

of the current probability distribution of the disturbance $\mathcal{N}(\mu_k, {\sigma_k}^2)$. The planner generates trajectories for a physical system. At every time step, k, LSAPA, observes a state, $\boldsymbol{x}_k$. By sampling the simulator, LSAPA finds a near-optimal input, $\boldsymbol{u}_k$, to apply to the system.

We model a robot as a discrete time, control-affine system with stochastic disturbance, $\boldsymbol{D} : X \times U \to X$,

$$\boldsymbol{D}: \quad \boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k) + \boldsymbol{g}(\boldsymbol{x}_k)(\boldsymbol{u}_k + \boldsymbol{\eta}_k). \tag{1}$$

States $\boldsymbol{x}_k \in X \subseteq \mathbb{R}^{d_x}$ belong to the position-velocity space and the control input is acceleration, $\boldsymbol{u}_k \in U \subseteq \mathbb{R}^{d_u}$. The input space is a compact set containing origin, $\mathbf{0} \in U$. The Lipschitz continuous function $\boldsymbol{g} : X \to \mathbb{R}^{d_x} \times \mathbb{R}^{d_u}$ is regular outside the origin, $\boldsymbol{x}_k \in X \setminus \{\mathbf{0}\}$. The drift $\boldsymbol{f} : X \to \mathbb{R}^{d_x}$, is bounded and Lipschitz continuous. The non-deterministic term, $\boldsymbol{\eta}_k$, is an independent and identically distributed random variable drawn from a Gaussian distribution $\mathcal{N}(\mu_{\boldsymbol{\eta}_k}, \sigma^2_{\boldsymbol{\eta}_k})$ known to the simulator, but not LSAPA; it acts as an additional and unpredictable external force on the system. Time step k is omitted when possible.

As in [11], our goal is to learn a preference-balancing task that takes the system to the origin in a timely-manner while reducing along the trajectory preferences given by matrix $\boldsymbol{A} = [\boldsymbol{a}_1 \dots \boldsymbol{a}_{d_g}]$. Each of the vectors $\boldsymbol{a}_i$ defines a task preference. For instance, vector $\boldsymbol{a}_i$ that corresponds to preference to reduce the displacement of the suspended load, will have components that correspond to the position of the suspended load set to one, while the rest of the components will be equal to zero. Vector $\boldsymbol{F}(\boldsymbol{x}) = [F_1(\boldsymbol{x}), ..., F_{d_g}(\boldsymbol{x})]^T$ is a feature vector, with components $F_i(\boldsymbol{x}) = \|\boldsymbol{a}_i^T \boldsymbol{x}\|^2, \quad i = 1, ..., d_g$.

The state-value function approximation is

$$V(\boldsymbol{x}) = \sum_{i=1}^{d_g} \theta_i F_i(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{A} \boldsymbol{\Theta} \boldsymbol{A}^T \boldsymbol{x} \tag{2}$$

where $\boldsymbol{\theta} = [\theta_1, ..., \theta_{d_g}]^T$ is the parametrization that we learn, and $\boldsymbol{\Theta}(\boldsymbol{x}) = \boldsymbol{\theta}^T \boldsymbol{I}_{d_g}$ is a diagonal matrix representation of the parametrization $\boldsymbol{\theta}$.

Greedy policy, $\boldsymbol{h}^*(\boldsymbol{x}) = \operatorname{argmax}_{\boldsymbol{u} \in U} V(D(\boldsymbol{x}, \boldsymbol{u}))$ is optimal with respect to the state-value function V. The problem is that in continuous spaces greedy policy calculation becomes an optimization problem over an unknown objective function $V \circ D$.

RL literature often works with *action-value function*, $Q : X \times U \to \mathbb{R}$, a measure of the discounted accumulated reward collected when action $\boldsymbol{u}$ is taken at state $\boldsymbol{x}$ [29]. In relation to the state-value function, V (2), action-value Q can be represented as

$$Q(\boldsymbol{x}, \boldsymbol{u}) = V(D(\boldsymbol{x}, \boldsymbol{u})) = \sum_{i=1}^{d_g} \theta_i \boldsymbol{F_i}(D(\boldsymbol{x}, \boldsymbol{u})) \tag{3}$$

Thus, we learn the approximation for the greedy policy using

$$\boldsymbol{h}^*(\boldsymbol{x}) = \operatorname*{argmax}_{\boldsymbol{u} \in U} Q(\boldsymbol{x}, \boldsymbol{u}), \tag{4}$$

and finding a near-optimal solution $\boldsymbol{h}(\boldsymbol{x})$ for (4).

IV. METHODS

The *Least squares axial policy approximation* (LSAPA) policy extends DAS to handle non-zero mean disturbances. This is done by first learning feature weights off-line *without* disturbances and then using those learned weights for online trajectory planning *with* disturbances. LSAPA bridges the gap between learning without disturbances and planning with them. The Lagrangian interpolation uses only three points to interpolate the underlying quadratic function and this compounds the error from the disturbances. In contrast, our new method, LSAPA, uses least squares regression with many sample points to compensate for the induced error.

Specifically DAS takes advantage of the facts that action-value function, Q, is a quadratic function of the input $\boldsymbol{u}$ for any fixed arbitrary state $\boldsymbol{x}$, in a control-affine system (1) with state-value approximation (2) [11]. DAS finds an approximation for the maximum local Q function for a fixed state x. It works in two steps, first finding maxima on each axis independently and then combining them together. To find a maximum on an axis, the method uses Lagrangian interpolation to find the coefficients of the quadratic polynomial representing the Q function. Then, an action that maximizes the Q function on each axis is found by zeroing the derivative. The final policy is a piecewise maximum of a convex and simple vector sums of the action maxima found on the axes. The method is computationally-efficient, scaling linearly with the action space dimensionality. It is also consistent, as the maximum selections do not depend on the selected samples. Because deterministic axial policies are sample independent, they do not adapt to changing conditions or external forces. We extend the deterministic axial policies to the presence of disturbances via LSAPA. LSAPA uses least squares regression, rather than Lagrangian interpolation, to select the maximum on a single axis. This change allows the LSAPA method to compensate for the error induced by non-zero mean disturbances.

We first show that the $\boldsymbol{Q}$ function remains quadratic with a maximum even when the system is influenced with a stochastic term.

***Proposition** 4.1:* Action-value function $Q(\boldsymbol{x}, \boldsymbol{u})$ (3) corresponding to state-value function V (2), and a discrete-time

system (1) is a quadratic function of input $\boldsymbol{u}$ for all states outside the origin, $\boldsymbol{x} \in X \backslash \{\mathbf{0}\}$. When $\boldsymbol{\Theta}$ is negative definite, the action-value function Q is concave and has a unique maximum.

Proof: The proof is similar to the proof of Proposition 3.1 [11], and relies on the fact that although the stochastic disturbance changes the system dynamics at every time step, it does not interact with the control input.

After rearranging terms we can write the dynamics (1) as

$$\boldsymbol{D}: \quad \boldsymbol{x}_{k+1} = \boldsymbol{f_1}(\boldsymbol{x}_k, \boldsymbol{\eta}_k) + \boldsymbol{g}(\boldsymbol{x}_k)\boldsymbol{u}_k,$$

where $\boldsymbol{f_1}(\boldsymbol{x}_k, \boldsymbol{\eta}_k) = \boldsymbol{f}(\boldsymbol{x}_k) + \boldsymbol{g}(\boldsymbol{x}_k)\boldsymbol{\eta}_k$ does not depend on input $\boldsymbol{u}_k$. Let us denote $\Lambda = \boldsymbol{A}\boldsymbol{\Theta}\boldsymbol{A}^T$. For an arbitrary state $\boldsymbol{x}$ and any value of the disturbance $\boldsymbol{\eta}$,

$$Q(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\eta}) = V(\boldsymbol{D}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\eta})) \tag{5}$$

$$= V(\boldsymbol{f_1}(\boldsymbol{x}, \boldsymbol{\eta}) + \boldsymbol{g}(\boldsymbol{x})\boldsymbol{u}) \tag{6}$$

$$= (\boldsymbol{f_1}(\boldsymbol{x}, \boldsymbol{\eta}) + \boldsymbol{g}(\boldsymbol{x})\boldsymbol{u}))^T \Lambda (\boldsymbol{f_1}(\boldsymbol{x}, \boldsymbol{\eta}) + \boldsymbol{g}(\boldsymbol{x})\boldsymbol{u}). \tag{7}$$

Thus, Q is a quadratic function of action $\boldsymbol{u}$ at any fixed state outside the origin, $\boldsymbol{x} \in X \setminus \{\mathbf{0}\}$, and fixed disturbance $\boldsymbol{\eta}$.

To show that Q has a maximum, we inspect Q's Hessian for fixed state $\boldsymbol{x}$ and disturbance $\boldsymbol{\eta}$,

$$HQ(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\eta}) = \begin{bmatrix} \frac{\partial^2 Q(\boldsymbol{x},\boldsymbol{u},\boldsymbol{\eta})}{\partial u_1 \partial u_1} & \dots & \frac{\partial^2 Q(\boldsymbol{x},\boldsymbol{u},\boldsymbol{\eta})}{\partial u_1 \partial u_{d_u}} \\ & \dots & \\ \frac{\partial^2 Q(\boldsymbol{x},\boldsymbol{u},\boldsymbol{\eta})}{\partial u_{d_u} \partial u_1} & \dots & \frac{\partial^2 Q(\boldsymbol{x},\boldsymbol{u},\boldsymbol{\eta})}{\partial u_{d_u} \partial u_{d_u}} \end{bmatrix}$$

$$= 2\boldsymbol{g}(\boldsymbol{x})^T \Lambda \boldsymbol{g}(\boldsymbol{x}),$$

which cancels the stochastic term, because the stochastic term does not affect square of the input $\boldsymbol{u}$ as seen in (7). Because $\boldsymbol{g}(\boldsymbol{x})$ is regular for all states $\boldsymbol{x} \in X \setminus \{\mathbf{0}\}$ and $\boldsymbol{\Theta} < 0$, the Hessian is negative definite, so Q is concave with a maximum for an arbitrary state outside of the origin. ■

Next, we present finding the maximum on ith axis using least squares linear regression with polynomial features.

***Definition** 4.2: Q-axial restriction* on i^{th} axis is a univariate function $Q_{\boldsymbol{x},i}(u) = Q(\boldsymbol{x}, u\boldsymbol{e_i})$, where $\boldsymbol{e_i}$ is a unit vector along of i^{th} axis.

Q-axial restriction on i^{th} axis is a quadratic function,

$$Q_{\boldsymbol{x},i}(u) = \boldsymbol{p}_i^T [u^2 \ u \ 1]^T, \tag{8}$$

for some vector $\boldsymbol{p}_i = [p_{2,i} \ p_{1,i} \ p_{0,i}]^T \in \mathbb{R}^3$ based on Proposition 4.1. Our goal is to find $\boldsymbol{p}_i$ by sampling the input space U at fixed state.

Suppose, we collect d_n input samples in the i^{th} axis, $U_i = [u_{1,i} \ ... \ u_{d_n,i}]^T$. The simulator returns state outcomes when the input samples are applied to the fixed state $\boldsymbol{x}$, $X_i = [\boldsymbol{x}'_{1,i} \ ... \ \boldsymbol{x}'_{d_n,i}]^T$, where $\boldsymbol{x}'_{j,i} \leftarrow D(\boldsymbol{x}, u_{j,i}), \quad j = 1, ..., d_n$. Next, Q-estimates are calculated with (3),

$$\boldsymbol{Q_i} = [Q_{\boldsymbol{x},1}(u_{1,i}) \ ... \ Q_{\boldsymbol{x},d_n}(u_{d_n,i})]^T, \tag{9}$$

where $Q_{\boldsymbol{x},j}(u_{j,i}) = \theta^T F(\boldsymbol{x}'_{j,i}), \quad j = 1, ..., d_n$. Using the supervised learning terminology the Q estimates, $\boldsymbol{Q_i}$, are the labels that match the training samples U_i. Matrix,

$$C_i = \begin{bmatrix} (u_{1,i})^2 & u_{1,i} & 1 \\ (u_{2,i})^2 & u_{2,i} & 1 \\ & \dots & \\ (u_{d_n,i})^2 & u_{d_n,i} & 1 \end{bmatrix}, \tag{10}$$

contains the training data projected onto the quadratic polynomial space. The solution to the supervised machine learning problem,

$$C_i \boldsymbol{p}_i = \boldsymbol{Q_i} \tag{11}$$

fits $\boldsymbol{p}_i$ into the training data C_i and labels $\boldsymbol{Q_i}$. The solution to (11),

$$\hat{\boldsymbol{p}}_i = \underset{\boldsymbol{p}_i}{\operatorname{argmin}} \sum_{j=1}^{d_n} (C_{j,i}\boldsymbol{p}_i - Q_{\boldsymbol{x},j}(u_{j,i}))^2 \tag{12}$$

is the coefficient estimate of the Q-axial restriction (8). A solution to $\frac{dQ_{\boldsymbol{x},i}(u)}{du} = 0$ is a critical point, and because Q is quadratic the critical point is

$$\hat{u}_i^* = -\frac{\hat{p}_{1,i}}{2\hat{p}_{2,i}}. \tag{13}$$

Lastly, we ensure that action selection falls within the allowed action limits,

$$\hat{u}_i = \min(\max(\hat{u}_i^*, u_i^l), u_i^u), \tag{14}$$

where u^l and u^u are lower and upper acceleration bounds on the i^{th} axis, respectively.

Repeating the process of estimating the maxima on all axes and obtaining $\hat{\boldsymbol{u}}_{\boldsymbol{i}} = [\hat{u}_1, ..., \hat{u}_{d_u}]$, we calculate the final policy with

$$\hat{\boldsymbol{h}}(\boldsymbol{x}) = \begin{cases} \boldsymbol{h}_{\boldsymbol{c}}^Q(\boldsymbol{x}), & Q(\boldsymbol{x}, \boldsymbol{h}_{\boldsymbol{c}}^Q(\boldsymbol{x})) \geq Q(\boldsymbol{x}, \boldsymbol{h}_{\boldsymbol{n}}^Q(\boldsymbol{x})) \\ \boldsymbol{h}_{\boldsymbol{n}}^Q(\boldsymbol{x}), & \text{otherwise} \end{cases} \tag{15}$$

where

$$\boldsymbol{h}_{\boldsymbol{c}}^Q(\boldsymbol{x}) = d_u^{-1} \boldsymbol{h}_{\boldsymbol{n}}^Q(\boldsymbol{x}) \qquad \text{(convex policy)}$$

$$\boldsymbol{h}_{\boldsymbol{n}}^Q(\boldsymbol{x}) = \sum_{i=1}^{d_u} \hat{u}_i \boldsymbol{e_i}, \qquad \text{(non-convex policy)}$$

The policy approximation (15) combines the simple vector sum of the non-convex policies (14) with the convex sum policy. The convex sum guarantees the system's monotonic progression towards the goal for a deterministic system [11], but the simple vector sum (non-convex policy) does not [11]. If, however, the vector sum performs better than the convex sum policy, then (15) allows us to use the better result. Disturbance changes $\boldsymbol{Q}$ function and the regression fits Q to the observed data. Thus, the algorithm adapts.

***Proposition** 4.3:* The computational cost to calculating LSAPA with (15) is $\mathcal{O}(d_g \cdot d_u \cdot d_x^2 \cdot d_n)$, assuming that $d_g \leq d_u \leq d_x \ll d_n$.

Proof: Complexity of calculating $\boldsymbol{x}'_{j,i}$ for one-dimensional input is $\mathcal{O}(d_x)$. Since $\boldsymbol{F}$ can be calculated in $\mathcal{O}(d_x)$, the complexity of (9) is $\mathcal{O}(d_g \cdot d_n \cdot d_x^2)$. Formulating matrix C_i in (10) is $\mathcal{O}(d_n)$. The solution to the regression problem (11) and (12) is asymptotically $O(d_n)$ [19], since we use polynomial of the second degree for the regression. Thus

the asymptotic complexity of calculating (13) is $\mathcal{O}(d_g \cdot d_n \cdot d_x^2)+\mathcal{O}(d_n)+\mathcal{O}(d_n) = \mathcal{O}(d_g \cdot d_n \cdot d_x^2)$. Finally, the complexity of the final input selection (14) is $\mathcal{O}(d_g \cdot d_u \cdot d_x^2 \cdot d_n)$ after repeating the process for all axes. ■

Therefore, LSAPA's running time depends on the state and input dimensions rather than their physical size. On the other hand, its running time depends on the number of features d_g and samples d_n. Since the number of features is small and fixed to the task, their impact to the complexity is minimal. However, the algorithm is sensitive to number of collected samples. In [9], we preformed a study that examined the optimal number of samples to be used for the Flying Inverted Pendulum task. The results showed that the trajectory quality increases exponentially with the sample size, and beyond a certain level the trajectories do not improve significantly. Due to LSAPA's complexity dependence on the sample size, it is useful to perform a one-time empirical study to find the optimal sample size, which produces good-quality trajectories and retains fast run time.

V. RESULTS

We evaluate LSAPA's 1) suitability for real-time planning of high-dimensional control-affine discrete time systems that require frequent input, 2) ability to complete preference-balancing tasks in the presence of different stochastic input disturbances, and 3) trajectories for feasibility on physical systems. All simulations are performed in MATLAB 2014a, on a single core of Intel Xeon CPU E5-1620 with 8 Gbs RAM running Windows 7 Enterprise. Experiments were performed on an AscTec Hummingbird Quadrotor equipped with a 62-centimeter suspended load weighing 45 grams. The quadrotor and load positions were captured with a motion capture system at 100 Hz. Testbed's internal tracking system tracked the quadrotor position using LSAPA generated trajectory as a reference.

A. Setup and Methodology

The *Swing-free Aerial Cargo Delivery task* requires a quadrotor, carrying a load on a suspended rigid cable, to deliver the cargo to a given location with the minimal residual load oscillations [10]. The task has applications in delivery supply and aerial transportation in urban environments. The task is easily described, yet, it is difficult for human demonstration as it requires a careful approach to avoid destabilizing the load. The state space is 10-dimensional joint system, $\boldsymbol{x} = [\boldsymbol{p}_q, \boldsymbol{\eta}, \dot{\boldsymbol{p}}_q, \dot{\boldsymbol{\eta}}]^T$, of the quadrotor's and the load's position and velocity, where $\boldsymbol{p}_q = [x, y, z]^T$ is the quadrotor's position, and $\boldsymbol{\eta} = [\phi, \psi]^T$ is the load's displacement in polar coordinates. The input is a 3-dimensional acceleration vector applied to the quadrotor's center of mass, $\boldsymbol{u} = [\ddot{x}, \ddot{y}, \ddot{z}]^T$ with a maximum acceleration of 3 m/s^2. The features are squared distances of quadrotor position, $F_1(\boldsymbol{x}) = \|\boldsymbol{p}_q\|^2$, load's displacement, $F_2(\boldsymbol{x}) = \|\boldsymbol{\eta}^2\|$, quadrotor's velocity, $F_3(\boldsymbol{x}) = \|\dot{\boldsymbol{p}}_q\|^2$, and the load's angular velocity, $F_4(\boldsymbol{x}) = \|\dot{\boldsymbol{\eta}}\|^2$.

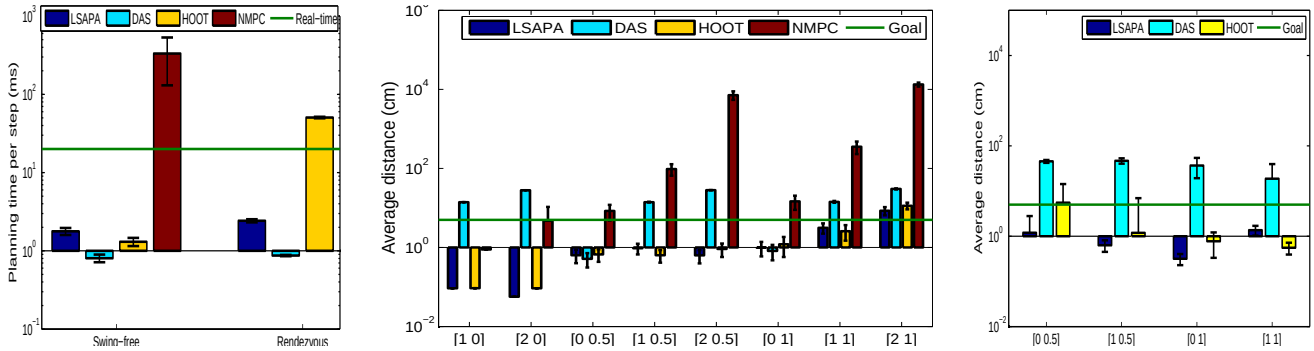
The *Rendezvous task* involves two heterogeneous robots, an aerial vehicle with suspended load, and a ground robot. The two robots work together for the cargo to be delivered on top of the ground robot (Fig. 1b), thus the load must have minimum oscillations when they meet. The state space is a 16-dimensional vector of the joint UAV-load-ground robot position-velocity space, $\boldsymbol{x} = [\boldsymbol{p}_q, \boldsymbol{p}_g, \boldsymbol{\eta}, \dot{\boldsymbol{p}}_q, \dot{\boldsymbol{p}}_g, \dot{\boldsymbol{\eta}}]^T$, and the action set is 5 dimensional acceleration vector on the UAV (3 dimensions), and the ground robot (2 dimensions), $\boldsymbol{u} = [\ddot{\boldsymbol{p}}_q, \ddot{\boldsymbol{p}}_g]^T$. The maximum acceleration of the UAV is $3\,m/s^2$, while the maximum acceleration of the ground robot is $2\,m/s^2$. Feature vector $\boldsymbol{F}$ contains: $F_1(\boldsymbol{x}) = \|\boldsymbol{p}_{q_{xy}} - \boldsymbol{p}_{g_{xy}}\|^2$, the distance between the ground and aerial robot's load x and y coordinates, $F_2(\boldsymbol{x}) = \|\boldsymbol{p}_{q_z} - \boldsymbol{p}_{g_z} - 0.6\|^2$, the difference in high equal to the suspension cable length, $F_3(\boldsymbol{x}) = \|\dot{\boldsymbol{p}}_q - \dot{\boldsymbol{p}}_g\|^2$, their relative speeds, and $F_4(\boldsymbol{x})\| = \boldsymbol{\eta}\|^2$ and $F_5(\boldsymbol{x}) = \|\dot{\boldsymbol{\eta}}\|^2$ the load's position and velocity relative to the UAV.

First, we learn the tasks using deterministic CAFVI, which results in the weights $\boldsymbol{\theta} = [-86290\ -350350\ -1430\ -1160]^T$ for Swing-free Aerial Cargo Delivery, and $\boldsymbol{\theta} = [-92256\ -44767\ -866\ -336\ -107]^T$ for Rendezvous. After a single deterministic learning phase, we generate trajectories for 25 different initial conditions using the learned weights, $\boldsymbol{\theta}$, and varying disturbance distributions. The initial condition for Swing-free Aerial Cargo Delivery are within $5\,m$ from the goal, and Rendezvous has the two robots start from within $10\,m$ from each other. Input disturbance distributions are evaluated with a mean of 0, 1, and $2\,m/s^2$, and a standard deviation of 0, 0.5 and $1\,m/s^2$. We run Rendezvous with mean disturbance of up to 1 m/s^2 because the maximum acceleration of the ground robot is $2\,m/s^2$ and the system cannot compensate for the larger disturbance.

We compare the proposed LSAPA to a DAS, HOOT, and NMPC for Aeriel Cargo Delivery and to to a DAS, HOOT for the Rendezvous task. HOOT uses three-level hierarchical search, with each level providing ten times finer discretization. NMPC tracks a trajectory generated assuming no disturbances. It is implemented using the MATLAB routine provided in [13] with a 5 time-step long horizon, and cost function $J(\boldsymbol{x}, \boldsymbol{u}) = \mathrm{E}[\|\boldsymbol{p'} - \boldsymbol{p}_r\|^2 + 0.1 \cdot \|\dot{\boldsymbol{p'}} - \dot{\boldsymbol{p}}_r\|^2])$, where $\boldsymbol{p'}$ is position of the state that results when input $\boldsymbol{u}$ is applied to $\boldsymbol{x}$. $\boldsymbol{p_r}$ is a position at the reference trajectory. The expectation is calculated as an average of 100 samples. NMPC uses the same disturbance-aware simulator used for LSAPA, DAS, and HOOT. The simulator calculates closed-loop optimization problem and simulates the plan.

B. Suitability for online input selection

Fig. 3 summarizes the planning results. Results in Fig. 3a show that for both tasks the time needed to calculate next input with LSAPA is an order of magnitude smaller than the $20\,ms$ time step (green line), allowing ample time to plan the trajectory in a real-time closed feedback loop. DAS calculates the next input faster than LSAPA. This is expected because the deterministic policy uses 3 samples



(a) Planning time (b) Swing-free distance (c) Rendezvous

Fig. 3. Summary of planning results with LSAPA, DAS, HOOT, NMPC policies for Swing-free Delivery and Rendezvous tasks averaged over 25 trials. Time needed to select a single action (a), and distance from the goal during the last 1 second of the flight for Swing-free Delivery (b) and Rendezvous (c) tasks. [n, m] signifies disturbance with $\mathcal{N}(m, n^2)$. Y-axes are logarithmic. Results below green line are suitable for real-time application (a), and complete the tasks (b) and (c).

per input dimension, while the stochastic policy in this case uses 300 samples. In contrast, although HOOT performs under $20\,ms$ for the Swing-free task, it scales exponentially with the input. As a result, HOOT takes $50\,ms$ to calculate input for the Rendezvous task, twice the length of the minimal time step required for real time planning. NMPC is two orders of magnitude slower for the lower-dimensional Swing-free task, averaging about $300\,ms$ to select an input, over ten times that the available window for the real-time control. NMPC for the Rendezvous task took 10 times longer than for the Swing-free task, requiring about 3 hours to calculate a single 15-second trajectory. Thus, we decided against running systematic NMPC tests with Rendezvous task because of the impractically long computational time.

Both LSAPA and DAS are computationally cheap, linear in the input dimensionality, while HOOT, and NMPC scale exponentially. The timing results show that, assuming LSAPA provides good quality trajectories, LSAPA can be a viable method for input selection in real-time on high-dimensional systems that require high-frequency control.

C. Trajectory quality in presence of input disturbances

Next, in Figs. 3b and 3c, we examine if the trajectories complete the task, reaching the goal region of $5\,cm$. Due to the constant presence of the disturbance, we consider the average position of the quadrotor rather than simply expecting to reach the goal region. Note that the accumulated squared error, typically used to measure quality of tracking methods, is not appropriate for LSAPA, HOOT, and DAS because they generate trajectories on the fly and have no reference trajectory. Thus, we measure if the system arrives and stays near the goal. As a control case, we first run simulations for repeatable disturbance ($\mathcal{N}(1, 0^2)$ and $\mathcal{N}(2, 0^2)$). LSAPA, NMPC, and HOOT methods complete the task (Fig. 3b).

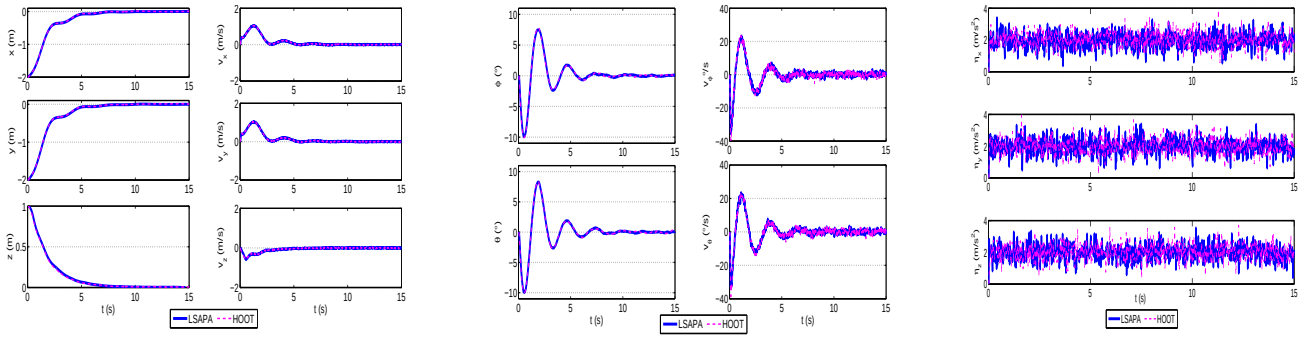
For a small standard deviation of 0.5 m/s^2 LSAPA performs similarly to HOOT on both tasks, producing trajectories that complete the task, unlike DAS and NMPC. DAS is able to compensate for zero-mean noise for the Swing-free task (Fig. 3b), but its performance degrades in the higher-dimensional Rendezvous (Fig. 3c). The quality of NMPC solution also degrades with the disturbance (Fig. 3b). It is more pronounced than with DAS, because NMPC makes input selection based on solving a fixed horizon optimization problem. The optimization problem accumulates the estimation error, thus invalidating the solution, and smaller horizon lengths are not sufficient to capture good tracking. For the larger standard deviation ($1\ m/s^2$), both LSAPA and HOOT create trajectories that still perform the tasks. But for Rendezvous, LSAPA produces the comparable results in an order of magnitude less time (Fig. 3a).

Figs. 4 and Fig. 5 show trajectories planned with LSAPA and HOOT for Swing-free task under exertion of disturbance with distribution $\mathcal{N}(2, 0.5^2)$ (Fig. 4c), and Rendezvous with disturbance with distribution $\mathcal{N}(1, 1^2)$. Although both the quadrotor's and the load's speeds are noisy (Figs. 4a and 4b), the position changes are smooth, and the quadrotor arrives near the goal position where it remains. Similarly, in Rendezvous the two robots meet in 4 seconds, after the initial failed coordinated slow down at 1 second. Note, that the targeted position for the quadrotor's altitude is 0.6 meters in order for the load to be positioned above the ground robot. The results in Fig. 3 confirm that both methods produce very similar trajectories, but recall that HOOT does not scale well for larger problems and did not produce Rendezvous trajectories in real-time. In contrast, LSAPA produced both trajectories in real-time.

D. Trajectory feasibility for Physical System

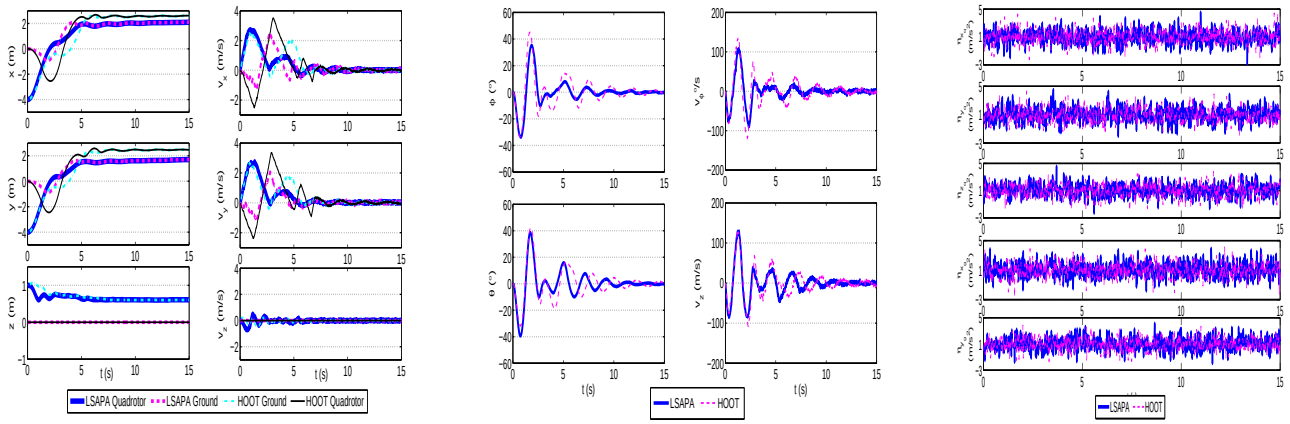
To evaluate how feasible the LSAPA trajectory is on a physical system and to assess the simulation fidelity, we compare experimentally LSAPA and DAS planned trajectories. We chose DAS for this experiment because it is the fastest input selection method, and it performed better than HOOT in previous experiments [11].

Fig. 6 shows the results of the experiment when a disturbance with distribution $\mathcal{N}(2, 0.5^2)$ is imposed on the system



(a) Quadrotor trajectory (b) Load trajectory (c) Injected disturbance

Fig. 4. Cargo delivery task - comparison of vehicle (a) and load (b) trajectories created with LSAPA and HOOT with disturbance of $\mathcal{N}(2, 0.5^2)$ (c).



(a) Quadrotor trajectory (b) Load trajectory (c) Injected disturbance

Fig. 5. Rendezvous task - comparison of vehicle (a) and load (b) trajectories created with LSAPA and HOOT with disturbance of $\mathcal{N}(1, 1^2)$ (c).

(Fig. 6c). The quadrotor starts at coordinates (-1, -1, 1.2) and the goal is at (0.5, 0.5, 1.2) meters. We notice (Fig. 6a) LSAPA experiences an overshoot of the goal after 2 seconds, but compensates and returns to the goal position. The DAS trajectory, however, does not compensate and continues with the slow drift past the goal. The load swing is not very different between the two trajectories. The enclosed supplied video contains the experiments and visualization of the simulations.

VI. CONCLUSIONS

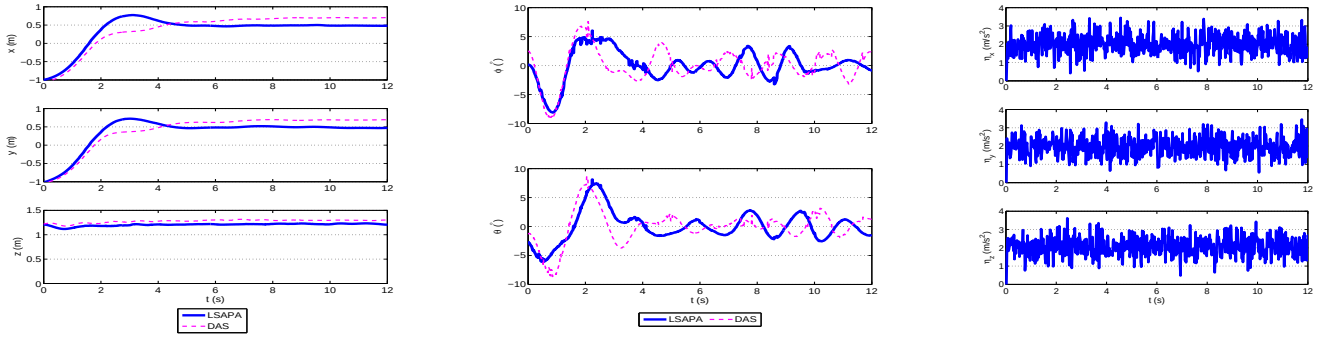
We presented LSAPA, a policy approximation method that extends PEARL to perform robotic preference-balancing tasks in environments with external stochastic disturbances. LSAPA scales linearly with the input dimensionality and produces good quality trajectories. Because of these characteristic LSAPA can be used to generate trajectories for high-dimensional control-affine systems that require frequent input, such as inherently unstable systems. We also showed that LSAPA can be used on physical robots. This paper takes an empirical approach to assess the safety of the policy. In future research, we will look into finding sufficient conditions for a goal's asymptotic stability, in order to be able predict the likelihood of system completing the task.

ACKNOWLEDGMENTS

The authors thank Peter Ruymgaart for helpful discussions on modeling the external disturbances, Particio Cruz for assisting with experiments, Angela Schoellig and Ivana Palunko for very useful feedback and discussions of Model Predictive Control, and John Baxter for comments on the manuscript. Faust was supported in part by NM Space Grant; Malone and Tapia were supported in part by the National Institutes of Health (NIH) Grant P20GM110907 to the Center for Evolutionary and Theoretical Immunology. Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energys National Nuclear Security Administration under contract DE-AC04-94AL85000.

REFERENCES

[1] K. Alexis, G. Nikolakopoulos, and A. Tzes. Constrained-control of a quadrotor helicopter for trajectory tracking under wind-gust disturbances. In *MELECON 2010-2010 15th IEEE Mediterranean Electrotechnical Conference*, pages 1411–1416. IEEE, 2010.



(a) Quadrotor trajectory (b) Load trajectory (c) Load trajectory

Fig. 6. Cargo delivery task: comparison of experimental vehicle (a) and load (b) LSAPA and DAS trajectories with disturbance $\mathcal{N}(2, 0.5^2)$ (c).

[2] A. Antos, C. Szepesvari, and R. Munos. Fitted Q-iteration in continuous action-space MDPs. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 9–16, Cambridge, MA, 2007. MIT Press.

[3] K. J. Astrom. *Introduction to Stochastic Control Theory*. Technology & Engineering. Academic Press, 1970.

[4] S. Bubeck, R. Munos, G. Stoltz, and C. Szepesvári. X-armed bandits. *J. Mach. Learn. Res.*, 12:1655–1695, July 2011.

[5] L. Busoniu, A. Daniels, R. Munos, and R. Babuska. Optimistic planning for continuous-action deterministic systems. In *2013 Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, in press 2013.

[6] F. F. Cedric de Crousaz and J. Buchli. Aggressive optimal control for agile flight with a slung load. In *Machine Learning in Planning and Control of Robot Motion Workshop at IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Chicago, IL*, September 2014.

[7] J. A. DeCastro and H. Kress-Gazit. Guaranteeing reactive high-level behaviors for robots with complex dynamics. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 749–756. IEEE, 2013.

[8] A. Faust. *Reinforcement Learning and Planning for Preference Balancing Tasks*. PhD thesis, University of New Mexico, Albuquerque, NM, July 2014.

[9] A. Faust, N. Malone, and L. Tapia. Planning preference-balancing motions with stochastic disturbances. In *Machine Learning in Planning and Control of Robot Motion Workshop at IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Chicago, IL*, September 2014.

[10] A. Faust, I. Palunko, P. Cruz, R. Fierro, and L. Tapia. Learning swing-free trajectories for uavs with a suspended load. In *IEEE International Conference on Robotics and Automation (ICRA), Karlsruhe, Germany*, pages 4887–4894, May 2013.

[11] A. Faust, P. Ruymgaart, M. Salman, R. Fierro, and L. Tapia. Continuous action reinforcement learning for control-affine systems with unknown dynamics. *Automatica Sinica, IEEE/CAA Journal of*, 1(3):323–336, July 2014.

[12] R. Figueroa, A. Faust, P. Cruz, L. Tapia, and R. Fierro. Reinforcement learning for balancing a flying inverted pendulum. In *Proc. The 11th World Congress on Intelligent Control and Automation*, July 2014.

[13] L. Grüne and J. Pannek. *Nonlinear Model Predictive Control: Theory and Algorithms*. Communications and Control Engineering. Springer, 1st ed. edition, 2011.

[14] H. Kawano. Study of path planning method for under-actuated blimp-type uav in stochastic wind disturbance via augmented-mdp. In *Advanced Intelligent Mechatronics (AIM), 2011 IEEE/ASME International Conference on*, pages 180–185, July 2011.

[15] H. Khalil. *Nonlinear Systems*. Prentice Hall, 1996.

[16] H. Kimura. Reinforcement learning in multi-dimensional state-action space using random rectangular coarse coding and gibbs sampling. In *Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS)*, pages 88–95, 2007.

[17] J. Kober, D. Bagnell, and J. Peters. Reinforcement learning in robotics: A survey. *International Journal of Robotics Research*, 32(11):1236–1272, 2013.

[18] A. Lazaric, M. Restelli, and A. Bonarini. Reinforcement learning in continuous action spaces through sequential monte carlo methods. *Advances in neural information processing systems*, 20:833–840, 2008.

[19] L. Li. A new complexity bound for the least-squares problem. *Computers & Mathematics with Applications*, 31(12):15 – 16, 1996.

[20] A. Majumdar and R. Tedrake. Robust online motion planning with regions of finite time invariance. In *Algorithmic Foundations of Robotics X*, pages 543–558. Springer, 2013.

[21] C. Mansley, A. Weinstein, and M. Littman. Sample-based planning for continuous action markov decision processes. In *Proc. of Int. Conference on Automated Planning and Scheduling*, 2011.

[22] A. A. Masoud. A harmonic potential field approach for planning motion of a uav in a cluttered environment with a drift field, Orlando, FL, USA. In *50th IEEE Conference on Decision and Control and European Control Conference*, pages 7665–7671, dec 2011.

[23] M. W. Mueller and R. D'Andrea. A model predictive controller for quadrocopter state interception. In *Control Conference (ECC), 2013 European*, pages 1383–1389. IEEE, 2013.

[24] I. Palunko, A. Faust, P. Cruz, L. Tapia, and R. Fierro. A reinforcement learning approach towards autonomous suspended load manipulation using aerial robots. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 4881–4886, May 2013.

[25] A. P. Schoellig, F. L. Mueller, and R. D'Andrea. Optimization-based iterative learning for precise quadrocopter trajectory tracking. *Autonomous Robots*, 33:103–127, 2012.

[26] S. Shen, N. Michael, and V. Kumar. Stochastic differential equation-based exploration algorithm for autonomous indoor 3d exploration with a micro-aerial vehicle. *I. J. Robotic Res.*, 31(12):1431–1444, 2012.

[27] K. Sreenath and V. Kumar. Dynamics, control and planning for cooperative manipulation of payloads suspended by cables from multiple quadrotor robots. In *Proceedings of Robotics: Science and Systems*, Berlin, Germany, June 2013.

[28] K. Sreenath, N. Michael, and V. Kumar. Trajectory generation and control of a quadrotor with a cable-suspended load – a differentially-flat hybrid system. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4873–4880, 2013.

[29] R. Sutton and A. Barto. *A Reinforcement Learning: an Introduction*. MIT Press, MIT, 1998.

[30] E. Todorov. Stochastic optimal control and estimation methods adapted to the noise characteristics of the sensorimotor system. *Neural Comput.*, 17(5):1084–1108, May 2005.

[31] T. J. Walsh, S. Goschin, and M. L. Littman. Integrating sample-based planning and model-based reinforcement learning. In M. Fox and D. Poole, editors, *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*, pages 612–617. AAAI Press, 2010.